Adrian-Horia Dediu
Carlos Martín-Vide
Ruslan Mitkov
Bianca Truthe (Eds.)

LNAI 7978

Statistical Language and Speech Processing

First International Conference, SLSP 2013
Tarragona, Spain, July 2013
Proceedings

Springer

Adrian-Horia Dediu   Carlos Martín-Vide
Ruslan Mitkov   Bianca Truthe (Eds.)

Statistical Language and Speech Processing

First International Conference, SLSP 2013
Tarragona, Spain, July 29-31, 2013
Proceedings

Springer

Volume Editors

Adrian-Horia Dediu
Carlos Martín-Vide
Universitat Rovira i Virgili
Research Group on Mathematical Linguistics
Avinguda Catalunya, 35, 43002 Tarragona, Spain
E-mail: {adrian.dediu, carlos.martin}@urv.cat

Ruslan Mitkov
University of Wolverhampton
Research Group in Computational Linguistics
Stafford St., Wolverhampton WV1 1SB, UK
E-mail: r.mitkov@wlv.ac.uk

Bianca Truthe
Otto-von-Guericke-Universität Magdeburg
Institut für Wissens- und Sprachverarbeitung
Universitätsplatz 2, 39106 Magdeburg, Germany
E-mail: truthe@iws.cs.uni-magdeburg.de

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-39592-5 e-ISBN 978-3-642-39593-2
DOI 10.1007/978-3-642-39593-2
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2013942556

CR Subject Classification (1998): I.2.7, I.2, I.5, G.3, F.4, H.3

LNCS Sublibrary: SL 7 – Artificial Intelligence

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

This volume contains the papers presented at the First International Conference on Statistical Language and Speech Processing (SLSP 2013), held in Tarragona, Spain, during July 29–31, 2013.

SLSP 2013 was the first event in a series to host and promote research on the wide spectrum of statistical methods that are currently in use in computational language or speech processing. SLSP aims to attract contributions from both fields. The conference encourages discussion on the employment of statistical methods (including machine learning) within language and speech processing. The scope of the SLSP series is rather broad, and includes the following areas: phonology, morphology; syntax, semantics; discourse, dialog, pragmatics; statistical models for natural language processing; supervised, unsupervised and semi-supervised machine learning methods applied to natural language, including speech; statistical methods, including biologically inspired methods; similarity; alignment; language resources; part-of-speech tagging; parsing; semantic role labeling; natural language generation; anaphora and coreference resolution; speech recognition; speaker identification/verification; speech transcription; text-to-speech synthesis; machine translation; translation technology; text summarization; information retrieval; text categorization; information extraction; term extraction; spelling correction; text and Web mining; opinion mining and sentiment analysis; spoken dialog systems; author identification, plagiarism and spam filtering.

SLSP 2013 received 61 submissions. Each submission was reviewed by three Program Committee members, some of them consulting with external reviewers as well. After a thorough and lively discussion, the committee decided to accept 24 papers (representing an acceptance rate of 39.34%). The program also included three invited talks.

Part of the success in the management of such a large number of submissions is due to the excellent facilities provided by the EasyChair conference management system. We would like to thank all invited speakers and authors for their contributions, the Program Committee and the reviewers for their cooperation, and Springer for its very professional publishing work.

May 2013

Adrian-Horia Dediu
Carlos Martín-Vide
Ruslan Mitkov
Bianca Truthe

Organization

SLSP 2013 was organized by the Research Group on Mathematical Linguistics (GRLMC) from the University Rovira i Virgili, Tarragona, Spain, and the Research Institute for Information and Language Processing (RIILP) from the University of Wolverhampton, UK.

Program Committee

Jerome Bellegarda	Apple Inc., Cupertino, USA
Robert C. Berwick	MIT, Cambridge, USA
Laurent Besacier	LIG, Grenoble, France
Bill Byrne	University of Cambridge, UK
Jen-Tzung Chien	National Chiao Tung University, Hsinchu, Taiwan
Kenneth Church	Thomas J. Watson Research Center, Yorktown Heights, USA
Koby Crammer	Technion, Haifa, Israel
Renato De Mori	McGill University, Montréal, Canada and University of Avignon, France
Thierry Dutoit	University of Mons, Belgium
Marcello Federico	Bruno Kessler Foundation, Trento, Italy
Katherine Forbes-Riley	University of Pittsburgh, USA
Sadaoki Furui	Tokyo Institute of Technology, Japan
Yuqing Gao	Thomas J. Watson Research Center, Yorktown Heights, USA
Ralph Grishman	New York University, USA
Dilek Hakkani-Tür	Microsoft Research, Mountain View, USA
Adam Kilgarriff	Lexical Computing Ltd., Brighton, UK
Dietrich Klakow	Saarland University, Saarbrücken, Germany
Philipp Koehn	University of Edinburgh, UK
Mikko Kurimo	Aalto University, Finland
Lori Lamel	CNRS-LIMSI, Orsay, France
Philippe Langlais	University of Montréal, Canada
Haizhou Li	Institute for Infocomm Research, Singapore
Qun Liu	Dublin City University, Ireland
Daniel Marcu	SDL, Los Angeles, USA

Carlos Martín-Vide (Co-chair)	University Rovira i Virgili, Tarragona, Spain
Ruslan Mitkov (Co-chair)	University of Wolverhampton, UK
Manuel Montes-y-Gómez	INAOE, Puebla, Mexico
Masaaki Nagata	NTT, Kyoto, Japan
Joakim Nivre	Uppsala University, Sweden
Kemal Oflazer	Carnegie Mellon University in Qatar, Doha, Qatar
Miles Osborne	University of Edinburgh, UK
Manny Rayner	University of Geneva, Switzerland
Giuseppe Riccardi	University of Trento, Italy
José A. Rodríguez Fonollosa	Technical University of Catalonia, Barcelona, Spain
Paolo Rosso	Technical University of Valencia, Spain
Mark Steedman	University of Edinburgh, UK
Tomek Strzalkowski	University at Albany, USA
Gökhan Tür	Microsoft Research, Redmond, USA
Stephan Vogel	Qatar Computing Research Institute, Doha, Qatar
Kuansan Wang	Microsoft Research, Redmond, USA
Dekai Wu	HKUST, Hong Kong
Min Zhang	Institute for Infocomm Research, Singapore
Yunxin Zhao	University of Missouri, Columbia, USA

External Reviewers

Karteek Addanki
Aviad Barzilay
Hadas Benisty
Lluís Formiga
Marc Franco-Salvador
Parth Gupta
Yun Huang
Itamar Katz
Ting Liu
Adrián Pastor López-Monroy
Fandong Meng
Fernando Sánchez-Vega
Zhaopeng Tu
Weibin Zhang
Daqi Zheng

Organizing Committee

Adrian-Horia Dediu, Tarragona
Carlos Martín-Vide, Tarragona (Co-chair)
Ruslan Mitkov, Wolverhampton (Co-chair)
Bianca Truthe, Magdeburg
Florentina-Lilica Voicu, Tarragona

Table of Contents

Deep Learning of Representations: Looking Forward

Yoshua Bengio

Department of Computer Science and Operations Research
Université de Montréal, Canada

Abstract. Deep learning research aims at discovering learning algorithms that discover multiple levels of distributed representations, with higher levels representing more abstract concepts. Although the study of deep learning has already led to impressive theoretical results, learning algorithms and breakthrough experiments, several challenges lie ahead. This paper proposes to examine some of these challenges, centering on the questions of scaling deep learning algorithms to much larger models and datasets, reducing optimization difficulties due to ill-conditioning or local minima, designing more efficient and powerful inference and sampling procedures, and learning to disentangle the factors of variation underlying the observed data. It also proposes a few forward-looking research directions aimed at overcoming these challenges.

1 Background on Deep Learning

Deep learning is an emerging approach within the machine learning research community. Deep learning algorithms have been proposed in recent years to move machine learning systems towards the discovery of multiple levels of representation. They have had important empirical successes in a number of traditional AI applications such as computer vision and natural language processing. See [10,17] for reviews and [14] and the other chapters of the book [95] for practical guidelines. Deep learning is attracting much attention both from the academic and industrial communities. Companies like Google, Microsoft, Apple, IBM and Baidu are investing in deep learning, with the first widely distributed products being used by consumers aimed at speech recognition. Deep learning is also used for object recognition (Google Goggles), image and music information retrieval (Google Image Search, Google Music), as well as computational advertising [36]. A deep learning building block (the *restricted Boltzmann machine*, or RBM) was used as a crucial part of the winning entry of a million-dollar machine learning competition (the Netflix competition) [115,134]. The New York Times covered the subject twice in 2012, with front-page articles.[1] Another series of articles (including a third New York Times article) covered a more recent event showing off the application of deep learning in a major Kaggle competition for

[1] `http://www.nytimes.com/2012/11/24/science/scientists-see-advances-in-deep-learning-a-part-of-artificial-intelligence.html`

drug discovery (for example see "Deep Learning - The Biggest Data Science Breakthrough of the Decade"[2]. Much more recently, Google bought out ("acqui-hired") a company (DNNresearch) created by University of Toronto professor Geoffrey Hinton (the founder and leading researcher of deep learning) and two of his PhD students, Ilya Sutskever and Alex Krizhevsky, with the press writing titles such as "Google Hires Brains that Helped Supercharge Machine Learning" (Robert McMillan for Wired, March 13th, 2013).

The performance of many machine learning methods is heavily dependent on the choice of data representation (or features) on which they are applied. For that reason, much of the actual effort in deploying machine learning algorithms goes into the design of preprocessing pipelines that result in a hand-crafted representation of the data that can support effective machine learning. Such feature engineering is important but labor-intensive and highlights the weakness of many traditional learning algorithms: their inability to extract and organize the discriminative information from the data. Feature engineering is a way to take advantage of human ingenuity and prior knowledge to compensate for that weakness. In order to expand the scope and ease of applicability of machine learning, it would be highly desirable to make learning algorithms less dependent on feature engineering, so that novel applications could be constructed faster, and more importantly for the author, to make progress towards artificial intelligence (AI).

A representation learning algorithm discovers explanatory factors or features. A deep learning algorithm is a particular kind of representation learning procedure that discovers *multiple levels of representation, with higher-level features representing more abstract aspects of the data.* This area of research was kick-started in 2006 by a few research groups, starting with Geoff Hinton's group, who initially focused on stacking unsupervised representation learning algorithms to obtain deeper representations [62,7,106,86]. Since then, this area has seen rapid growth, with an increasing number of workshops (now one every year at the NIPS and ICML conferences, the two major conferences in machine learning) and even a new specialized conference just created in 2013 (ICLR – the International Conference on Learning Representations).

Transfer learning is the ability of a learning algorithm to exploit commonalities between different learning tasks in order to share statistical strength, and *transfer knowledge* across tasks. Among the achievements of unsupervised representation learning algorithms are the impressive successes they obtained at the two transfer learning challenges held in 2011. First, the Transfer Learning Challenge, presented at an ICML 2011 workshop of the same name, was won using unsupervised layer-wise pre-training [11,92]. A second Transfer Learning Challenge was held the same year and won by [48] using unsupervised representation learning. Results were presented at NIPS 2011's Challenges in Learning Hierarchical Models Workshop.

[2] `http://oreillynet.com/pub/e/2538`

2 Quick Overview of Deep Learning Algorithms

The central concept behind all deep learning methodology is the automated discovery of abstraction, with the belief that more abstract representations of data such as images, video and audio signals tend to be more useful: they represent the semantic content of the data, divorced from the low-level features of the raw data (e.g., pixels, voxels, or waveforms). Deep architectures lead to abstract representations because more abstract concepts can often be constructed in terms of less abstract ones.

Deep learning algorithms are special cases of representation learning with the property that they learn multiple levels of representation. Deep learning algorithms often employ shallow (single-layer) representation learning algorithms as subroutines. Before covering the unsupervised representation learning algorithms, we quickly review the basic principles behind supervised representation learning algorithms such as the good old multi-layer neural networks. Supervised and unsupervised objectives can of course be combined (simply added, with a hyper-parameter as coefficient), like in [79]'s discriminative RBM.

2.1 Deep Supervised Nets, Convolutional Nets, Dropout

Before 2006, it was believed that training deep supervised neural networks [114] was too difficult (and indeed did not work). The first breakthrough in training them happened in Geoff Hinton's lab with unsupervised pre-training by RBMs [62], as discussed in the next subsection. However, more recently, it was discovered that one could train deep supervised nets by proper initialization, just large enough for gradients to flow well and activations to convey useful information [46,127].[3] Another interesting ingredient in the success of training the deep supervised networks of [46] (and later of [77]) is the presence of rectifying non-linearities (such as $\max(0, x)$) instead of sigmoidal non-linearities (such as $1/(1 + \exp(-x))$ or $\tanh(x)$). See [71,97] for earlier work on rectifier-like non-linearities. We return to this topic in Section 4. These good results with purely supervised training of deep nets seem to be especially clear when large quantities of labeled data are available, and it was demonstrated with great success for speech recognition [123,59,40] and object recognition [77] with breakthroughs reducing the previous state-of-the-art error rates by 30% to 50% on difficult to beat benchmarks.

One of the key ingredients for success in the applications of deep learning to speech, images, and natural language processing [9,35] is the use of *convolutional* architectures [83], which alternate *convolutional layers* and *pooling layers.* Units on hidden layers of a convolutional network are associated with a spatial or temporal position and only depend on (or generate) the values in a particular window of the raw input. Furthermore, units on convolutional layers share parameters with other units of the same "type" located at different positions, while at each location one finds all the different types of units. Units on pooling layers aggregate the outputs of units at a lower layer, either aggregating over different

[3] And potentially with the use of momentum [127].

nearby spatial positions (to achieve a form of local spatial invariance) or over different unit types. For example, a *max-pooling* unit outputs the maximum over some lower level units, which can therefore be seen to compete towards sending their signal forward.

Another key ingredient in the success of many recent breakthrough results in the area of object recognition is the idea of *dropouts* [64,77,52]. Interestingly, it consists in *injecting noise* (randomly dropping out units with probability 1/2 from the neural network during training, and correspondingly multiplying by 1/2 the weights magnitude at test time) that prevents a too strong co-adaptation of hidden units: hidden units must compute a feature that will be useful even when half of the other hidden units are stochastically turned off (masked). This acts like a powerful regularizer that is similar to bagging aggregation but over an exponentially large number of models (corresponding to different masking patterns, i.e., subsets of the overall network) that share parameters.

2.2 Unsupervised or Supervised Layer-Wise Pre-training

One of the key results of recent years of research in deep learning is that deep compositions of non-linearities – such as found in deep feedforward networks or in recurrent networks applied over long sequences – can be very sensitive to initialization (some initializations can lead much better or much worse results after training). The first type of approaches that were found useful to reduce that sensitivity is based on *greedy layer-wise pre-training* [62,7]. The idea is to train one layer at a time, starting from lower layers (on top of the input), so that there is a clear training objective for the currently added layer (which typically avoids the need for back-propagating error gradients through many layers of non-linearities). With unsupervised pre-training, each layer is trained to model the distribution of values produced as output of the previous layer. As a side-effect of this training, a new representation is produced, which can be used as input for deeper layers. With the less common *supervised* pre-training [7,141,124], each additional layer is trained with a supervised objective (as part of a one hidden layer network). Again, we obtain a new representation (e.g., the hidden or output layer of the newly trained supervised model) that can be re-used as input for deeper layers. The effect of unsupervised pre-training is apparently most drastic in the context of training deep *auto-encoders* [63], unsupervised learners that learn to reconstruct their input: unsupervised pre-training allows to find much lower training and test reconstruction error.

2.3 Directed and Undirected Graphical Models with Anonymous Latent Variables

Anonymous latent variables are latent variables that do not have a predefined semantics in terms of predefined human-interpretable concepts. Instead they are meant as a means for the computer to discover underlying explanatory factors present in the data. We believe that although non-anonymous latent variables can be very useful when there is sufficient prior knowledge to define them, anonymous latent variables are very useful to let the machine discover complex probabilistic structure: they lend flexibility to the model, allowing an otherwise parametric model to non-parametrically adapt to the amount of data when more anonymous variables are introduced in the model.

Principal components analysis (PCA), *independent components analysis* (ICA), and sparse coding all correspond to a directed graphical model in which the observed vector x is generated by first independently sampling some underlying factors (put in vector h) and then obtaining x by Wh plus some noise. They only differ in the type of prior put on h, and the corresponding *inference* procedures to recover h (its posterior $P(h \mid x)$ or expected value $\mathbb{E}[h \mid x]$) when x is observed. Sparse coding tends to yield many zeros in the estimated vector h that could have generated the observed x. See section 3 of [17] for a review of representation learning procedures based on directed or undirected graphical models.[4] Section 2.5 describes sparse coding in more detail.

An important thing to keep in mind is that directed graphical models tend to enjoy the property that in computing the posterior, the different factors *compete with each other*, through the celebrated *explaining away effect*. Unfortunately, except in very special cases (e.g., when the columns of W are orthogonal, which eliminates explaining away and its need), this results in computationally expensive inference. Although *maximum a posteriori* (MAP) inference[5] remains polynomial-time in the case of sparse coding, this is still very expensive, and unnecessary in other types of models (such as the stacked auto-encoders discussed below). In fact, exact inference becomes intractable for deeper models, as discussed in section 5.

Although RBMs enjoy tractable inference, this is obtained at the cost of a lack of explaining away between the hidden units, which could potentially limit the representational power of $\mathbb{E}[h \mid x]$ as a good representation for the factors that could have generated x. However, RBMs are often used as building blocks for training deeper graphical models such as the *deep belief network* (DBN) [62] and the *deep Boltzmann machine* (DBM) [118], which can compensate for the lack of explaining away in the RBM hidden units via a rich prior (provided by the upper layers) which can introduce potentially complex interactions and competition between the hidden units. Note that there is explaining away (and intractable exact inference) in DBNs and something analogous in DBMs.

2.4 Regularized Auto-Encoders

Auto-encoders include in their training criterion a form of reconstruction error, such as $||r(x) - x||^2$, where $r(\cdot)$ is the learned reconstruction function, often decomposed as $r(x) = g(f(x))$ where $f(\cdot)$ is an encoding function and $g(\cdot)$ a decoding function. The idea is that auto-encoders should have low reconstruction

[4] Directed and undirected: just two different views on the semantics of probabilistic models, not mutually exclusive, but views that are more convenient for some models than others.

[5] Finding h that approximately maximizes $P(h \mid x)$.

error at the training examples, but high reconstruction error in most other configurations of the input. In the case of auto-encoders, good generalization means that test examples (sampled from the same distribution as training examples) also get low reconstruction error. Auto-encoders have to be regularized to prevent them from simply learning the identity function $r(x) = x$, which would be useless. *Regularized auto-encoders* include the old bottleneck auto-encoders (like in PCA) with less hidden units than input, as well as the denoising auto-encoders [136] and contractive auto-encoders [112]. The denoising auto-encoder takes a noisy version $N(x)$ of original input x and tries to reconstruct x, e.g., it minimizes $||r(N(x)) - x||^2$. The contractive auto-encoder has a regularization penalty in addition to the reconstruction error, trying to make hidden units $f(x)$ as constant as possible with respect to x (minimizing the contractive penalty $||\frac{\partial f(x)}{\partial x}||_F^2$). A Taylor expansion of the denoising error shows that it is also approximately equivalent to minimizing reconstruction error plus a contractive penalty on $r(\cdot)$ [1]. As explained in [17], the tug-of-war between minimization of reconstruction error and the regularizer means that the intermediate representation must mostly capture the variations necessary to distinguish training examples, i.e., the directions of variations on the *manifold* (a lower dimensional region) near which the data generating distribution concentrates. *Score matching* [67] is an inductive principle that can be an interesting alternative to maximum likelihood, and several connections have been drawn between reconstruction error in auto-encoders and score matching [128]. It has also been shown that denoising auto-encoders and some forms of contractive auto-encoders estimate the score[6] of the underlying data generating distribution [135,1]. This can be used to endow regularized auto-encoders with a probabilistic interpretation and to sample from the implicitly learned density models [110,15,1] through some variant of Langevin or Metropolis-Hastings *Monte-Carlo Markov chains* (MCMC).

Even though there is a probabilistic interpretation to regularized auto-encoders, this interpretation does not involve the definition of intermediate anonymous latent variables. Instead, they are based on the construction of a direct parametrization of an encoding function which immediately maps an input x to its representation $f(x)$, and they are motivated by geometrical considerations in the spirit of manifold learning algorithms [17]. Consequently, there is no issue of tractability of inference, even with deep auto-encoders obtained by stacking single-layer ones.

It was previously believed [107], including by the author himself, that reconstruction error should only be small where the estimated density has a peak, e.g., near the data. However, recent theoretical and empirical results [1] show that the reconstruction error will be small where the estimated density has a peak (a mode) but also where it has a trough (a minimum). This is because the reconstruction error vector (reconstruction minus input) estimates the score $\frac{\partial \log p(x)}{\partial x}$, i.e., the reconstruction error is small where $||\frac{\partial \log p(x)}{\partial x}||$ is small. This can happen at a local maximum but also at a local minimum (or saddle point)

[6] Derivative of the log-density with respect to the data; this is different from the usual definition of score in statistics, where the derivative is with respect to the parameters.

of the estimated density. This argues against using reconstruction error itself as an *energy* function,[7] which should only be low near high probability points.

2.5 Sparse Coding and PSD

Sparse coding [100] is a particular kind of directed graphical model with a linear relationship between visible and latent variables (like in PCA), but in which the latent variables have a prior (e.g., Laplace density) that encourages sparsity (many zeros) in the MAP posterior. Sparse coding is not actually very good as a generative model, but has been very successful for unsupervised feature learning [104,31,142,55,72,2]. See [17] for a brief overview in the context of deep learning, along with connections to other unsupervised representation learning algorithms. Like other directed graphical models, it requires somewhat expensive inference, but the good news is that for sparse coding, MAP inference is a convex optimization problem for which several fast approximations have been proposed [89,53]. It is interesting to note the results obtained by [31] which suggest that sparse coding is a *better encoder but not a better learning algorithm* than RBMs and sparse auto-encoders (none of which has explaining away). Note also that sparse coding can be generalized into the spike-and-slab sparse coding algorithm [49], in which MAP inference is replaced by variational inference, and that was used to win the NIPS 2011 transfer learning challenge [48].

Another interesting variant on sparse coding is the *predictive sparse coding* (PSD) algorithm [73] and its variants, which combine properties of sparse coding and of auto-encoders. Sparse coding can be seen as having only a parametric "generative" decoder (which maps latent variable values to visible variable values) and a non-parametric encoder (find the latent variables value that minimizes reconstruction error and minus the log-prior on the latent variable). PSD adds a parametric encoder (just an affine transformation followed by a non-linearity) and learns it jointly with the generative model, such that the output of the parametric encoder is close to the latent variable values that reconstructs well the input.

3 Scaling Computations

From a computation point of view, how do we scale the recent successes of deep learning to much larger models and huge datasets, such that the models are actually richer and capture a very large amount of information?

3.1 Scaling Computations: The Challenge

The beginnings of deep learning in 2006 have focused on the MNIST digit image classification problem [62,7], breaking the supremacy of SVMs (1.4% error) on

[7] To define energy, we write probability as the normalized exponential of minus the energy.

this dataset.[8] The latest records are still held by deep networks: [29] currently claim the title of state-of-the-art for the unconstrained version of the task (e.g., using a convolutional architecture and stochastically deformed data), with 0.27% error.

In the last few years, deep learning has moved from digits to object recognition in natural images, and the latest breakthrough has been achieved on the ImageNet dataset.[9] bringing down the state-of-the-art error rate (out of 5 guesses) from 26.1% to 15.3% [77]

To achieve the above scaling from 28×28 grey-level MNIST images to 256×256 RGB images, researchers have taken advantage of convolutional architectures (meaning that hidden units do not need to be connected to all units at the previous layer but only to those in the same spatial area, and that pooling units reduce the spatial resolution as we move from lower to higher layers). They have also taken advantage of GPU technology to speed-up computation by one or two orders of magnitude [105,23,21,77].

We can expect computational power to continue to increase, mostly through increased parallelism such as seen in GPUs, multicore machines, and clusters. In addition, computer memory has become much more affordable, allowing (at least on CPUs) to handle potentially huge models (in terms of capacity).

However, whereas the task of recognizing handwritten digits is solved to the point of achieving roughly human-level performance, this is far from true for tasks such as general object recognition, scene understanding, speech recognition, or natural language understanding. What is needed to nail those tasks and scale to even more ambitious ones?

As we approach AI-scale tasks, it should become clear that our trained models will need to be much larger in terms of number of parameters. This is suggested by two observations. First, AI means understanding the world around us at roughly the same level of competence as humans. Extrapolating from the current state of machine learning, the amount of knowledge this represents is bound to be large, many times more than what current models can capture. Second, more and more empirical results with deep learning suggest that larger models systematically work better [30,64,77,52], provided appropriate regularization is used, such as the dropouts technique described above.

Part of the challenge is that the current capabilities of a single computer are not sufficient to achieve these goals, even if we assume that training complexity would scale linearly with the complexity of the task. This has for example motivated the work of the Google Brain team [81,39] to parallelize training of deep nets over a very large number of nodes. As we will see in Section 4, we hypothesize that as the size of the models increases, our current ways of training deep networks become less and less efficient, so that the computation required

[8] For the knowledge-free version of the task, where no image-specific prior is used, such as image deformations or convolutions, where the current state-of-the-art is around 0.8% and involves deep learning [111,64].

[9] The 1000-class ImageNet benchmark, whose results are detailed here: `http://www.image-net.org/challenges/LSVRC/2012/~results.html`

to train larger models (to capture correspondingly more information) is likely to scale much worse than linearly [38].

Another part of the challenge is that the increase in computational power has been mostly coming (and will continue to come) from parallel computing. Unfortunately, when considering very large datasets, our most efficient training algorithms for deep learning (such as variations on *stochastic gradient descent* or SGD) are inherently sequential (each update of the parameters requires having completed the previous update, so they cannot be trivially parallelized). Furthermore, for some tasks, the amount of available data available is becoming so large that it does not fit on a disk or even on a file server, so that it is not clear how a single CPU core could even scan all that data (which seems necessary in order to learn from it and exploit all of it, if training is inherently sequential).

3.2 Scaling Computations: Solution Paths

Parallel Updates: Asynchronous SGD. One idea that we explored in [6] is that of *asynchronous SGD*: train multiple versions of the model in parallel, each running on a different node and seeing different subsets of the data (on different disks), but with an asynchronous lock-free sharing mechanism which keeps the different versions of the model not too far from each other. If the sharing were synchronous, it would be too inefficient because most nodes would spend their time waiting for the sharing to be completed and would be waiting for the slowest of the nodes. This idea has been analyzed theoretically [108] and successfully engineered on a grand scale recently at Google [81,39]. However, current large-scale implementations (with thousands of nodes) are still very inefficient (in terms of use of the parallel resources), mostly because of the communication bottleneck requiring to regularly exchange parameter values between nodes. The above papers also take advantage of a way to train deep networks which has been very successful for GPU implementations, namely the use of rather large minibatches (blocks of examples after which an update is performed), making some parallelization (across the examples in the minibatch) easier. One option, explored by [32] is to use as building blocks for learning features algorithms such as k-means that can be run efficiently over large minibatches (or the whole data) and thus parallelized easily on a cluster (they learned 150,000 features on a cluster with only 30 machines).

Another interesting consideration is the optimization of trade-off between communication cost and computation cost in distributed optimization algorithms, e.g., as discussed in [132].

Sparse Updates. One idea that we propose here is to change the learning algorithms so as to obtain *sparse updates*, i.e., for any particular minibatch there is only a small fraction of parameters that are updated. If the amount of sparsity in the update is large, this would mean that a much smaller fraction of the parameters need to be exchanged between nodes when performing an asynchronous

SGD[10]. Sparse updates could be obtained simply if the gradient is very sparse. This gradient sparsity can arise with approaches that select paths in the neural network. We already know methods which produce slightly sparse updates, such as dropouts [64],[11] maxout [52][12] and other hard-pooling mechanisms, such as the recently proposed and very successful stochastic pooling [143]. These methods do not provide enough sparsity, but this could be achieved in two ways. First of all, we could choose to only pay attention to the largest elements of the gradient vector. Second, we could change the architecture along the lines proposed next.

Conditional Computation. A central idea (that applies whether one parallelizes or not) that we put forward is that of *conditional computation*: instead of dropping out paths independently and at random, drop them in a learned and optimized way. Decision trees remain some of the most appealing machine learning algorithms because prediction time can be on the order of the logarithm of the number of parameters. Instead, in most other machine learning predictors, scaling is linear (i.e., much worse). This is because decision trees exploit conditional computation: for a given example, as additional computations are performed, one can discard a gradually larger set of parameters (and avoid performing the associated computation). In deep learning, this could be achieved by combining *truly sparse activations* (values not near zero like in sparse auto-encoders, but actual zeros) and *multiplicative connections* whereby some hidden units *gate* other hidden units (when the gater output is zero it turns off the output of the gated unit). When a group A of hidden units has a sparse activation pattern (with many actual zeros) and it multiplicatively gates other hidden units B, then only a small fraction of the hidden units in B may need to be actually computed, because we know that these values will not be used. Such gating is similar to what happens when a decision node of a decision tree selects a subtree and turns off another subtree. More savings can thus be achieved if units in B themselves gate other units, etc. The crucial difference with decision trees (and e.g., the hard mixture of experts we introduced a decade ago [33]) is that the gating units should *not be mutually exclusive* and should instead form a *distributed pattern*. Indeed, we want to keep the advantages of distributed representations and avoid the limited local generalization suffered by decision trees [18]. With a high level of conditional computation, some parameters are used often (and are well tuned) whereas other parameters are used very rarely, requiring more data to estimate. A trade-off and appropriate regularization therefore needs to be

[10] Although the gain would be reduced considerably in a minibatch mode, roughly by the size of the minibatch.

[11] Where half of the hidden units are turned off, although clearly, this is not enough sparsity for reaching our objective; unfortunately, we observed that randomly and independently dropping a lot more than half of the units yielded substantially worse results.

[12] Where in addition to dropouts, only one out of k filters wins the competition in max-pooling units, and only one half of those survives the dropouts masking, making the sparsity factor $2k$.

established which will depend on the amount of training signals going into each parameter. Interestingly, *conditional computation also helps to achieve sparse gradients*, and the fast convergence of hard mixtures of experts [33] provides positive evidence that a side benefit of conditional computation will be easier and faster optimization.

Another existing example of conditional computation and sparse gradients is with the first layer of neural language models, deep learning models for text data [6,9]. In that case, there is one parameter vector per word in the vocabulary, but each sentence only "touches" the parameters associated with the words in the sentence. It works because the input can be seen as extremely sparse. The question is how to perform conditional computation in the rest of the model.

One issue with the other example we mentioned, hard mixtures of experts [33], is that its training mechanism only make sense when the gater operates at the output layer. In that case, it is easy to get a strong and clean training signal for the gater output: one can just evaluate what the error would have been if a different expert had been chosen, and train the gater to produce a higher output for the expert that would have produced the smallest error (or to reduce computation and only interrogate two experts, require that the gater correctly ranks their probability of being the best one). The challenge is how to produce training signals for gating units that operate in the middle of the model. One cannot just enumerate all the gating configurations, because in a distributed setting with many gating units, there will be an exponential number of configurations. Interestingly, this suggests *introducing randomness* in the gating process itself, e.g., stochastically choosing one or two choices out of the many that a group of gating units could take. This is interesting because this is the second motivation (after the success of dropouts as a regularizer) for re-introducing randomness in the middle of deep networks. This randomness would allow configurations that would otherwise not be selected (if only a kind of "max" dictated the gating decision) to be sometimes selected, thus allowing to accumulate a training signal about the value of this configuration, i.e., a training signal for the gater. The general question of *estimating or propagating gradients through stochastic neurons* is treated in another exploratory article [12], where it is shown that one can obtain an unbiased (but noisy) estimator of the gradient of a loss through a discrete stochastic decision. Another interesting idea explored in that paper is that of adding noise just before the non-linearity (max-pooling ($\max_i x_i$) or rectifier ($\max(0, x)$)). Hence the winner is not always the same, and when a choice wins it has a smooth influence on the result, and that allows a gradient signal to be provided, pushing that winner closer or farther from winning the competition on another example.

4 Optimization

4.1 Optimization: The Challenge

As we consider larger and larger datasets (growing faster than the size of the models), training error and generalization error converge. Furthermore many pieces of evidence in the results of experiments on deep learning suggest that training deep networks (including recurrent networks) involves a difficult optimization [13,56,16]. It is not yet clear how much of the difficulty is due to local minima and how much is due to ill-conditioning (the two main types of optimization difficulties in continuous optimization problems). It is therefore interesting to study the optimization methods and difficulties involved in deep learning, for the sake of obtaining better generalization. Furthermore, better optimization could also have an impact on scaling computations, discussed above.

One important thing to keep in mind, though, is that in a deep supervised network, the top two layers (the output layer and the top hidden layer) can rather easily be made to overfit, simply by making the top hidden layer large enough. However, to get good generalization, what we have found is that one needs to *optimize the lower layers*, those that are far removed from the immediate supervised training signal [7]. These observations mean that only looking at the training criterion is not sufficient to assess that a training procedure is doing a good job at optimizing the lower layers well. However, under constraints on the top hidden layer size, training error can be a good guide to the quality of the optimization of lower layers. Note that supervised deep nets are very similar (in terms of the optimization problem involved) to deep auto-encoders and to recurrent or recursive networks, and that properly optimizing RBMs (and more so deep Boltzmann machines) seems more difficult: progress on training deep nets is therefore likely to be a key to training the other types of deep learning models.

One of the early hypotheses drawn from experiments with layer-wise pre-training as well as of other experiments (semi-supervised embeddings [138] and slow feature analysis [140,22]) is that the training signal provided by backpropagated gradients is sometimes too weak to properly train intermediate layers of a deep network. This is supported by the observation that all of these successful techniques somehow inject a training signal into the intermediate layers, helping them to figure out what they should do. However, the more recent successful results with supervised learning on very large labeled datasets suggest that with some tweaks in the optimization procedure (including initialization), it is sometimes possible to achieve as good results with or without unsupervised pre-training or semi-supervised embedding intermediate training signals.

4.2 Optimization: Solution Paths

In spite of these recent encouraging results, several more recent experimental results again point to a fundamental difficulty in training intermediate and lower layers.

Diminishing Returns with Larger Networks. First, [38] show that with well-optimized SGD training, as the size of a neural net increases, the "return on investment" (number of training errors removed per added hidden unit) decreases, given a fixed number of training iterations, until the point where it goes below 1 (which is the return on investment that would be obtained by a brain-dead memory-based learning mechanism – such as Parzen Windows – which just copies an incorrectly labeled example into the weights of the added hidden unit so as to produce just the right answer for that example only). This suggests that larger models may be fundamentally more difficult to train, probably because there are now more second-order interactions between the parameters, increasing the condition number of the Hessian matrix (of second derivatives of model parameters with respect to the training criterion). This notion of return on investment may provide a useful metric by which to measure the effect of different methods to improve the scaling behavior of training and optimization procedures for deep learning.

Intermediate Concepts Guidance and Curriculum. Second, [56] show that there are apparently simple tasks on which standard black-box machine learning algorithms completely fail. Even *supervised and pre-trained deep networks* were tested and failed at these tasks. These tasks have in common the characteristic that the correct labels are obtained by the composition of at least two levels of non-linearity and abstraction: e.g., the first level involves the detection of objects in a scene and the second level involves a non-linear logical operation on top of these (such as the detecting presence of multiple objects of the same category). On the other hand, the task becomes easily solvable by a deep network whose intermediate layer is first pre-trained to solve the first-level sub-task. This raises the question of how humans might learn even more abstract tasks, and [13] studies the hypothesis that the use of language and the evolution of culture could have helped humans reduce that difficulty (and gain a serious advantage over other less cultured animals). It would be interesting to explore multi-agent learning mechanisms inspired by the the mathematical principles behind the evolution of culture in order to bypass this optimization difficulty. The basic idea is that humans (and current learning algorithms) are limited to "local descent" optimization methods, that make small changes in the parameter values with the effect of reducing the expected loss in average. This is clearly prone to the presence of local minima, while a more global search (in the spirit of both genetic and cultural evolution) could potentially reduce this difficulty. One hypothesis is that more abstract learning tasks involve more challenging optimization difficulties, which would make such global optimization algorithms necessary if we want computers to learn such abstractions from scratch. Another option, following the idea of curriculum learning [19], is to provide guidance ourselves to learning machines (as exemplified in the toy example of [56]), by "teaching them" gradually more complex concepts to help them understand the world around us (keeping in mind that we also have to do that for humans and that it takes 20 years to complete).

Changing the Learning Procedure and the Architecture. Regarding the basic optimization difficulty of a single deep network, three types of solutions should be considered. First, there are solutions based on improved general-purpose optimization algorithms, such as for example the recent work on adaptive learning rates [121], online natural gradient [82,102] or large-minibatch second order methods [90].

Another class of attacks on the optimization problem is based on changing the architecture (family of functions and its parametrization) or the way that the outputs are produced (for example by adding noise). As already introduced in [84], changes in the preprocessing, training objective and architecture can change the difficulty of optimization, and in particularly improve the conditioning of the Hessian matrix (of second derivatives of the loss with respect to parameters). With gradient descent, training time into a quadratic bowl is roughly proportional to the condition number of the Hessian matrix (ratio of largest to smallest eigenvalue). For example [84] recommends centering and normalizing the inputs, an idea recently extended to hidden layers of Boltzmann machines with success [95]. A related idea that may have an impact on ill-conditioning is the idea of skip-connections, which forces both the mean output and the mean slope of each hidden unit of a deep multilayer network to be zero [103], a centering idea which originates from [122].

There has also been very successful recent work exploiting rectifier non-linearities for deep supervised networks [45,77]. Interestingly, such non-linearities can produce rather sparse unit outputs, which could be exploited, if the amount of sparsity is sufficiently large, to considerably reduce the necessary computation (because when a unit output is 0, there is no need to actually multiply it with its outgoing weights). Very recently, we have discovered a variant on the rectifier non-linearity called maxout [52] which appears to open a very promising door towards more efficient training of deep networks. As confirmed experimentally [52], maxout networks can train deeper networks and allow lower layers to undergo more training. The more general principle at stake here may be that when the *gradient is sparse*, i.e., only a small subset of the hidden units and parameters is touched by the gradient, the optimization problem may become easier. We hypothesize that sparse gradient vectors have a positive effect on reducing the ill-conditioning difficulty involved in training deep nets. The intuition is that by making many terms of the gradient vector 0, one also knocks off many off-diagonal terms of the Hessian matrix, making this matrix more diagonal-looking, which would reduce many of the ill-conditioning effects involved, as explained below. Indeed, gradient descent relies on an invalid assumption: that one can modify a parameter θ_i (in the direction of the gradient $\frac{\partial C}{\partial \theta_i}$) without taking into account the changes in $\frac{\partial C}{\partial \theta_i}$ that will take place when also modifying other parameters θ_j. Indeed, this is precisely the information that is captured (e.g. with second-order methods) by the off-diagonal entries $\frac{\partial^2 C}{\partial \theta_i \partial \theta_j} = \frac{\partial}{\partial \theta_j} \frac{\partial C}{\partial \theta_i}$, i.e., how changing θ_j changes the gradient on θ_i. Whereas second-order methods may have their own limitations[13] it would be interesting if substantially reduced ill-conditioning could be achieved by modifying the architecture and training procedure. Sparse gradients would be just one weapon in this line of attack.

As we have argued above, adding noise in an appropriate way can be useful as a powerful regularizer (as in dropouts), and it can also be used to make the

[13] First, practical implementations never come close to actually inverting the Hessian, and second, they often require line searches that may be computationally inefficient if the optimal trajectory is highly curved.

gradient vector sparser, which would reinforce the above positive effect on the optimization difficulty. If some of the activations are also sparse (as our suggestions for conditional computation would require), then more entries of the gradient vector will be zeroed out, also reinforcing that beneficial optimization effect. In addition, it is plausible that the masking noise found in dropouts (as well as in denoising auto-encoders) encourages a faster *symmetry-breaking*: quickly moving away from the condition where all hidden units of a neural network or a Boltzmann machine do the same thing (due to a form of symmetry in the signals they receive), which is a non-attractive fixed point with a flat (up to several orders) likelihood function. This means that gradient descent can take a lot of time to pull apart hidden units which are behaving in a very similar way. Furthermore, when starting from small weights, these symmetry conditions (where many hidden units do something similar) are actually attractive from far away, because initially all the hidden units are trying to grab the easiest and most salient job (explain the gradients on the units at the layer above). By randomly turning off hidden units we obtain a *faster specialization* which helps training convergence.

A related concept that has been found useful in understanding and reducing the training difficulty of deep or recurrent nets is the importance of *letting the training signals (back-propagated gradients) flow, in a focused way.* It is important that error signals flow so that *credit and blame is clearly assigned* to different components of the model, those that could change slightly to improve the training loss. The problem of vanishing and exploding gradients in recurrent nets [65,8] arises because the effect of a long series of non-linear composition tends to produce gradients that can either be very small (and the error signal is lost) or very large (and the gradient steps diverge temporarily). This idea has been exploited to propose successful initialization procedures for deep nets [46]. A composition of non-linearities is associated with a product of Jacobian matrices, and a way to reduce the vanishing problem would be to make sure that they have a spectral radius (largest eigenvalue) close to 1, like what is done in the weight initialization for Echo State Networks [70] or in the carousel self-loop of LSTM [66] to help propagation of influences over longer paths. A more generic way to avoid gradient vanishing is to incorporate a training penalty that encourages the propagated gradient vectors to maintain their magnitude [101]. When combined with a gradient clipping[14] heuristic [93] to avoid the detrimental effect of overly large gradients, it allows to train recurrent nets on tasks on which it was not possible to train them before [101].

5 Inference and Sampling

All of the graphical models studied for deep learning except the humble RBM require a non-trivial form of inference, i.e., guessing values of the latent variables h that are appropriate for the given visible input x. Several forms of inference have been investigated in the past: MAP inference is formulated like an optimization problem (looking for h that approximately maximizes $P(h \mid x)$); MCMC

[14] When the norm of the gradient is above a threshold τ, reduce it to τ.

inference attempts to sample a sequence of h's from $P(h \mid x)$; variational inference looks for a simple (typically factorial) approximate posterior $q_x(h)$ that is close to $P(h \mid x)$, and usually involves an iterative optimization procedure. See a recent machine learning textbook for more details [24,4,96].

In addition, a challenge related to inference is sampling (not just from $P(h \mid x)$ but also from $P(h, x)$ or $P(x)$), which like inference is often needed in the inner loop of learning algorithms for probabilistic models with latent variables, energy-based models [85] or Markov Random Fields [74] (also known as undirected graphical models), where $P(x)$ or $P(h, x)$ is defined in terms of a parametrized energy function whose normalized exponential gives probabilities.

Deep Boltzmann machines [118] combine the challenge of inference (for the *"positive phase"* where one tries to push the energies associated with the observed x down) and the challenge of sampling (for the *"negative phase"* where one tries to push up the energies associated with x's sampled from $P(x)$). Sampling for the negative phase is usually done by MCMC, although some learning algorithms [34,57,25] involve "negative examples" that are sampled through simpler procedures (like perturbations of the observed input). In [118], inference for the positive phase is achieved with a mean-field variational approximation.[15]

5.1 Inference and Sampling: The Challenge

There are several challenges involved with all of the these inference and sampling techniques.

The first challenge is practical and computational: these are all iterative procedures that can considerably slow down training (because inference and/or sampling is often in the inner loop of learning).

Potential Huge Number of Modes. The second challenge is more fundamental and has to do with the potential existence of highly multi-modal posteriors: all of the currently known approaches to inference and sampling are making very strong explicit or implicit assumptions on the form the distribution of interest ($P(h \mid x)$ or $P(h, x)$). As we argue below, these approaches make sense if this target distribution is either approximately unimodal (MAP), (conditionally) factorizes (variational approximations, i.e., the different factors h_i are approximately independent[16] of each other given x), or has only a few modes between which it is easy to mix (MCMC). However, approximate inference can be potentially hurtful, not just at test time but for training, because it is often in the inner loop of the learning procedure [78].

[15] In the mean-field approximation, computation proceeds like in Gibbs sampling, but with stochastic binary values replaced by their conditional expected value (probability of being 1), given the outputs of the other units. This deterministic computation is iterated like in a recurrent network until convergence is approached, to obtain a marginal (factorized probability) approximation over all the units.

[16] This can be relaxed by considering tree-structured conditional dependencies [120] and mixtures thereof.

Imagine for example that h represents many explanatory variables of a rich audio-visual scene with a highly ambiguous raw input x, including the presence of several objects with ambiguous attributes or categories, such that one cannot really disambiguate one of the objects independently of the others (the so-called "structured output" scenario, but at the level of latent explanatory variables). Clearly, a factorized or unimodal representation would be inadequate (because these variables are not at all independent, given x) while the number of modes could grow exponentially with the number of ambiguous factors present in the scene. For example, consider a visual scene x through a haze hiding most details, yielding a lot of uncertainty. Say it involves 10 objects (e.g., people), each having 5 ambiguous binary attributes (out of 20) (e.g., how they are dressed) and uncertainty between 100 categorical choices for each element (e.g., out of 10000 persons in the database, the marginal evidence allows to reduce the uncertainty for each person to about 100 choices). Furthermore, suppose that these uncertainties cannot be factorized (e.g., people tend to be in the same room with other people involved in the same activity, and friends tend to stand physically close to each other, and people choose to dress in a way that socially coherent). To make life hard on mean-field and other factorized approximations, this means that only a small fraction (say 1%) of these configurations are really compatible. So one really has to consider $1\% \times (2^5 \times 100)^{10} \approx 10^{33}$ *plausible configurations* of the latent variables. If one has to take a decision y based on x, e.g., $P(y \mid x) = \sum_h P(y \mid h)P(h \mid x)$ involves summing over a huge number of non-negligible terms of the posterior $P(h \mid x)$, which we can consider as modes (the actual dimension of h is much larger, so we have reduced the problem from $(2^{20} \times 10000)^{10} \approx 10^{100}$ to about 10^{33}, but that is still huge. One way or another, *summing explicitly over that many modes seems implausible*, and assuming single mode (MAP) or a factorized distribution (mean-field) would yield very poor results. Under some assumptions on the underlying data-generating process, it might well be possible to do inference that is exact or a provably good approximations, and searching for graphical models with these properties is an interesting avenue to deal with this problem. Basically, these assumptions work because we assume a specific structure in the form of the underlying distribution. Also, if we are lucky, a few Monte-Carlo samples from $P(h \mid x)$ might suffice to obtain an acceptable approximation for our y, because somehow, as far as y is concerned, many probable values of h yield the same answer y and a Monte-Carlo sample will well represent these different "types" of values of h. That is one form of regularity that could be exploited (if it exists) to approximately solve that problem. What if these assumptions are not appropriate to solve challenging AI problems? Another, more general assumption (and thus one more likely to be appropriate for these problems) is similar to what we usually do with machine learning: although the space of functions is combinatorially large, we are able to generalize by postulating a rather large and flexible family of functions (such as a deep neural net). Thus an interesting avenue is to assume that there exists a computationally tractable function that can compute $P(y \mid x)$ in spite of the apparent complexity of going through the intermediate steps involving h, and that we may learn $P(y \mid x)$ through (x, y) examples. This idea will be developed further in Section 5.2.

Mixing between Modes. What about MCMC methods? They are hurt by the problem of mode mixing, discussed at greater length in [20], and summarized here. To make the mental picture simpler, imagine that there are only two kinds of probabilities: tiny and high. MCMC transitions try to stay in configurations that have a high probability (because they should occur in the chain much more often than the tiny probability configurations). Modes can be thought of as islands of high probability, but they may be separated by vast seas of tiny probability configurations. Hence, it is difficult for the Markov chain of MCMC methods to jump from one mode of the distribution to another, when these are separated by large low-density regions embedded in a high-dimensional space, a common situation in real-world data, and under the *manifold hypothesis* [27,98]. This hypothesis states that natural classes present in the data (e.g., visual object categories) are associated with low-dimensional regions[17] (i.e., manifolds) near which the distribution concentrates, and that different class manifolds are well-separated by regions of very low density. Here, what we consider a mode may be more than a single point, it could be a whole (low-dimensional) manifold. Slow mixing between modes means that consecutive samples tend to be correlated (belong to the same mode) and that it takes a very large number of consecutive sampling steps to go from one mode to another and even more to cover all of them, i.e., to obtain a large enough representative set of samples (e.g. to compute an expected value under the sampled variables distribution). This happens because these jumps through the low-density void between modes are unlikely and rare events. When a learner has a poor model of the data, e.g., in the initial stages of learning, the model tends to correspond to a smoother and higher-entropy (closer to uniform) distribution, putting mass in larger volumes of input space, and in particular, between the modes (or manifolds). This can be visualized in generated samples of images, that look more blurred and noisy[18]. Since MCMCs tend to make moves to nearby probable configurations, mixing between modes is therefore initially easy for such poor models. However, as the model improves and its corresponding distribution sharpens near where the data concentrate, mixing between modes becomes considerably slower. Making one unlikely move (i.e., to a low-probability configuration) may be possible, but making N such moves becomes exponentially unlikely in N. Making moves that are far *and probable* is fundamentally difficult in a high-dimensional space associated with a peaky distribution (because the exponentially large fraction of the far moves would be to an unlikely configuration), unless using additional (possibly learned) knowledge about the structure of the distribution.

[17] E.g. they can be charted with a few coordinates.

[18] See examples of generated images with some of the current state-of-the-art in learned generative models of images [37,88].

5.2 Inference and Sampling: Solution Paths

Going into a Space Where Mixing Is Easier. The idea of *tempering* [69] for MCMCs is analogous to the idea of simulated annealing [75] for optimization, and it is designed for and looks very appealing to solve the mode mixing problem: consider a smooth version (higher temperature, obtained by just dividing the energy by a temperature greater than 1) of the distribution of interest; it therefore spreads probability mass more uniformly so one can mix between modes at that high temperature version of the model, and then gradually cool to the target distribution while continuing to make MCMC moves, to make sure we end up in one of the "islands" of high probability. [42,28,117,116] have all considered various forms of tempering to address the failure of Gibbs chain mixing in RBMs. Unfortunately, convincing solutions (in the sense of making a practical impact on training efficiency) have not yet been clearly demonstrated. It is not clear why this is so, but it may be due to the need to spend much time at some specific (critical) temperatures in order to succeed. More work is certainly warranted in that direction.

An interesting observation [20] which could turn out to be helpful is that after we train a deep model such as a DBN or a stack of regularized auto-encoders, we can observe that mixing between modes is much easier at higher levels of the hierarchy (e.g. in the top-level RBM or top-level auto-encoder): mixing between modes is easier at deeper levels of representation. This is achieved by running the MCMC in a high-level representation space and then projecting back in raw input space to obtain samples at that level. The hypothesis proposed [20] to explain this observation is that unsupervised representation learning procedures (such as for the RBM and contractive or denoising auto-encoders) tend to discover a representation whose distribution has more entropy (the distribution of vectors in higher layers is more uniform) and that better "disentangles" or separates out the underlying factors of variation (see next section for a longer discussion of the concept of disentangling). For example, suppose that a perfect disentangling had been achieved that extracted the factors out of images of objects, such as object category, position, foreground color, etc. A single Gibbs step could thus switch a single top-level variable (like object category) when that variable is resampled given the others, a very local move in that top-level disentangled representation but a very far move (going to a very different place) in pixel space. Note that maximizing mutual information between inputs and their learned *deterministic* representation, which is what auto-encoders basically do [136], is equivalent to maximizing the entropy of the learned representation,[19] which supports this hypothesis. An interesting idea[20] would therefore be to use higher levels of a deep model to help the lower layers mix better, by using them in a way analogous to parallel tempering, i.e., to suggest configurations sampled from a different mode.

Another interesting potential avenue for solving the problem of sampling from a complex and rough (non-smooth) distribution would be to take advantage of

[19] Salah Rifai, personal communication.
[20] Guillaume Desjardins, personal communication.

quantum annealing effects [113] and analog computing hardware (such as produced by D-Wave). NP-hard problems (such as sampling or optimizing exactly in an Ising model) still require exponential time but experimental evidence has shown that for some problems, quantum annealing is far superior to standard digital computation [26]. Since quantum annealing is performed by essentially implementing a Boltzmann machine in analog hardware, it might be the case that drawing samples from a Boltzmann machine is one problem where quantum annealing would be dramatically superior to classical digital computing.

Learned Approximate Inference and Predicting a Rich Posterior. If we stick to the idea of obtaining actual values of the latent variables (either through MAP, factorized variational inference or MCMC), then a promising path is based on *learning approximate inference*, i.e., optimizing a learned approximate inference mechanism so that it performs a better inference faster. This idea is not new and has been shown to work well in many settings. This idea was actually already present in the wake-sleep algorithm [61,44,62] in the context of variational inference for Sigmoidal Belief Networks and DBNs. Learned approximate inference is also crucial in the predictive sparse coding (PSD) algorithm [73]. This approach is pushed further with [54] in which the parametric encoder has the same structural form as a fast iterative sparse coding approximate inference algorithm. The important consideration in both cases is not just that we have *fast* approximate inference, but that (a) it is learned, and (b) the model is learned jointly with the learned approximate inference procedure. See also [119] for learned fast approximate variational inference in DBMs, or [3,126] for learning fast approximate inference (with fewer steps than would otherwise be required by standard general purpose inference) based on loopy belief propagation.

The traditional view of probabilistic graphical models is based on the clean separation between modeling (defining the model), optimization (tuning the parameters), inference (over the latent variables) and sampling (over all the variables, and possibly over the parameters as well in the Bayesian scenario). This modularization has clear advantages but may be suboptimal. By bringing learning into inference and jointly learning the approximate inference and the "generative model" itself, one can hope to obtain "specialized" inference mechanisms that could be much more efficient and accurate than generic purpose ones; this was the subject of a recent ICML workshop [43]. The idea of learned approximate inference may help deal with the first (purely computational) challenge raised above regarding inference, i.e., it may help to speed up inference to some extent, but it generally keeps the approximate inference parameters separate from the model parameters.

But what about the challenge from a huge number of modes? What if the number of modes is too large and/or these are too well-separated for MCMC to visit efficiently or for variational/MAP inference to approximate satisfactorily? If we stick to the objective of computing actual values of the latent variables, the logical conclusion is that we should learn to approximate a posterior that is represented by a rich multi-modal distribution. To make things concrete, imagine that we learn (or identify) a function $f(x)$ of the visible variable x that computes the parameters $\theta = f(x)$ of an approximate posterior distribution $Q_{\theta=f(x)}(h)$ but where $Q_{\theta=f(x)}(h) \approx P(h \mid x)$ can be highly multi-modal, e.g., an RBM with visible variables h (coupled with *additional latent variables* used only to represent the richness of the posterior over h itself). Since the parameters of the RBM are obtained through a parametric computation taking x as input,[21] this is really a *conditional RBM* [130,129]. Whereas variational inference is usually limited to a non-parametric approximation of the posterior, $Q(h)$ (one that is analytically and iteratively optimized for each given x) one could consider a parametric approximate posterior that is learned (or derived analytically) while allowing for a rich multi-modal representation (such as what an RBM can capture, i.e., up to an exponential number of modes).

Avoiding Inference Altogether by Learning to Perform the Required Marginalization. We now propose to consider an even more *radical departure from traditional thinking regarding probabilistic models with latent variables.* It is motivated by the observation that even with the last proposal, something like a conditional RBM to capture the posterior $P(h \mid x)$, when one has to actually make a decision or a prediction, it is necessary for optimal decision-making to *marginalize over the latent variables.* For example, if we want to predict y given x, we want to compute something like $\sum_h P(y \mid h)P(h \mid x)$. If $P(h \mid x)$ is complex and highly multi-modal (with a huge number of modes), then even if we can *represent* the posterior, performing this sum exactly is out of the question, and even an MCMC approximation may be either very poor (we can only visit at most N modes with N MCMC steps, and that is very optimistic because of the mode mixing issue) or very slow (requiring an exponential number of terms being computed or a very very long MCMC chain). It seems that we have not really addressed the original "fundamental challenge with highly multi-modal posteriors" raised above.

To address this challenge, we propose to avoid explicit inference altogether by *avoiding to sample, enumerate, or represent* actual values of the latent variables h. Instead, one can just directly learn to predict $P(y \mid x)$, in the example of the previous paragraph. Hence *the only approximation error we are left with is due to to function approximation.* This might be important because the compounding of approximate inference with function approximation could be very hurtful [78].

To get there, one may wish to mentally go through an intermediate step. Imagine we had a good approximate posterior $Q_{\theta=f(x)}(h)$ as proposed above, with parameters $\theta = f(x)$. Then we could imagine learning an approximate decision model that approximates and skips the intractable sum over h, instead directly going from $\theta = f(x)$ to a prediction of y, i.e., we would estimate $P(y \mid x)$ by $g(f(x))$. Now since we are already learning $f(x)$, why learn $g(\theta)$ separately? We could simply directly learn to estimate $\pi(x) = g(f(x)) \approx P(y \mid x)$.

[21] For many models, such as deep Boltzmann machines, or bipartite discrete Markov random fields [91], f does not even need to be learned, it can be derived analytically from the form of $P(h \mid x)$.

Now that may look trivial, because this is already what we do in discriminant training of deep networks or recurrent networks, for example. And don't we lose all the advantages of probabilistic models, such as, handling different forms of uncertainty, missing inputs, and being able to answer any "question" of the form "predict any variables given any subset of the others"? Yes, if we stick to the traditional deep (or shallow) neural networks like those discussed in Section 2.1.[22] But there are other options.

We propose to get the advantages of probabilistic models without the need for explicitly going through many configurations of the latent variables. Let x_c be a subset of elements of x that are clamped, x_{-c} the rest and x_v a subset of x_{-c} for which we have a prediction to make and "target" observation. We want to be able to sample from $P(x_v \mid x_c)$. We want train a model such that, in average over the examples, with their observed subset s of variables, we maximize $\log P(x_s)$. For example, we could do this by generalized pseudo-likelihood and maximize $\log P(x_v \mid x_c)$ for randomly chosen partitions (v, c) of s. *The important requirement is that the same parameters be used to model all the predictions* $P(x_v \mid x_c)$ *for any choice of* (v, c). Another possible way to train such a model is to generalize the training criterion for regularized (possibly deep) auto-encoders in order to accomodate missing inputs (which is straightforward in the case of denoising auto-encoders with masking noise).

In the case of the generalized pseudo-likelihood approach, we would specify a computation that maps the model parameters to a training criterion equivalent to maximizing $\log P(x_v \mid x_c)$. The form of this computation could be inspired by existing or novel inference mechanisms, as has been done for learned approximate inference. However, because the training criterion would be expressed in terms of the observed x, the interpretation of the latent variables as latent variables in $P(x, h)$ becomes superfluous. In fact, because we start from an approximate inference scheme, if we train the parameters with respect to some form of input reconstruction (like generalized pseudo-likelihood), there is no guarantee that the original interpretation of the estimated posterior $P(h \mid x)$ continues to be meaningful. What is meaningful, though, is the interpretation of the parameterized computational graph that produces $P(x_v \mid x_c)$ for any (v, c) pair as a formal definition of the learned model of the data.

The approximate inference is not anymore an approximation of something else, it is the definition of the model itself. This is actually good news because we thus eliminate the issue that the approximate inference may be poor. The only thing we need to worry about is whether the parameterized computational graph that produces $P(x_v \mid x_c)$ is rich enough (or may overfit) to capture the unknown data generating distribution, and whether it makes it easy or difficult to optimize the parameters. This would be similar to *dependency networks* [58], but re-using the same parameters for every possible question-answer partition and training the system to answer for any subset of variables rather than singletons

[22] Although, using something like these deep nets would be appealing because they are currently beating benchmarks in speech recognition, language modeling and object recognition.

like in pseudo-likelihood. For the same reason, it raises the question of whether the different estimated conditionals are coherent with a global joint distribution. In the case where the computational graph is obtained from the template of an inference mechanism for a joint distribution (such as variational inference), then clearly, we keep the property that these conditionals are coherent with a global joint distribution. With the mean-field variational inference, the computational graph looks like a recurrent neural network converging to a fixed point, and where we stop the iterations after a fixed number of steps or according to a convergence criterion. Such a trained parametrized computational graph is used in the iterative variational approach introduced in [51] for training and missing value inference in deep Boltzmann machines, with an inpainting-like criterion in which arbitrary subsets of pixels are predicted given the others (a generalized pseudo-likelihood criterion). It has also been used in a recursion that follows the template of loopy belief propagation to fill-in the missing inputs and produce outputs [126]. Although in these cases there are latent variables (e.g. the latent variables of the deep Boltzmann machine) that motivate the "template" used for the learned approximate inference, what we propose here is to stop thinking about them as actual latent factors, but rather just as a way to parametrize this template for a question answering mechanism regarding missing inputs, i.e., the "generic conditional prediction mechanism" implemented by the recurrent computational graph that is trained to predict any subset of variables given any other subset. Although [51] assume a factorial distribution across the predicted variables, we propose to investigate non-factorial posterior distributions over the observed variables, i.e., in the spirit of the recent flurry of work on *structured output* machine learning [133].

We can think of this parametrized computational graph as a family of functions, each corresponding to answering a different question (predict a specific set of variables given some others), but all sharing the same parameters. We already have examples of such families in machine learning, e.g., with recurrent neural networks or dynamic Bayes nets (where the functions in the family are indexed by the length of the sequence). This is also analogous to what happens with dropouts, where we have an exponential number of neural networks corresponding to different sub-graphs from input to output (indexed by which hidden units are turned on or off). For the same reason as in these examples, we obtain a form of generalization across subsets. Following the idea of learned approximate inference, the parameters of the question-answering inference mechanism would be taking advantage of the specific underlying structure in the data generating distribution. Instead of trying to do inference on the anonymous latent variables, it would be trained to do good inference only over observed variables or over high-level features learned by a deep architecture, obtained deterministically from the observed input.

The idea that we should train with the approximate inference as part of the computational graph for producing a decision (and a loss) was first introduced by [126], and we simply push it further here, by proposing to allow the computational graph to depart in any way we care to explore from the template provided by existing inference mechanism, i.e., potentially losing the connection and the reference to probabilistic latent variables. Once we free ourselves from the constraint of interpreting this parametrized question answering computational graph as corresponding to approximate inference involving latent variables, all kinds of architectures and parametrizations are possible, where current approximate inference mechanisms can serve as inspiration and starting points. It is quite possible that this new freedom could give rise to much better models. The important point is that this mechanism is *trained* to do well at question answering on the provided data, and that it is really a family of functions indexed by all the possible question/answer subsets, but sharing their parameters. If the resulting family of computational graphs does not correspond to a form of inference in a globally coherent joint distribution (like in the case of dependency networks [58]), what do we lose? Re-using the arguments from [126], we could argue that what matters is to optimize the expected loss. If our loss function regards the ability to answer any question about the variables, then a generalized pseudo-likelihood criterion would better much the ultimate objective.

To go farther than [51] and [126] it would be good to go beyond the kind of factorized prediction common in variational and loopy belief propagation inference. One idea is to represent the estimated joint distribution of the predicted variables (given the clamped variables) by a powerful model such as an RBM or a regularized auto-encoder, e.g., as has been done for structured output predictions when there is complex probabilistic structure between the output variables [94,87].

Although conditional RBMs have been already explored, conditional distributions provided by regularized auto-encoders remain to be studied. Alternatively, a denoising auto-encoder (whether it is shallow or deep) with masking noise[23] is trained to perform something very similar to generalized pseudo-likelihood. Note that sampling algorithms based on Langevin or Metropolis-Hastings MCMC have already been proposed [110,1,15], for regularized auto-encoders[24] and they could easily be adapted to conditional sampling by clamping the fixed inputs and (optionally, to increase representational capacity) by making the hidden unit biases an arbitrarily complex (but deterministic) functions of the observed inputs. These theoretical analyses and sampling methods for regularized auto-encoders have been performed for the case of continuous inputs with squared error, and remain to be generalized to discrete inputs.

As as refinement, and in the spirit of a long tradition of discriminatively oriented machine learning, when some of the observed variables y are of particular interest (because we often want to predict them), one would naturally present examples of the prediction of y given x more often to the learning algorithm than random subsets of observed variables. Hybrids of generative and discriminant training criteria have been very successful for RBMs [79,80] and would make practical sense here as well.

[23] In which some of the inputs are set to 0 and the auto-encoder is trying to predict them, as well as the rest, in its reconstruction.

[24] These methods iterate between encoding, decoding, and injecting noise, with the possibility of rejecting poor configurations.

All these ideas lead to the question: what is the interpretation of hidden layers, if not directly of the underlying generative latent factors? The answer may simply be that they provide a better representation of these factors, a subject discussed in the next section. But what about the representation of uncertainty about these factors? The author believes that humans and other animals carry in their head an internal representation that *implicitly* captures both the most likely interpretation of any of these factors (in case a hard decision about some of them has to be taken) and uncertainty about their joint assignment. This is of course a speculation. Somehow, our brain would be *operating on implicit representations of the joint distribution between these explanatory factors*, generally without having to commit until a decision is required or somehow provoked by our attention mechanisms (which seem related to our tendancy to verbalize a discrete interpretation). A good example is foreign language understanding for a person who does not master that foreign language. Until we consciously think about it, we generally don't commit to a particular meaning for ambiguous word (which would be required by MAP inference), or even to the segmentation of the speech in words, but we can take a hard decision that depends on the interpretation of these words if we have to, without having to go through this intermediate step of discrete interpretation, instead treating the ambiguous information as soft cues that may inform our decision. In that example, a factorized posterior is also inadequate because some word interpretations are more compatible with each other.

To summarize, what we propose here, unlike in previous work on approximate inference, is to drop the pretense that the learned approximate inference mechanism actually approximates the latent variables distribution, mode, or expected value. Instead, we only consider the approximate inference over observed variables (or of values of features computed from the observed variables at a higher level of a deep architecture) and we consider that this mechanism is itself the model, rather than some approximation, and we train it with a training criterion that is consistent with that interpretation. By removing the interpretation of approximately marginalizing over latent variables, we free ourselves from a strong constraint and open the door to any parametrized computation which has the requirement that its parameters can be shared across any question/answer subset.

This discussion is of course orthogonal to the use of Bayesian averaging methods in order to produce better-generalizing predictions, i.e., handling uncertainty due to a small number of training examples. The proposed methods can be made Bayesian just like neural networks have their Bayesian variants [99], by somehow maintaining an implicit or explicit distribution over parameters. A promising step in this direction was proposed by [137], making such Bayesian computation tractable by exploiting the randomness introduced with stochastic gradient descent to also produce the Bayesian samples over the uncertain parameter values.

6 Disentangling

6.1 Disentangling: The Challenge

What are “underlying factors” explaining the data? The answer is not obvious. One answer could be that these are factors that can be *separately controlled* (one could set up way to change one but not the others). This can actually be observed by looking at sequential real-world data, where only a small proportion of the factors typically change from t to $t+1$. Complex data arise from the rich interaction of many sources. These factors interact in a complex web that can complicate AI-related tasks such as object classification. If we could identity and separate out these factors (i.e., disentangle them), we would have almost solved the learning problem. For example, an image is composed of the interaction between one or more light sources, the object shapes and the material properties of the various surfaces present in the image. It is important to distinguish between the related but distinct goals of learning invariant features and learning to disentangle explanatory factors. The central difference is the preservation of information. Invariant features, by definition, have reduced sensitivity in the directions of invariance. This is the goal of building features that are insensitive to variation in the data that are uninformative to the task at hand. Unfortunately, it is often difficult to determine *a priori* which set of features and variations will ultimately be relevant to the task at hand. Further, as is often the case in the context of deep learning methods, the feature set being trained may be destined to be used in multiple tasks that may have distinct subsets of relevant features. Considerations such as these lead us to the conclusion that the most robust approach to feature learning is *to disentangle as many factors as possible, discarding as little information about the data as is practical.*

Deep learning algorithms that can do a much better job of disentangling the underlying factors of variation would have tremendous impact. For example, suppose that the underlying factors can be “guessed” (predicted) from a simple (e.g. linear) transformation of the learned representation, ideally a transformation that only depends on a few elements of the representation. That is what we mean by a representation that disentangles the underlying factors. It would clearly make learning a new supervised task (which may be related to one or a few of them) much easier, because the supervised learning could quickly learn those linear factors, zooming in on the parts of the representation that are relevant.

Of all the challenges discussed in this paper, this is probably the most ambitious, and success in solving it the most likely to have far-reaching impact. In addition to the obvious observation that disentangling the underlying factors is almost like pre-solving any possible task relevant to the observed data, having disentangled representations would also solve other issues, such as the issue of mixing between modes. We believe that it would also considerably reduce the optimization problems involved when new information arrives and has to be reconciled with the world model implicit in the current parameter setting. Indeed, it would allow only changing the parts of the model that involve the factors that are relevant to the new observation, in the spirit of *sparse updates* and *reduced ill-conditioning* discussed above.

6.2 Disentangling: Solution Paths

Deeper Representations Disentangle Better. There are some encouraging signs that our current unsupervised representation-learning algorithms are reducing the "entanglement" of the underlying factors[25] when we apply them to raw data (or to the output of a previous representation learning procedure, like when we stack RBMs or regularized auto-encoders).

First, there are experimental observations suggesting that sparse convolutional RBMs and sparse denoising auto-encoders achieve in their hidden units a greater degree of disentangling than in their inputs [50,47]. What these authors found is that some hidden units were particularly sensitive to a known factor of variation while being rather insensitive (i.e., invariant) to others. For example, in a sentiment analysis model that sees unlabeled paragraphs of customer comments from the Amazon web site, some hidden units specialized on the topic of the paragraph (the type of product being evaluated, e.g., book, video, music) while other units specialized on the sentiment (positive vs negative). The disentanglement was never perfect, so the authors made quantitative measurements of sensitivity and invariance and compared these quantities on the input and the output (learned representation) of the unsupervised learners.

Another encouraging observation (already mentioned in the section on mixing) is that deeper representations were empirically found to be more amenable to quickly mixing between modes [20]. Two (compatible) hypotheses were proposed to explain this observation: (1) RBMs and regularized auto-encoders deterministically transform[26] their input distribution into one that is more uniform-looking, that better fills the space (thus creating easier paths between modes), and (2) these algorithms tend to discover representations that are more disentangled. The advantage of a higher-level disentangled representation is that a small MCMC step (e.g. Gibbs) in that space (e.g. flipping one high-level variable) can move in one step from one input-level mode to a distant one, e.g., going from one shape / object to another one, adding or removing glasses on the face of a person (which requires a very sharp coordination of pixels far from each other because glasses occupy a very thin image area), or replacing foreground and background colors (such as going into a "reverse video" mode).

Although these observations are encouraging, we do not yet have a clear understanding as to *why* some representation algorithms tend to move towards more disentangled representations, and there are other experimental observations suggesting that this is *far from sufficient.* In particular, [56] show an example of a task on which deep supervised nets (and every other black-box machine learning algorithm tried) fail, on which a completely disentangled input representation makes the task feasible (with a maxout network [52]). Unfortunately, unsupervised pre-training applied on the raw input images failed to produce enough

[25] As measured by how predictive some individual features are of known factors.
[26] When considering the features learned, e.g., the $P(h_i = 1 \mid x)$, for RBMs.

disentangling to solve the task, even with the appropriate convolutional structure. What is interesting is that we now have a simple artificial task on which we can evaluate new unsupervised representation learning methods for their disentangling ability. It may be that a variant of the current algorithms will eventually succeed at this task, or it may be that altogether different unsupervised representation learning algorithms are needed.

Generic Priors for Disentangling Factors of Variation. A general strategy was outlined in [17] to enhance the discovery of representations which disentangle the underlying and unknown factors of variation: it relies on exploiting *priors* about these factors. We are most interested in *broad generic priors* that can be useful for a large class of learning problems of interest in AI. We list these priors here:

• **Smoothness:** assumes the function f to be learned is s.t. $x \approx y$ generally implies $f(x) \approx f(y)$. This most basic prior is present in most machine learning, but is insufficient to get around the curse of dimensionality.

• **Multiple Explanatory Factors:** the data generating distribution is generated by different underlying factors, and for the most part what one learns about one factor generalizes in many configurations of the other factors. The objective is to recover or at least disentangle these underlying factors of variation. This assumption is behind the idea of **distributed representations**. More specific priors on the form of the model can be used to enhance disentangling, such as multiplicative interactions between the factors [131,41] or orthogonality of the features derivative with respect to the input [111,109,125]. The parametrization and training procedure may also be used to disentangle discrete factors (e.g., detecting a shape) from associated continuous-valued factors (e.g., pose parameters), as in transforming auto-encoders [60], spike-and-slab RBMs with pooled slab variables [37] and other pooling-based models that learn a feature subspace [76,68].

• **A Hierarchical Organization of Explanatory Factors:** the concepts that are useful for describing the world around us can be defined in terms of other concepts, in a hierarchy, with more **abstract** concepts higher in the hierarchy, defined in terms of less abstract ones. This assumption is exploited with deep representations. Although stacking single-layer models has been rather successful, much remains to be done regarding the joint training of all the layers of a deep unsupervised model.

• **Semi-supervised Learning:** with inputs X and target Y to predict, given X, a subset of the factors explaining X's distribution explain much of Y, given X. Hence representations that are useful for spelling out $P(X)$ tend to be useful when learning $P(Y \mid X)$, allowing sharing of statistical strength between the unsupervised and supervised learning tasks. However, many of the factors that explain X may dominate those that also explain Y, which can make it useful to incorporate observations of Y in training the learned representations, i.e., by semi-supervised representation learning.

• **Shared Factors across Tasks:** with many Y's of interest or many learning tasks in general, tasks (e.g., the corresponding $P(Y \mid X, \text{task})$) are explained by factors that are shared with other tasks, allowing sharing of statistical strength across tasks, e.g. for multi-task and transfer learning or domain adaptation. This can be achieved by sharing embeddings or representation functions across tasks [34,25].

• **Manifolds:** probability mass concentrates near regions that have a much smaller dimensionality than the original space where the data lives. This is exploited with regularized auto-encoder algorithms, but training criteria that would explicitly take into account that we are looking for a concentration of mass in an integral number directions remain to be developed.

• **Natural Clustering:** different values of categorical variables such as object classes are associated with separate manifolds. More precisely, the local variations on the manifold tend to preserve the value of a category, and a linear interpolation between examples of different classes in general involves going through a low density region, i.e., $P(X \mid Y = i)$ for different i tend to be well separated and not overlap much. For example, this is exploited in the Manifold Tangent Classifier [111]. This hypothesis is consistent with the idea that humans have *named* categories and classes because of such statistical structure (discovered by their brain and propagated by their culture), and machine learning tasks often involves predicting such categorical variables.

• **Temporal and Spatial Coherence:** this prior introduced in [5] is similar to the natural clustering assumption but concerns sequences of observations: consecutive (from a sequence) or spatially nearby observations tend to be *easily predictable from each other*. In the special case typically studied, e.g., slow feature analysis [139], one assumes that consecutive values are close to each other, or that categorical concepts remain either present or absent for most of the transitions. More generally, different underlying factors change at different temporal and spatial scales, and this could be exploited to sift different factors into different categories based on their temporal scale.

• **Sparsity:** for any given observation x, only a small fraction of the possible factors are relevant. In terms of representation, this could be represented by features that are often zero (as initially proposed by [100]), or more generally by the fact that most of the extracted features are *insensitive* to small variations of x. This can be achieved with certain forms of priors on latent variables (peaked at 0), or by using a non-linearity whose value is often flat at 0 (i.e., 0 and with a 0 derivative), or simply by penalizing the magnitude of the derivatives of the function mapping input to representation. A variant on that hypothesis is that for any given input, only a small part of the model is relevant and only a small subset of the parameters need to be updated.

• **Simplicity of Factor Dependencies**: in good high-level representations, the factors are related to each other through simple, typically linear, dependencies. This can be seen in many laws of physics, and is assumed when plugging a linear predictor on top of a learned representation.

7 Conclusion

Deep learning and more generally representation learning are recent areas of investigation in machine learning and recent years of research have allowed to clearly identify several major challenges for approaching the performance of these algorithms from that of humans. We have broken down these challenges into four major areas: scaling computations, reducing the difficulties in optimizing parameters, designing (or avoiding) expensive inference and sampling, and helping to learn representations that better disentangle the unknown underlying factors of variation. There is room for exploring many paths towards addressing all of these issues, and we have presented here a few appealing directions of research towards these challenges.

Acknowledgments. The author is extremely grateful for the feedback and discussions he enjoyed with collaborators Ian Goodfellow, Guillaume Desjardins, Aaron Courville, Pascal Vincent, Roland Memisevic and Nicolas Chapados, which greatly contributed to help form the ideas presented here and fine-tune this manuscript. He is also grateful for the funding support from NSERC, CIFAR, the Canada Research Chairs, and Compute Canada.

References

1. Alain, G., Bengio, Y.: What regularized auto-encoders learn from the data generating distribution. Tech. Rep. Arxiv report 1211.4246, Université de Montréal (2012)
2. Bach, F., Jenatton, R., Mairal, J., Obozinski, G.: Structured sparsity through convex optimization. Tech. rep., arXiv.1109.2397 (2011)
3. Bagnell, J.A., Bradley, D.M.: Differentiable sparse coding. In: NIPS 2009, pp. 113–120 (2009)
4. Barber, D.: Bayesian Reasoning and Machine Learning. Cambridge University Press (2011)
5. Becker, S., Hinton, G.: A self-organizing neural network that discovers surfaces in random-dot stereograms. Nature 355, 161–163 (1992)
6. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A neural probabilistic language model. JMLR 3, 1137–1155 (2003)
7. Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H.: Greedy layer-wise training of deep networks. In: NIPS 2006 (2007)
8. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks 5(2), 157–166 (1994), http://www.iro.umontreal.ca/~lisa/pointeurs/ieeetrnn94.pdf

9. Bengio, Y.: Neural net language models. Scholarpedia 3(1) (2008)
10. Bengio, Y.: Learning deep architectures for AI. Now Publishers (2009)
11. Bengio, Y.: Deep learning of representations for unsupervised and transfer learning. In: JMLR W&CP: Proc. Unsupervised and Transfer Learning (2011)
12. Bengio, Y.: Estimating or propagating gradients through stochastic neurons. Tech. Rep. arXiv, Universite de Montreal (to appear, 2013)
13. Bengio, Y.: Evolving culture vs local minima. In: Kowaliw, T., Bredeche, N., Doursat, R. (eds.) Growing Adaptive Machines: Integrating Development and Learning in Artificial Neural Networks, No. also as ArXiv 1203.2990v1. Springer (2013), `http://arxiv.org/abs/1203.2990`
14. Bengio, Y.: Practical recommendations for gradient-based training of deep architectures. In: Montavon, G., Orr, G.B., Müller, K.-R. (eds.) Neural Networks: Tricks of the Trade, 2nd edn. LNCS, vol. 7700, pp. 437–478. Springer, Heidelberg (2012)
15. Bengio, Y., Alain, G., Rifai, S.: Implicit density estimation by local moment matching to sample from auto-encoders. Tech. rep., arXiv:1207.0057 (2012)
16. Bengio, Y., Boulanger-Lewandowski, N., Pascanu, R.: Advances in optimizing recurrent networks. In: ICASSP 2013 (2013)
17. Bengio, Y., Courville, A., Vincent, P.: Unsupervised feature learning and deep learning: A review and new perspectives. IEEE Trans. Pattern Analysis and Machine Intelligence, PAMI (2013)
18. Bengio, Y., Delalleau, O., Simard, C.: Decision trees do not generalize to new variations. Computational Intelligence 26(4), 449–467 (2010)
19. Bengio, Y., Louradour, J., Collobert, R., Weston, J.: Curriculum learning. In: ICML 2009 (2009)
20. Bengio, Y., Mesnil, G., Dauphin, Y., Rifai, S.: Better mixing via deep representations. In: ICML 2013 (2013)
21. Bergstra, J., Bastien, F., Breuleux, O., Lamblin, P., Pascanu, R., Delalleau, O., Desjardins, G., Warde-Farley, D., Goodfellow, I., Bergeron, A., Bengio, Y.: Theano: Deep learning on gpus with python. In: Big Learn Workshop, NIPS (2011)
22. Bergstra, J., Bengio, Y.: Slow, decorrelated features for pretraining complex cell-like networks. In: NIPS 2009 (December 2009)
23. Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., Bengio, Y.: Theano: a CPU and GPU math expression compiler. In: Proceedings of the Python for Scientific Computing Conference, SciPy (2010)
24. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer (2006)
25. Bordes, A., Glorot, X., Weston, J., Bengio, Y.: A semantic matching energy function for learning with multi-relational data. Machine Learning: Special Issue on Learning Semantics (2013)
26. Brooke, J.J., Bitko, D., Rosenbaum, T.F., Aeppli, G.: Quantum annealing of a disordered magnet. Tech. Rep. cond-mat/0105238 (May 2001)
27. Cayton, L.: Algorithms for manifold learning. Tech. Rep. CS2008-0923, UCSD (2005)
28. Cho, K., Raiko, T., Ilin, A.: Parallel tempering is efficient for learning restricted Boltzmann machines. In: IJCNN 2010 (2010)
29. Ciresan, D., Meier, U., Schmidhuber, J.: Multi-column deep neural networks for image classification. Tech. rep., arXiv:1202.2745 (2012)

30. Coates, A., Lee, H., Ng, A.Y.: An analysis of single-layer networks in unsupervised feature learning. In: AISTATS 2011 (2011)
31. Coates, A., Ng, A.Y.: The importance of encoding versus training with sparse coding and vector quantization. In: ICML 2011 (2011)
32. Coates, A., Karpathy, A., Ng, A.: Emergence of object-selective features in unsupervised feature learning. In: NIPS 2012 (2012)
33. Collobert, R., Bengio, Y., Bengio, S.: Scaling large learning problems with hard parallel mixtures. International Journal of Pattern Recognition and Artificial Intelligence 17(3), 349–365 (2003)
34. Collobert, R., Weston, J.: A unified architecture for natural language processing: Deep neural networks with multitask learning. In: ICML 2008 (2008)
35. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. Journal of Machine Learning Research 12, 2493–2537 (2011)
36. Corrado, G.: Deep networks for predicting ad click through rates. In: ICML 2012 Online Advertising Workshop (2012)
37. Courville, A., Bergstra, J., Bengio, Y.: Unsupervised models of images by spike-and-slab RBMs. In: ICML 2011 (2011)
38. Dauphin, Y., Bengio, Y.: Big neural networks waste capacity. Tech. Rep. arXiv:1301.3583, Universite de Montreal (2013)
39. Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Le, Q., Mao, M., Ranzato, M., Senior, A., Tucker, P., Yang, K., Ng, A.Y.: Large scale distributed deep networks. In: NIPS 2012 (2012)
40. Deng, L., Li, J., Huang, J.T., Yao, K., Yu, D., Seide, F., Seltzer, M., Zweig, G., He, X., Williams, J., Gong, Y., Acero, A.: Recent advances in deep learning for speech research at Microsoft. In: ICASSP 2013 (2013)
41. Desjardins, G., Courville, A., Bengio, Y.: Disentangling factors of variation via generative entangling (2012)
42. Desjardins, G., Courville, A., Bengio, Y., Vincent, P., Delalleau, O.: Tempered Markov chain Monte Carlo for training of restricted Boltzmann machine. In: AISTATS, vol. 9, pp. 145–152 (2010)
43. Eisner, J.: Learning approximate inference policies for fast prediction. Keynote Talk at ICML Workshop on Inferning: Interactions Between Search and Learning (June 2012)
44. Frey, B.J., Hinton, G.E., Dayan, P.: Does the wake-sleep algorithm learn good density estimators? In: NIPS 1995, pp. 661–670. MIT Press, Cambridge (1996)
45. Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier neural networks. In: AISTATS (2011)
46. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: AISTATS 2010 (2010)
47. Glorot, X., Bordes, A., Bengio, Y.: Domain adaptation for large-scale sentiment classification: A deep learning approach. In: ICML 2011 (2011)
48. Goodfellow, I., Courville, A., Bengio, Y.: Spike-and-slab sparse coding for unsupervised feature discovery. In: NIPS Workshop on Challenges in Learning Hierarchical Models (2011)
49. Goodfellow, I., Courville, A., Bengio, Y.: Large-scale feature learning with spike-and-slab sparse coding. In: ICML 2012(2012)
50. Goodfellow, I., Le, Q., Saxe, A., Ng, A.: Measuring invariances in deep networks. In: NIPS 2009, pp. 646–654 (2009)

51. Goodfellow, I.J., Courville, A., Bengio, Y.: Joint training of deep Boltzmann machines for classification. Tech. rep., arXiv:1301.3568 (2013)
52. Goodfellow, I.J., Warde-Farley, D., Mirza, M., Courville, A., Bengio, Y.: Maxout networks. In: ICML 2013 (2013)
53. Gregor, K., LeCun, Y.: Learning fast approximations of sparse coding. In: Bottou, L., Littman, M. (eds.) Proceedings of the Twenty-seventh International Conference on Machine Learning (ICML 2010). ACM (2010)
54. Gregor, K., LeCun, Y.: Learning fast approximations of sparse coding. In: ICML 2010 (2010)
55. Grosse, R., Raina, R., Kwong, H., Ng, A.Y.: Shift-invariant sparse coding for audio classification. In: UAI 2007 (2007)
56. Gulcehre, C., Bengio, Y.: Knowledge matters: Importance of prior information for optimization. Tech. Rep. arXiv:1301.4083, Universite de Montreal (2013)
57. Gutmann, M., Hyvarinen, A.: Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In: AISTATS 2010 (2010)
58. Heckerman, D., Chickering, D.M., Meek, C., Rounthwaite, R., Kadie, C.: Dependency networks for inference, collaborative filtering, and data visualization. Journal of Machine Learning Research 1, 49–75 (2000)
59. Hinton, G., Deng, L., Dahl, G.E., Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T., Kingsbury, B.: Deep neural networks for acoustic modeling in speech recognition. IEEE Signal Processing Magazine 29(6), 82–97 (2012)
60. Hinton, G., Krizhevsky, A., Wang, S.: Transforming auto-encoders. In: ICANN 2011 (2011)
61. Hinton, G.E., Dayan, P., Frey, B.J., Neal, R.M.: The wake-sleep algorithm for unsupervised neural networks. Science 268, 1158–1161 (1995)
62. Hinton, G.E., Osindero, S., Teh, Y.: A fast learning algorithm for deep belief nets. Neural Computation 18, 1527–1554 (2006)
63. Hinton, G.E., Salakhutdinov, R.: Reducing the dimensionality of data with neural networks. Science 313(5786), 504–507 (2006)
64. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Improving neural networks by preventing co-adaptation of feature detectors. Tech. rep., arXiv:1207.0580 (2012)
65. Hochreiter, S.: Untersuchungen zu dynamischen neuronalen Netzen. Diploma thesis, Institut für Informatik, Lehrstuhl Prof. Brauer, Technische Universität München (1991), `http://www7.informatik.tu-muenchen.de/~Ehochreit`
66. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation 9(8), 1735–1780 (1997)
67. Hyvärinen, A.: Estimation of non-normalized statistical models using score matching. J. Machine Learning Res. 6 (2005)
68. Hyvärinen, A., Hoyer, P.: Emergence of phase and shift invariant features by decomposition of natural images into independent feature subspaces. Neural Computation 12(7), 1705–1720 (2000)
69. Iba, Y.: Extended ensemble monte carlo. International Journal of Modern Physics C12, 623–656 (2001)
70. Jaeger, H.: Echo state network. Scholarpedia 2(9), 2330 (2007)
71. Jarrett, K., Kavukcuoglu, K., Ranzato, M., LeCun, Y.: What is the best multi-stage architecture for object recognition? In: ICCV 2009 (2009)

72. Jenatton, R., Audibert, J.Y., Bach, F.: Structured variable selection with sparsity-inducing norms. Tech. rep., arXiv:0904.3523 (2009)
73. Kavukcuoglu, K., Ranzato, M., LeCun, Y.: Fast inference in sparse coding algorithms with applications to object recognition. CBLL-TR-2008-12-01, NYU (2008)
74. Kindermann, R.: Markov Random Fields and Their Applications (Contemporary Mathematics; V. 1). American Mathematical Society (1980)
75. Kirkpatrick, S., Gelatt Jr., C.D., Vecchi, M.P.: Optimization by simulated annealing. Science 220, 671–680 (1983)
76. Kohonen, T.: Emergence of invariant-feature detectors in the adaptive-subspace self-organizing map. Biological Cybernetics 75, 281–291 (1996), `http://dx.doi.org/10.1007/s004220050295`, doi:10.1007/s004220050295
77. Krizhevsky, A., Sutskever, I., Hinton, G.: ImageNet classification with deep convolutional neural networks. In: NIPS 2012 (2012)
78. Kulesza, A., Pereira, F.: Structured learning with approximate inference. In: NIPS 2007 (2008)
79. Larochelle, H., Bengio, Y.: Classification using discriminative restricted Boltzmann machines. In: ICML 2008 (2008)
80. Larochelle, H., Mandel, M., Pascanu, R., Bengio, Y.: Learning algorithms for the classification restricted Boltzmann machine. JMLR 13, 643–669 (2012)
81. Le, Q., Ranzato, M., Monga, R., Devin, M., Corrado, G., Chen, K., Dean, J., Ng, A.: Building high-level features using large scale unsupervised learning. In: ICML 2012 (2012)
82. Le Roux, N., Manzagol, P.A., Bengio, Y.: Topmoumoute online natural gradient algorithm. In: NIPS 2007 (2008)
83. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient based learning applied to document recognition. Proc. IEEE (1998)
84. LeCun, Y., Bottou, L., Orr, G.B., Müller, K.: Efficient backprop. In: Neural Networks, Tricks of the Trade (1998)
85. LeCun, Y., Chopra, S., Hadsell, R., Ranzato, M.A., Huang, F.J.: A tutorial on energy-based learning. In: Bakir, G., Hofman, T., Scholkopf, B., Smola, A., Taskar, B. (eds.) Predicting Structured Data, pp. 191–246. MIT Press (2006)
86. Lee, H., Ekanadham, C., Ng, A.: Sparse deep belief net model for visual area V2. In: NIPS 2007 (2008)
87. Li, Y., Tarlow, D., Zemel, R.: Exploring compositional high order pattern potentials for structured output learning. In: CVPR 2013 (2013)
88. Luo, H., Carrier, P.L., Courville, A., Bengio, Y.: Texture modeling with convolutional spike-and-slab RBMs and deep extensions. In: AISTATS 2013 (2013)
89. Mairal, J., Bach, F., Ponce, J., Sapiro, G.: Online dictionary learning for sparse coding. In: ICML 2009 (2009)
90. Martens, J.: Deep learning via Hessian-free optimization. In: Bottou, L., Littman, M. (eds.) Proceedings of the Twenty-seventh International Conference on Machine Learning (ICML 2010), pp. 735–742. ACM (June 2010)
91. Martens, J., Sutskever, I.: Parallelizable sampling of Markov random fields. In: AISTATS 2010 (2010)
92. Mesnil, G., Dauphin, Y., Glorot, X., Rifai, S., Bengio, Y., Goodfellow, I., Lavoie, E., Muller, X., Desjardins, G., Warde-Farley, D., Vincent, P., Courville, A., Bergstra, J.: Unsupervised and transfer learning challenge: a deep learning approach. In: JMLR W&CP: Proc. Unsupervised and Transfer Learning, vol. 7 (2011)

93. Mikolov, T.: Statistical Language Models based on Neural Networks. Ph.D. thesis, Brno University of Technology (2012)
94. Mnih, V., Larochelle, H., Hinton, G.: Conditional restricted Boltzmann machines for structure output prediction. In: Proc. Conf. on Uncertainty in Artificial Intelligence, UAI (2011)
95. Montavon, G., Müller, K.-R.: Deep Boltzmann machines and the centering trick. In: Montavon, G., Orr, G.B., Müller, K.-R. (eds.) Neural Networks: Tricks of the Trade, 2nd edn. LNCS, vol. 7700, pp. 621–637. Springer, Heidelberg (2012)
96. Murphy, K.P.: Machine Learning: a Probabilistic Perspective. MIT Press, Cambridge (2012)
97. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: ICML 2010 (2010)
98. Narayanan, H., Mitter, S.: Sample complexity of testing the manifold hypothesis. In: NIPS 2010 (2010)
99. Neal, R.M.: Bayesian Learning for Neural Networks. Ph.D. thesis, Dept. of Computer Science, University of Toronto (1994)
100. Olshausen, B.A., Field, D.J.: Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Nature 381, 607–609 (1996)
101. Pascanu, R., Bengio, Y.: On the difficulty of training recurrent neural networks. Tech. Rep. arXiv:1211.5063, Universite de Montreal (2012)
102. Pascanu, R., Bengio, Y.: Revisiting natural gradient for deep networks. Tech. rep., arXiv:1301.3584 (2013)
103. Raiko, T., Valpola, H., LeCun, Y.: Deep learning made easier by linear transformations in perceptrons. In: AISTATS 2012 (2012)
104. Raina, R., Battle, A., Lee, H., Packer, B., Ng, A.Y.: Self-taught learning: transfer learning from unlabeled data. In: ICML 2007 (2007)
105. Raina, R., Madhavan, A., Ng, A.Y.: Large-scale deep unsupervised learning using graphics processors. In: Bottou, L., Littman, M. (eds.) ICML 2009, pp. 873–880. ACM, New York (2009)
106. Ranzato, M., Poultney, C., Chopra, S., LeCun, Y.: Efficient learning of sparse representations with an energy-based model. In: NIPS 2006 (2007)
107. Ranzato, M., Boureau, Y.L., LeCun, Y.: Sparse feature learning for deep belief networks. In: NIPS 2007, pp. 1185–1192. MIT Press, Cambridge (2008)
108. Recht, B., Re, C., Wright, S., Niu, F.: Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In: NIPS 2011 (2011)
109. Rifai, S., Bengio, Y., Courville, A., Vincent, P., Mirza, M.: Disentangling factors of variation for facial expression recognition. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part VI. LNCS, vol. 7577, pp. 808–822. Springer, Heidelberg (2012)
110. Rifai, S., Bengio, Y., Dauphin, Y., Vincent, P.: A generative process for sampling contractive auto-encoders. In: ICML 2012 (2012)
111. Rifai, S., Dauphin, Y., Vincent, P., Bengio, Y., Muller, X.: The manifold tangent classifier. In: NIPS 2011 (2011)
112. Rifai, S., Vincent, P., Muller, X., Glorot, X., Bengio, Y.: Contractive auto-encoders: Explicit invariance during feature extraction. In: ICML 2011 (2011)
113. Rose, G., Macready, W.: An introduction to quantum annelaing. Tech. rep., D-Wave Systems (2007)
114. Rumelhart, D., Hinton, G., Williams, R.: Learning representations by back-propagating errors. Nature 323, 533–536 (1986)

115. Salakhutdinov, R., Mnih, A., Hinton, G.: Restricted Boltzmann machines for collaborative filtering. In: ICML 2007. pp. 791–798 (2007)
116. Salakhutdinov, R.: Learning deep Boltzmann machines using adaptive MCMC. In: ICML 2010 (2010)
117. Salakhutdinov, R.: Learning in Markov random fields using tempered transitions. In: NIPS 2010 (2010)
118. Salakhutdinov, R., Hinton, G.: Deep Boltzmann machines. In: AISTATS 2009, pp. 448–455 (2009)
119. Salakhutdinov, R., Larochelle, H.: Efficient learning of deep Boltzmann machines. In: AISTATS 2010 (2010)
120. Saul, L.K., Jordan, M.I.: Exploiting tractable substructures in intractable networks. In: NIPS 1995. MIT Press, Cambridge (1996)
121. Schaul, T., Zhang, S., LeCun, Y.: No More Pesky Learning Rates. Tech. rep., New York University, arxiv 1206.1106 (June 2012), http://arxiv.org/abs/1206.1106
122. Schraudolph, N.N.: Centering neural network gradient factors. In: Orr, G.B., Müller, K.-R. (eds.) Neural Networks: Tricks of the Trade. LNCS, vol. 1524, pp. 207–226. Springer, Heidelberg (1998)
123. Seide, F., Li, G., Yu, D.: Conversational speech transcription using context-dependent deep neural networks. In: Interspeech 2011, pp. 437–440 (2011)
124. Seide, F., Li, G., Yu, D.: Feature engineering in context-dependent deep neural networks for conversational speech transcription. In: ASRU 2011 (2011)
125. Sohn, K., Zhou, G., Lee, H.: Learning and selecting features jointly with point-wise gated Boltzmann machines. In: ICML 2013 (2013)
126. Stoyanov, V., Ropson, A., Eisner, J.: Empirical risk minimization of graphical model parameters given approximate inference, decoding, and model structure. In: AISTATS 2011 (2011)
127. Sutskever, I.: Training Recurrent Neural Networks. Ph.D. thesis, CS Dept., U. Toronto (2012)
128. Swersky, K., Ranzato, M., Buchman, D., Marlin, B., de Freitas, N.: On autoencoders and score matching for energy based models. In: ICML 2011. ACM (2011)
129. Taylor, G., Hinton, G.: Factored conditional restricted Boltzmann machines for modeling motion style. In: Bottou, L., Littman, M. (eds.) ICML 2009, pp. 1025–1032. ACM (2009)
130. Taylor, G., Hinton, G.E., Roweis, S.: Modeling human motion using binary latent variables. In: NIPS 2006, pp. 1345–1352. MIT Press, Cambridge (2007)
131. Tenenbaum, J.B., Freeman, W.T.: Separating style and content with bilinear models. Neural Computation 12(6), 1247–1283 (2000)
132. Tsianos, K., Lawlor, S., Rabbat, M.: Communication/computation tradeoffs in consensus-based distributed optimization. In: NIPS 2012 (2012)
133. Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Large margin methods for structured and interdependent output variables. J. Mach. Learn. Res. 6, 1453–1484 (2005)
134. Tscher, A., Jahrer, M., Bell, R.M.: The bigchaos solution to the netflix grand prize (2009)
135. Vincent, P.: A connection between score matching and denoising autoencoders. Neural Computation 23(7), 1661–1674 (2011)
136. Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A.: Extracting and composing robust features with denoising autoencoders. In: ICML 2008 (2008)

137. Welling, M., Teh, Y.W.: Bayesian learning via stochastic gradient Langevin dynamics. In: ICML 2011, pp. 681–688 (2011)
138. Weston, J., Ratle, F., Collobert, R.: Deep learning via semi-supervised embedding. In: ICML 2008 (2008)
139. Wiskott, L., Sejnowski, T.J.: Slow feature analysis: Unsupervised learning of invariances. Neural Computation 14(4), 715–770 (2002)
140. Wiskott, L., Sejnowski, T.: Slow feature analysis: Unsupervised learning of invariances. Neural Computation 14(4), 715–770 (2002), `http://itb.biologie.hu-berlin.de/~wiskott/Publications/WisSej2002-LearningInvariances-NC.ps.gz`
141. Yu, D., Wang, S., Deng, L.: Sequential labeling using deep-structured conditional random fields. IEEE Journal of Selected Topics in Signal Processing (December 2010)
142. Yu, K., Lin, Y., Lafferty, J.: Learning image representations from the pixel level via hierarchical sparse coding. In: CVPR 2011 (2011)
143. Zeiler, M.D., Fergus, R.: Stochastic pooling for regularization of deep convolutional neural networks. Tech. rep., New York University, arXiv 1301.3557 (2013)

Challenges and Opportunities of Multilingual Information Access

Christof Monz

Informatics Institute, University of Amsterdam
P.O. Box 94323, 1090 GH Amsterdam, The Netherlands
`c.monz@uva.nl`
`http://www.science.uva.nl/~christof`

Abstract. The amount of digitally available information that is authored in languages other than English has been rapidly increasing over the last decade. This results in a distribution of information not only across different sources but also different languages. While various areas within natural language processing, such as machine translation, information extraction, and cross-language information retrieval, have made substantial advances over the last few years, they have done so mostly by addressing challenges that lie within the confines of their respective sub-area. Multilingual information access on the other hand spans the entire spectrum of these areas, requiring approaches that combine insights and methods that cross the boundaries of these individual research areas. In this talk, I will give an overview of a number of scenarios where cross-language technologies from different areas within natural language processing have benefited from each other in the past, but also discuss some of the challenges and opportunities that lie ahead of us.

Keywords: machine translation, cross-language information retrieval.

Unsupervised Rhyme Scheme Identification in Hip Hop Lyrics Using Hidden Markov Models

Karteek Addanki and Dekai Wu

Human Language Technology Center
Dept. of Computer Science and Engineering
Hong Kong University of Science and Technology, Hong Kong
{vskaddanki,dekai}@cs.ust.hk

Abstract. We attack a woefully under-explored language genre—lyrics in music—introducing a novel hidden Markov model based method for completely unsupervised identifica-tion of rhyme schemes in hip hop lyrics, which to the best of our knowledge, is the first such effort. Unlike previous approaches that use supervised or semi-supervised approaches for the task of rhyme scheme identification, our model does not assume any prior phonetic or labeling information whatsoever. Also, unlike previous work on rhyme scheme identification, we attack the difficult task of hip hop lyrics in which the data is more highly unstructured and noisy. A novel feature of our approach comes from the fact that we do not manually segment the verses in lyrics according to any pre-specified rhyme scheme, but instead use a number of hidden states of varying rhyme scheme lengths to automatically impose a *soft segmentation*. In spite of the level of difficulty of the challenge, we nevertheless were able to obtain a surprisingly high precision of 35.81% and recall of 57.25% on the task of identifying the rhyming words, giving a total f-score of 44.06%. These encouraging results were obtained in the face of highly noisy data, lack of clear stanza segmentation, and a very wide variety of rhyme schemes used in hip hop.

1 Introduction

Among the many genres of language that have been studied in computational linguistics and spoken language processing, there has been an inexplicable dearth of work on lyrics in music, despite the major impact that this form of language has across almost all human cultures. We aim to spur research addressing this gap, by bringing the statistical methods of language technologies to bear upon modeling issues in song lyrics. An ideal starting point for this investigation is hip hop, a genre of music that emphasizes rapping, spoken or chanted rhyming lyrics against strong beats or simple melodies. This complex domain presents a fertile range of challenges for learning since there is typically no obvious structure in terms of rhyme scheme, meter, or overall meaning.

As a first challenge in learning hip hop lyrics, we attack the problem of rhyme scheme identification. Rhyming is central to many genres of music, and is taken to an extreme level of importance in hip hop. Although statistical methods have been applied to identify rhyming patterns in poetry, most of the approaches either restricted themselves to a narrow and structured domain such as sonnets. Some of the approaches incorporated morphological and phonological features that are language specific while some used supervised learning methods using labeled data.

The domain of hip hop lyrics is very *unstructured* when compared to classical poetry which makes it hard for the model to make any kind of assumptions about the data. We use to the term *unstructured* to describe degree of permitted variation in the meter of the lyrics and the large number of colloquial words and slangs that are common in hip hop subculture. The variance in the permitted meter makes it hard to make any assumptions about the stress patterns of verses in order to identify the rhyming words. The broad range of terms used in hip hop make it hard to use of off-the-shelf NLP tools for doing phonological and/or morphological analysis. Problems discussed above are exacerbated by differences in intonation of the same word and lack of robust transcription.[1]

However, hip hop lyrics *do* contain enough information to identify words that rhyme as rhyming is one of the characteristic features of hip hop lyrics. This high degree of rhyming enable us to learn the rhyme schemes in a completely unsupervised manner without making any language dependent assumptions. Identification of rhyme schemes also provides useful information about the overall structure of the songs and the variation of rhyme schemes within a single song. Further, accurate identification of rhyming words will enable compilation of large scale rhyming dictionaries which are currently manually compiled [1]

In this paper, we introduce a novel method for completely unsupervised identification of rhyme schemes in hip hop lyrics that utilizes an HMM model, which to the best of our knowledge is first such effort. We discuss some of the previous work that applies statistical methods to the problems similar to rhyme scheme detection in various domains in section 2. In section 3 we discuss our algorithm and provide motivation for our choice of the HMM structure. Sections 4 and 5 talk about our experimental setup and results respectively. Conclusions and future research directions are proposed in section 6.

2 Related Work

Most of the work in the past involved automatic analysis of poetry to discover word to word relationships, stress patterns [2] and rhyming words [3]. Results were combined with language models to generate new poems [2,4,5].

The stress pattens of words in English rhythmic poetries were analyzed by [2]. Their task was to assign stress patterns to words where the meter of each line is known. A meter is the beat that is heard when the poem is read aloud. For example, Iambic is a common meter where the words sound like da-DUM da-DUM da-DUM. The words were labeled S for light tones and S' for stressed tones, e.g. beLIEVE would be labeled S S'. Finite state transducer was applied to all

[1] `http://languagelog.ldc.upenn.edu/nll/?p=2824`

the words in Shakespeare's sonnets to assign probable stress patterns. Combining stress patterns with language models for fluency, a poem was generated.

Apart from stress patterns, the positions of the words in a line of Chinese poems were also suitable to learn word associations. Researchers have used statistical machine translation (SMT) to generate lines in a Chinese couplet, duilian [5]. Given a single line of a couplet, an n-best list of "translations" that were potential candidates for the next line were generated. Subsequently, each of these lines were analyzed and eliminated according by applying linguistic constrains to obtain the most suitable option.

SMT algorithms were used in conjunction with stress patterns and rhymes found in a pronunciation dictionary to produce translations of poems [4]. Although many constraints were applied in translating full verses of poems, in practice it was difficult to satisfy all the constraints. For example, some of the translations contained rhyming words but did not give the same meter to the line.

Graph theory had been applied to analyze the rhyming and therefore inferred the pronunciation of words in old poetry for an English rhyming corpus [6]. Training data was obtained from poetries where the pronunciation of the words could found in a pronunciation dictionary. The pronunciations were represented by the International Phonetic Alphabet(IPA) symbols. A word was assumed to rhyme with another when the IPA symbols ended similarly. The rhyming words were organized into rhyme graphs where the nodes were words and edges were rhymes. However, this method gave large clusters of words that had the same IPA endings but did not fully rhyme, such as 'bloom' and 'numb'.

Automatic lyric generation given melodies was also investigated [7]. The training data for the model included melodies and the associated lyrics, where the lyrics were represented using the *KNM* system, where *K*, *N* and *M* represented the long vowel, short vowels and consonants respectively. The trained model was then used to label any input melody with the *KNM* system. Subsequently, words were chosen to fit the *KNM* system of the lyrics and constraints were applied to make sure that the lyrics were meaningful.

Most of the past work in this vein can be classified into two categories. The first category uses some form of prior linguistic knowledge about the domain, such as pronunciation dictionaries or phonological or morphological information. The second category uses unsupervised learning methods to identify word association probabilities but enforces harsher constraints warranted by the domain, such as a set number of words in a line, or a set meter. Our current work differs in the sense that we present a completely unsupervised model on a domain that inherently has very few such constraints. For example, not all words in the lyrics are a part of the lexicon. Hip hop does not require a set number of syllables in a line, unlike poems (especially in classical poetry where, for example, an octave has exactly 10 syllables per line and 8 lines per stanza). Also, surprising and unlikely rhymes in hip hop are frequently achieved via intonation and assonance, which makes it hard to apply prior phonological constraints.

Our current work was inspired by the work done on unsupervised identification of rhyming words in poems [3]. To learn the probabilities of rhyming words, a set of chosen hidden rhyming patterns were applied by labeling the last word of each line with a rhyme label, such as **A** in a **ABAB** rhyme scheme. However, unlike the original model our model cannot afford to make any assumptions about the meter of the lyrics and the length of each rhyming scheme. Hence we extend the model to accommodate the more unstructured domain of hip hop lyrics. Further details about our extension are provided in the following section.

3 Detection of Rhyming Words

A collection of hip hop lyrics typically contains *repetition of rhyming pairs*. For example, the word *y'all* rhymes with *real* across different songs by different artists. This can be partly attributed to the scarcity of the rhymes and to the similarity in the pronunciation caused by similarity in the music "beat" patterns.

In this section, we will describe an unsupervised algorithm that harnesses this repetition, based on a verse generation model. This algorithm shares some similarity to the one proposed in [3]. But their model was proposed for English poetry which is much more structured than hip hop lyrics. In order to account for the wide range of variations in the domain of hip hop lyrics, we generalize their model to account for the varying lyric lengths and possible sequences of rhyming patterns in hip hop lyrics. We also use the exhaustive set of all possible rhyme schemes (up to length 4) instead of manually selecting rhyme schemes. The states are selected so that it is possible to generate all the rhyme schemes and can be represented using a minimum number of states.

In this section, we will describe our model to identify the rhyming schemes and the experimental pipeline that was used to obtain and process the lyrics. A brief description of the choice of the rhyme schemes and the underlying motivation is also provided.

3.1 Generative Model of a Verse

We assume that our data is a set of independent and identically distributed verses. We propose the following generative model for a verse given a HMM of rhyming schemes. The rhyming schemes could be of varying length and the choice of the appropriate rhyming schemes is discussed in the subsequent subsections.

1. We have a fully connected HMM with $i+1$ states, where each state represents a different rhyme scheme such as **AA**, **ABAB** *etc.* The output of each state has variable length and transitions are possible from one state to all other states as shown in Figure 1. The start state is not shown for the sake of brevity.
2. For each $l \in [1, n]$, pick a word sequence $x_{1...T}$, choosing the last words and the words before commas in each line as follows[2] :

[2] Manual inspection of the data revealed that words before commas also rhyme in addition to words at the end of the line.

(a) Choose a rhyme scheme r corresponding to the state i, from the start state with a probability of π_i using the transition probability of the HMM.
(b) If, according to r, the i^{th} line does not rhyme with any previous line in the verse, pick as word x_i from a vocabulary of line-end words with probability $P(x_i)$
(c) If the i^{th} line rhymes with some previous line(s) j according to r, choose a word x_i that rhymes with the last words of all such lines with probability $\prod_{j<i:r_i=r_j} P(x_i|x_j)$.
(d) Choose a rhyme scheme r corresponding to the state j with a probability of a_{ij} using the transition probability of the HMM.
(e) Goto step (b).

The probability of a verse $P(x)$ is obtained by summing over all state sequences $\mathcal{S}$ that generate the x as shown in 1, where o_t is the output of state s_t, S is a sequence of states $s_1, \ldots, s_t$ and π_{s_t} denotes the probability that s_t will be the first state in the HMM.

$$P(x) = \sum_{S\in\mathcal{S}} P(x)P(x|S) = \sum_{S\in\mathcal{S}} (\prod_{t\in[1,|S|]} p(o_t|s_t)a_{s_t s_{t+1}} + \pi_{s_1}) \tag{1}$$

The probability of generating a sequence of tokens o_t of length n given a state s_t corresponding to a rhyme scheme r (similar to the emission probabilities in traditional HMMs) is given by 2. $I_{i,r}$ is the indicator variable for whether line i rhymes with at least one previous line under r.

$$P(o_t|s_t) = P(x|s_t) = \prod_{i=1}^{n} (1 - I_{i,r})P(x_i) + I_{i,r} \prod_{j<i:r_i=r_j} P(x_i|x_j) \tag{2}$$

3.2 Learning

We denote our data by $\mathcal{X}$, a set of verses. Each verse x is represented as a sequence of the line-end tokens and words before commas, $x_1, \ldots, x_l$. We also have a fully connected HMM, with a finite number of states, that is capable of producing all the partitions of up to length 4 (details are given the next subsection) via different path sequences. If each verse in the data is generated independently the log likelihood of the data is $\sum_{x\in\mathcal{X}} \log P(x)$. We apply expectation maximization learning algorithm to maximize this over all possible rhyme sequences produced by the HMM. While doing so, we re-estimate the latent variables θ, which represents a *pairwise rhyming strength*, and π_i,a_{ij}, the initial and transition probabilities for each of the states in the HMM. The distribution of rhyme schemes $\theta_{v,w}$ is defined for all words v and w as a non-negative real value indicating how strongly the words v and w rhyme.

The formulation described above is slightly different from the traditional parameter learning problem of HMMs in the sense that states do not have a multinomial distribution in terms of the emitted variables (or alphabets in a

FSA formulation). They do however, share the distribution of *pairwise rhyming strength* parameter, θ that needs to be updated instead of the emission probabilities in the maximization step. The $\theta_{v,w}$ reflect the co-occurrence of v and w which in turn indicates the probability that they might be rhymes.

Initialize: θ,π and a uniformly (giving θ the same positive value for all word pairs that co-occur in a verse)

Expectation Step: Compute $P(r|x_{u,\ldots,v}) = P(s_i|x_{u,\ldots,v})$, where r is the rhyme scheme of the state s_i. $P(s_i|x_{u,\ldots,v}) = \alpha_i(u,\ldots,v)\beta_i(u,\ldots,v)$ which indicates the probability of being in the state i given the tokens in the sequence. The likelihood of generating the sequence $x_{u,\ldots,v}$ given a rhyme scheme r (or a state s_i) is

$$P(x|r) = \prod_{i=1}^{n}(1 - I_{i,r})P(x_i) + I_{i,r} \prod_{j<i:r_i=r_j} \frac{\theta_{x_i,x_j}}{\sum_w \theta_{w,x_j}} \tag{3}$$

$P(x_i)$ is simply the relative frequency of the word x_i in the data.

Maximization Step: Update θ, a and π, where the state s_t has a rhyming scheme r

$$\theta_{v,w} = \sum_{r,x:v \text{ rhymes with } w} P(r|x) = \sum_{r,x:v \text{ rhymes with } w} \gamma_{s_t}(x) \tag{4}$$

$$\gamma_x(s_t) = \frac{\alpha_x(s_t)\beta_x(s_t)}{P(x)} \tag{5}$$

where $\alpha_x(s_t)$ and $\beta_x(s_t)$ are the forward and backward probabilities for the state s_t outputting x. The update rules for the transition probabilities are similar to that of a usual HMM formulation where $u - w$ is the rhyme length of state i.

$$\xi_{u,v}(i,j) = \frac{\alpha_{w,u}(i)a_{ij}P(x_{u,\ldots,v}|j)}{P(x)} \tag{6}$$

$$a_{ij} = \frac{\sum_{\forall u,\forall v} \xi_{u,v}(i,j)}{\sum_{\forall u,\forall v} \gamma_{u,v}(i,j)} \tag{7}$$

where $\gamma_{u,v}(i,j)$ is the same as the one mentioned in 5.

After convergence: Label each verse x with the best rhyme scheme, $\arg\max_{S\in\mathcal{S}} P(S|x)$

3.3 Choosing the HMM States

As mentioned earlier, the total number of rhyming patterns given a sequence of length n is n^{th} Bell number[3] which is also equal to the total number of partitions of size n. It is evident that the number of states grow exponentially

[3] http://oeis.org/A000110

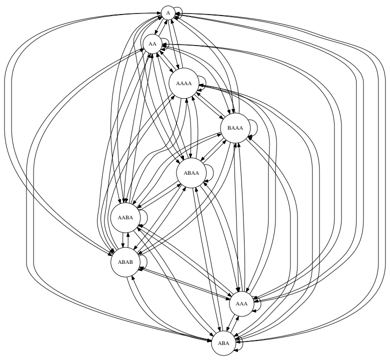


Fig. 1. Fully connected Hidden Markov Model

as n becomes larger and exhaustively considering all the rhyme schemes for a given verse could explode the size and therefore the computational complexity of the training algorithm. In addition to that, considering most of these rhyming schemes is not helpful for two reasons: (1) rhyme schemes that have rhyming words seperated over long distances (such as **ABCDA**) are meaningless as the listeners cannot keep track of more than previous four lines, and (2) it becomes very hard to find rhyming word group that contain more than five or six words that provide relevant meaning to the lyrics. Keeping these reasons in mind, we have decided to place an upper bound of four on the length of the rhyme schemes. This helps keep the computation tractable while providing enough flexibility and co-occurrence counts.

The total number of partitions for all the sequences of length 4 are 23. However, we do not need to exhaustively include all of them as most of the sequences can be generated from sequences of smaller length. For example, a rhyme scheme of length 3 **AAB** can be generated by a sequence of rhyme schemes of length 2 and 1, **AA** and **B** respectively. Therefore, for $n = 1$ to 4, we only choose those sequences that cannot be generated from smaller sequences.

After the application of above constraints, we finally end up with the following 9 states **A**, **AA**, **ABA**, **AAA**, **ABAB**, **AABA**, **ABAA**, **BAAA**, **AAAA** which makes the HMM significantly compact compared to exhaustively considering all possible rhyme schemes. We hope that the transition probabilities will help achieve the necessary bias towards states that are not explicitly represented in the HMM.

4 Experiments

4.1 Dataset

We exploited the vast amount of user generated hip hop lyrics available online. We implemented a web crawler to scrape websites that offer hip hop lyrics.

We downloaded the lyrics of about 52,000 hip hop songs (about 800MB of raw html content). The data was cleaned by stripping HTML tags, various metadata (e.g., the artist, song, lines corresponding to a chorus, beats, etc.) Verses were identified using simple heuristics and marked up. We also normalized for special characters and case differences, and filtered out any malformed HTML tags left in the data. We then extracted the end-of-line words and words before all the commas from each verse. We obtained a corpus containing 260,000 verses with 4.2 Million tokens (with around 153,000 unique tokens).

4.2 Evaluation

The performance of our model was evaluated on the task of labeling a given verse with the rhyme schemes. As there are currently no annotated gold standard corpora for the evaluation of performance on such a task we performed manual evaluation. As our model is completely unsupervised, we chose a random sample of 75 sentences from our training data as our test set. Two native English speakers were asked to annotate the verse with a gold standard rhyme scheme. They were asked to segment any rhyme schemes of length more than four appropriately. Precision and recall were aggregated for the Viterbi parse [8] of each verse against this gold standard and the f-score was calculated.

5 Results

In this section, we present the results of our experiment. Firstly, we report the number of types and tokens in our corpus and make an empirical comparison of the frequency of the types to that of a Zipfian distribution to establish that our corpus is comparable with other naturally occurring natural language data [9]. In the next subsection, we present some examples of the identified rhyming pairs and discuss some errors that the model is prone to making. Finally, we discuss precision and recall scores on a randomly selected subset of the training data.

5.1 Cumulative Probability Distribution

The occurrences of each of the 153673 words in all the songs were counted, given by the cumulative frequency. The words were organized into bins where each bin represented a cumulative frequency interval of 100, i.e. the bin ranked 1 will have words that appeared between 0-100 times. The total number of words in each bin was counted to get the probability distribution over the range of frequencies. Figure 2 showed a graphical illustrations of the cumulative probability distribution of each bin against rank of the bin.

Results show that the cumulative probability of the words and the ranks follow a distribution similar to the the Zipfian distribution to a certain extend. The law states that many words appear only a few times in the language and a few words appear many times, which can be roughly approximated by $1/Rank$ where the most occurred word occurred approximately twice as much as the

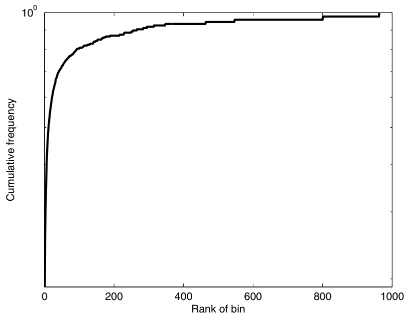


Fig. 2. Frequency density of word occurrence in the corpus

second and three times as much as the third. This relationship is said to hold for any corpus in natural language utterance. The similarity suggests that the data used here is a representative sample of naturally occurring data and our method can be applied to other hip hop lyrics corpora with similar distribution and expect similar performance.

5.2 Rhyme Scheme Variations

In this section, we briefly describe some observations regarding the occurence of rhyme schemes as learned by our model. As is the model bias with HMMs, the transitional probabilities are higher when the transitioning state covers a longer span. Rhyme schemes typically begin with states **ABAB**, **AABA** and **ABAA**. In fact, there is a strong bias against beginning with states that correspond to smaller rhyme schemes, which is a weakness. We have observed that this strong bias towards states with large rhyme schemes sometimes results in improper segmentation i.e., the model chooses to assign a longer rhyme scheme to certain lines in the verse where using a short rhyme scheme would improve the overall score. Somewhat surprisingly, we noticed that the transition probability from the state **AAAA** to the state **AA** is almost 1. Upon data inspection, we observed that this was due to the presence of chorus lines in the data which contained successive repeated lines thereby biasing the model towards transitioning to the state **AA**. The presence of chorus lines also explained the very high transition probabilities for transitions from states **ABAB**, **AABA** and **ABAA** to **AA**.

5.3 Preprocessing Errors Cause False Positive Rhyme Pairs

In order to determine whether or not our model identifies rhyming words correctly, we manually inspected the rhyming pairs with highest pairwise rhyming strength. From Table 1, we can notice that the model does identify rhyming word pairs fairly accurately. The model also identifies rhyming words that are not in English such as “sonorisateur – l’heure”. Most of the pairs with high scores were

Table 1. Samples from some of the top rhyming pairs

examples of rhyming pairs		log probability
sonorisateur	l'heure	0
seazin	weezin	0
rieces	telekineses	0
on	longjohn	-0.000608862
in	homosapien	-0.0018521
convenant	parent	-1.05447
convenant	it	-1.09564
convenant	transparent	-1.14791
convenant	terminated	-45.0261
convenant	loot	-48.0058
convenant	incarcerated	-102.884

fairly accurate although we found some pairs which were not really rhymes but were semantically related such as "*steppenwolf – wild*", "*voldemort – horcruxes*" and "*eenie-meenie – minie-mo*". This is due to the fact that these tokens occurred very infrequently and were mislabeled as rhymes because there was no other way to explain the data given our model. We also show an example of the top pairwise rhyming strengths of the word "*covenant*" with other words in Table 1. The rhyming words "*parent*" and "*transparent*" are identified correctly and the pairwise rhyming strength is low for others. The rhyming pair "*it*" is incorrect but has a high probability. Upon inspecting the data, we noticed that this was due to the presence of a comma after "*it*" in the lyric which caused it to be considered as a likely rhyming candidate. Such preprocessing errors were a major source of errors in rhyme pair identification and most of the false positives contained stop words before a comma. We believe better preprocessing heuristics will help alleviate the problem and improve the accuracy of our model.

5.4 Measurement of Labeling Accuracy

We obtain a precision of 35.81% and a recall of 57.25% giving an f-score of 44.06%. We find these results to be very encouraging in the face of highly noisy data, lack of clear stanza segmentation, and a very wide variety of rhyme schemes used in hip hop. Table 2 shows an example verse labeled with the viterbi parse using the model after the 10 iterations of EM. The stanzas "*sicker—remenants—liquor*" and "*radiation—head-ahead—instead*" are labeled correctly, while the stanza "*are—these—leave—tv*" has "*these*" incorrectly labeled as rhyming with "*leave*" and "*tv*". The model is prone towards committing such false positive errors especially for tokens that do not belong to the rhyme scheme which also explains the lower precision and higher recall. This is due to the inherent bias of HMM models towards minimizing the number of transitions. Hence, the token is incorrectly added to the rhyme scheme instead of creating a separate state for the single token. A possible solution could be to introduce an appropriate bias in

Table 2. Viterbi labeling of the example

The clock ticks, —	**A**
— your name off the list	**B**
Life moves quick so we gotta move quicker	**A**
The world's devilish, —	**B**
— makin me sicker	**A**
I smell the remanents, —	**B**
— of drugs sex and liqour	**A**
The kids are, —	**B**
— immune to these	**A**
Cos they leave, —	**A**
— they get a fix on they new tv	**A**
Parents dont need to talk to their kids no more	**B**
They feed em videos, —	**A**
— show em the internet	**A**
Log on to the porn site, —	**A**
— here we go	**B**
Television, —	**A**
— the drug of a nation	**A**
Breeding ignorance, —	**A**
— feeding radiation	**B**
Frenzhal hit the nail on the head	**A**
Wish we were sailing ahead	**A**
But we're shipwrecked and failing instead	**A**
But the tests have changed,	**B**
— we're all passin	**A**
So long as your a expert in arson	**A**
I'm askin, —	**A**
— which ways up,	**B**
cos my compass has gone	**A**
You know what? I'm a let my beatin heart be my guide through the eye of the	**A**
storm	**A**

the initial transition probabilities to counteract the model bias of choosing longer rhyme schemes. Also, noisy transcription of lyrics such as incorrectly placing the last token "*storm*" on a separate verse line, affect the performance of the model.

6 Conclusion and Future Work

We presented a novel unsupervised algorithm to identify the rhyme schemes in hip hop lyrics. We presented the learning algorithm, discussed the nature of the corpus and presented precision and recall against a manually annotated gold standard. These results are very encouraging given that our task is highly unstructured compared to similar tasks performed on other domains . Also, no clear segmentation of rhyme schemes makes our task more challenging. In the future, we would like to experiment with different rhyme scheme selections and weighting schemes for the transition probabilities. Also, the rhyme pairs learnt in the model can be used in tandem with language models to generate fluent and rhyming hip hop lyrics.

Acknowledgements. This material is based upon work supported in part by the Hong Kong Research Grants Council (RGC) research grants GRF620811, GRF621008, FSGRF13EG28, and GRF612806; the Defense Advanced Research Projects Agency (DARPA) under BOLT contract no. HR0011-12-C-0016, and GALE contract nos. HR0011-06-C-0022 and HR0011-06-C-0023; and by the European Union under the FP7 grant agreement no. 287658. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the RGC, EU, or DARPA. We would also like to thank Belsy Yuen and Subrotho Mondal Kumar for their help in human evaluation and data analysis.

References

1. Mitchell, K.: Hip-hop rhyming dictionary. Alfred Publishing Company, Incorporated (2003)
2. Greene, E., Bodrumlu, T., Knight, K.: Automatic analysis of rhythmic poetry with applications to generation and translation. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pp. 524–533. Association for Computational Linguistics (2010)
3. Reddy, S., Knight, K.: Unsupervised discovery of rhyme schemes. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers, vol. 2, pp. 77–82. Association for Computational Linguistics (2011)
4. Genzel, D., Uszkoreit, J., Och, F.: Poetic statistical machine translation: rhyme and meter. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pp. 158–166. Association for Computational Linguistics (2010)
5. Jiang, L., Zhou, M.: Generating chinese couplets using a statistical mt approach. In: Proceedings of the 22nd International Conference on Computational Linguistics, COLING 2008, vol. 1, pp. 377–384. Association for Computational Linguistics, Stroudsburg (2008)
6. Sonderegger, M.: Applications of graph theory to an english rhyming corpus. Computer Speech & Language 25(3), 655–678 (2011)
7. Ramakrishnan A, A., Kuppan, S., Devi, S.L.: Automatic generation of tamil lyrics for melodies. In: Proceedings of the Workshop on Computational Approaches to Linguistic Creativity, pp. 40–46. Association for Computational Linguistics (2009)
8. Forney Jr., G.: The viterbi algorithm. Proceedings of the IEEE 61(3), 268–278 (1973)
9. Manning, C., Schütze, H.: Foundations of statistical natural language processing. MIT press (1999)

Complementarity of Lexical Cohesion and Speaker Role Information for Story Segmentation of French TV Broadcast News

Abdessalam Bouchekif, Géraldine Damnati, and Delphine Charlet

Orange Labs, 2 avenue Pierre Marzin 22307 Lannion, France
{abdessalam.bouchekif,geraldine.damnati,delphine.charlet}@orange.com

Abstract. Topic boundary detection in French TV Broadcast News is addressed in this paper with an approach based on the combination of two views: lexical cohesion and speaker role analysis. We propose an improved selection strategy from the classical lexical cohesion curve as well as an integrated supervised classification approach that jointly exploits the two views. The combination of these methods leads to significant improvements on a rich French database composed of shows from 7 different channels.

Keywords: Topic boundary detection, lexical cohesion, speaker role.

1 Introduction

Story segmentation is the task of detecting topic boundaries in a given content. It has received attention for TV Broadcast News (TVBN) segmentation through several research programs (SDR,TDT, TRECVID), but still remains a problem of interest. In particular, previous studies applied on a corpora of *traditional* TVBN format while channels nowadays are trying to differentiate from this and propose some more modern settings.

The most commonly adopted thematic structuration consists to use of packages: the anchor speaker in studio introduces a new topic, followed by a report or an interview. However, some shows propose variations in this structure. For this study, a corpus of TVBN shows from 7 different French channels has been considered. These channels can be separated in two categories. The first one corresponds to the three main French generalist channels (TF1, France2, France3) and can be grouped under the term Big Three, analogously to the American Broadcast network typology. They propose a more "classical" TVBN show, whether at midday or at night. The other one corresponds to specialized News channels (LCI, France24) or generalized channels but with more "modern" structuration of their shows (Arte, M6). Some are fully composed of anchor voice-overs (a video topic narrated by the anchor speaker) without any studio scene. In this case (eg. for M6), the anchor is not visible. Some shows may also contain *readers*, where the anchor reads a (usually short) topic without any additional illustration. Several short *readers* can occur one after the other. A topic may also not involve the anchor person at all. For instance, in the middle of the Arte shows, a succession of short voice-overs narrated by a reporter is included among traditional packages. At the end of some France3 shows, series of local news package extracts are added to the national News show, each of them being narrated by their corresponding local reporter. Taking into account all this diversity, makes it a difficult segmentation task.

Three categories of cues or features have been explored in BN: **lexical** (lexical cohesion, chaining strength), **acoustic** (pause duration, speaker change, speech type) and **visual** (new title caption, anchor face) cues. This latter can be very efficient when designed for a specific show but not always generic. Indeed, Anchor face isn't guaranteed at all times (e.g channel M6). Title Captions can be a good indicator of topic transitions, but approaches based on this cue fail if there is no caption. Lexical cues [21] and audio evidence of the anchor speaker [19] are turning out to more generic cues and we have chosen to focus on these features. Many topic segmentation systems for BN are based on lexical cohesion over automatic transcription of the spoken content. For example in [2], [22] and [15] the authors focus on lexical Similarity Computation which reveals topic boundaries via semantic variations across the transcription.

In this paper, we rely on the audio, not exploiting any information from the video. We propose to exploit lexical cohesion and speaker role analysis in several ways, whether through unsupervised boundary selection approaches or supervised classification approaches. We significantly improve the unsupervised selection approach by suitable use of speaker role information and further increase performances by merging the unsupervised and supervised approaches.

The rest of this paper is divided into five sections. In Section 2, we describe the lexical cohesion approach implemented in this study. In section 3, the integration of structural information is presented. The details of the properties of our database and present experimental results are given in Section 4. Finally in Section 5, we discuss the related works.

2 Lexical Cohesion Based Segmentation

In order to compute lexical cohesion in this study, we have implemented a variation of the TextTiling algorithm with slight adaptation to the context of processing automatic transcriptions of speech. First, the segment units under consideration are *breath groups* (BG): sequences of words between two pauses in a speech turn. Thematic boundaries are searched among potential boundaries between each adjacent BG pairs. Lexical units are lemmas obtained thanks to the *lia_tagg*[1] probabilistic syntactic parser. For a given show, the set of lexical units (or tokens) is the set of different lemmas obtained after discarding function words and words whose confidence score is below a given threshold.

[1] http://pageperso.lif.univ-mrs.fr/~frederic.bechet/download.html

2.1 $TF - IDF$ Weighting

In Information Retrieval (IR), $tf - idf$ weighting is widely used to translate the capacity of a term t to discriminate the document d relatively to a collection of documents. Usually, $tf - idf$ coefficients are estimated on large corpora (e.g [9]). Following [13] who propose a segmentation algorithm for spoken lectures, we have chosen to be independent to any external data information and to estimate $tf - idf$ only from the content of the BN show which is being processed.

In [13], the authors splitted the show into N uniform chunks, each of them representing the notion of document in the classical IR.

A term t in a breath group x will be associated to a weight $w(c(x), t)$ depending on the chunk $c(x)$ in which it occurs. In the following equation, $c(x)$ is the index of the chunk containing the BG x.

$$w(c(x), t) = tf_{c(x),t} \times idf_t, where \quad idf_t = \log(\frac{N}{n_t}) \tag{1}$$

where $tf_{c(x),t}$ is the number of times the term t occurs in chunk c_x and n_t is the number of chunks containing term t.

2.2 Similarity Computation

TextTiling considers a sliding window of size k and consists to compute the lexical cohesion for each potential boundary between preceding and following blocs of k BGs. The cosine similarity is adapted in order to integrate chunk based on $tf - idf$ weighting.

For a given potential boundary i between BGs x_i and x_{i+1}, the similarity (or lexical cohesion) between the k preceding BG_s (b_i) and the k following BG_s (b_{i+1}) is:

$$sim(i) = \frac{\sum_t \left(\left(\sum_{x \in b_i} f_{x,t} \times w\left(c\left(x\right), t\right)\right) \times \left(\sum_{y \in b_{i+1}} f_{y,t} \times w\left(c\left(y\right), t\right)\right)\right)}{\sqrt{\sum_t \left(\sum_{x \in b_i} f_{x,t} \times w\left(c\left(x\right), t\right)\right)^2} \times \sqrt{\sum_t \left(\sum_{y \in b_{i+1}} f_{y,t} \times w\left(c\left(y\right), t\right)\right)^2}} \tag{2}$$

where $f_{x,t}$ is the frequency of term t in BG x.

The number of chunks N is computed automatically for each show as a function of its overall duration and the average topic duration estimated on a held-out set of shows.

The benefit of our Similarity Computation approach with the intra-show scheme is that, it does not require *a priori* information unlike other approaches. For example, [9] make use of a keyword extraction tool in [12] for the $tf - idf$ computation which is trained on a large news database. The authors of [2] use pivot-documents in similarity computation. Blocs of breath groups are considered similar if they are similar to the same pivot-documents.

2.3 Boundary Selection in the Unsupervised Approach

Several strategies have been explored to select boundaries from the lexical cohesion curve plotted along all the BGs of a show. The classical boundary selection approach [11] consists to detect valleys as the lowest point between two peaks (local maxima), compute their depth as the sum of the differences between the lowest point value and the left and right peaks values, and select x_i as a boundary if the valley depth $d(i)$ is above a given threshold (thresholding approach). Note that $d(i)$ is equal to 0 when there is no valley at the i^{th} position.

Similarity measures based on sparse matrices are prone to local phenomena which make it difficult to exploit the lexical cohesion curve with the classical threshold on valley depth method. Several deep valleys can occur in a very short time interval or a thematic boundary may result in several successive valleys with low depth value.

We introduce an iterative algorithm which directly exploits lexical cohesion, and builds partition S of a show in segments.

Iterative Splitting Algorithm

1. For initialisation, the entire show is placed in S as one coherent segment.
2. Each segment is split into two, the split point is the minimum value of score (if below a fixed threshold).
3. BGs within a time interval around the selected split point are discarded for the next iterations.
4. The resulting segments are submitted to step 2.
5. Algorithm should be discontinued if there is no possible partition.

Step 3 guarantees that local phenomena with several very close low values of lexical cohesion will not lead to several consecutive topic boundary selections.

3 Integration of Lexical and Structural Information

3.1 Selection Criterion in the Unsupervised Approach

If the latter approach is more robust to local minima, it remains dependent on a threshold which has to be set globally. Without any other information source, it remains likely to generate too many false alarms if we want to achieve a satisfactory level of recall. In order to alleviate this drawback, we propose to add structural information in the boundary selection process. Speaker role information has been integrated with two perspectives:

– Posterior validation of boundary hypotheses: apply detection on the whole document, and select only those hypotheses that correspond to the anchor.
– Prior selection of candidate boundaries: apply boundary detection only in anchor speaker turns.

These two paradigms can apply to both the thresholding approach and the split approach. As valley detection is a local process, posterior validation and prior selection are equivalent, which is not the case for the split approach. Notice that these paradigms assume that all topic boundaries are associated to the anchor speaker which is not always the case as mentioned in the introduction. This will be commented in the experimental part.

3.2 Features for Supervised Classification

In the previous section, we have proposed various selection strategies in order to better exploit the lexical cohesion curve. In this section we propose to integrate structural and lexical cohesion information through machine learning approaches. Each BG x_i has to be classified into boundary or non-boundary, thanks to a statistical classifier that relies on the following set of features: lexical cohesion value $sim(i)$, valley depth $d(i)$ and speaker role.

Two statistical classifiers have been used. First, a classifier based on Conditional Random Fields (CRF) has been designed. If $CRFs$ are more traditionally used for sequence labeling tasks we have chosen to use them for this binary classification task because of their ability to efficiently model contextual information. Features associated to the preceding and following BG are considered, as well as the combination between $sim(i)$ and $d(i)$. An overall bigram on *BGs* is also implemented. In order to optimally convert numerical cohesion and depth value into discrete features, the *discretize4crf* tool implementing the MDLPC optimization method [5] is used[2]. On the other hand, the *icsiboost* classifier [4] has been implemented for this binary classification task with a number of iterations equal to 600. If CRFs are able to model contextual information, *icsiboost* handles numerical values in a finer-grained way. For instance for our data, more than half of the iterations choose as a weak classifier a threshold on $sim(i)$, (with around 100 different thresholds).

Due to this complementarity, we have chosen to use both classifiers and to merge their hypothesis with the merging strategy described in the next section.

3.3 Merging Boundary Hypothesis

Multiple boundary hypotheses coming from various approaches (supervised and unsupervised) can finally be fused according to the following process: first a new set of hypothesis is considered by simply taking the union of the initial sets, then close boundaries are merged. The merging step is performed iteratively: find the 2 closest boundaries, replace them by one boundary (e.g. the temporal average), repeat iteratively while the temporal span between the 2 closest boundaries is below a given threshold (e.g $10s$).

4 Experimental Framework

4.1 A Multi-channel Database

Experiments are carried out on a set of 33 French TVBN shows from 7 different channels. The first set (BigThree) corresponds to the three main French generalist channels (TF1, France2, France3). The other set corresponds to specialized News channels (LCI, France24) or generalist channels but with more modern structuration of their shows (Arte, M6). As can be seen in Table 1, shows from

[2] `http://www.irisa.fr/texmex/people/raymond/Tools/tools.html`

Table 1. Corpus description

	BigThree	Other	Total
Number of shows	18	15	33
Av. duration	$\approx$ 30 min	$\approx$ 15 min	$\approx$ 33 min
Nb. of topic boundaries	366	113	379
Av. duration of topics	124s	91s	115s

these channels are shorter on average, with shorter thematic segments. Shows from the Big Three propose twice as much topics in a show but last three times as longer. TVBN shows from other channels have a steadier rhythm.

In all cases, TVBN shows are presented by one anchor speaker who is not always visible but this is not a problem since our approach only relies on audio information. A topic may not involve the anchor person at all: this is the case for 5.8% of the topic boundaries in our corpus.

4.2 Experimental Framework

Automatic transcription is performed with an industrial Speech to Text (STT) engine: the VoxSigma speech recognizer V3.5 from Vocapia Research, based on LIMSI technology [6]. It achieves 16.1% word error rate on our corpus. Words that have a confidence measure below 0.5 are discarded. The window size for lexical cohesion has been optimized on a held-out corpus and set to 16. Speaker role analysis is performed following the multi-stage process described in [3].The first stage determines the anchor speaker through a specific speaker clustering subtask with temporal distribution information. The remaining speaker turns are classified into reporter or other on the basis of lexical, structural and acoustical information. This process achieves 90% accuracy at speaker turn level [3].

The speaker role classification system has been trained on automatic transcriptions and automatic speaker turn segmentation. Hence our overall system only needs training data labeled in terms of speaker role and topic boundaries. All experiments dealing with supervised classification are achieved through the *leaving-one-out* framework, each show being processed by a model trained on all the remaining shows. As can be found in several other studies, the first and the last topics of a show are discarded when they correspond respectively to the titles presentation or summary. These segments should be processed with a different approach. Performances are measured in terms of recall and precision by comparing time information associated to hypotheses and reference boundaries. The timestamp of a boundary hypothesis is the beginning of the BG immediately following it, while a reference timestamp corresponds to the beginning of the first speaker turn of the topic segment. The assumption that boundaries are concomitant to breath groups is reasonable: the average distance between our reference boundaries and the beginning of the BG containing it is $1.1s$ ($2.1s$ standard deviation). Following other studies in the literature, an interval of $10s$ before and after a boundary hypothesis is tolerated in order to decide if it is correct.

5 Experimental Results

Fig. 1 illustrates unsupervised approaches performances. The Valley curve corresponds to the classical selection strategy associated to the original TextTiling algorithm. For $Valley\_posteriorvalid$, only boundaries corresponding to the anchor speaker are kept. Both curves are plotted by varying the threshold on the valley depth. Posterior validation increases the F-max by 15.6% absolute.

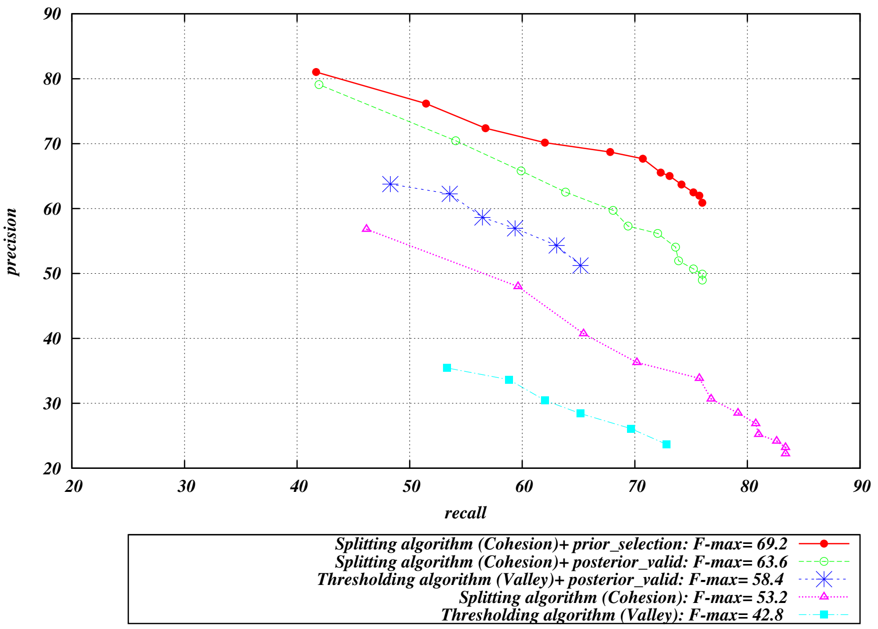


Fig. 1. Recall and precision for the unsupervised approaches

The results obtained with the baseline TextTiling Valley approach (42.8% F-measure) are slightly better than the ones presented in [7] (38.8% F-measure on France2 shows), it can be explained by a more robust implementation of tf-idf relying only on the current content. The Split strategy without any validation already provides better performance than Valley with a 53.2% F-max. All Split curves are plotted by varying the threshold on lexical cohesion. Both anchor based selection approaches further improve performances, with an advantage for prior selection with a 69.2% F-max.

This first set of experiments reflects that significant improvements can be achieved on the basis of the same lexical cohesion curve with appropriate boundary selection strategy and appropriate use of speaker role information, with a 61% relative improvement over the baseline thresholding approach. It is interesting to note that anchor information itself is not sufficient. In fact, systematically selecting the first BG of an anchor speaker turn as a topic boundary would yield a 57.5% F-measure (58.6% recall and 56.5% precision).

Supervised classification approaches alone perform poorly (36.1% F-measure for boost and 49.0% F-measure for CRF) but they both have a precision rate above 80%. Actually, the binary classification process suffers from the unbalanced data issue as the proportion of topic boundaries among the total amount of BGs is low. Merging the two supervised approaches (boost + crf) yields 53.9% F-measure. Table 2 shows how merging hypotheses obtained with the best unsupervised approach can further improve performances. 2.5% absolute improvement in F-max is obtained when merging with boost + crf.

Table 2. Merging with supervised approaches

Split priorselect	recall	precision	F-measure
alone	70.7%	67.7%	69.1%
with crf	72.9%	69.5%	71.2%
with boost	73.2%	68.6%	70.8%
with boost+crf	74.2%	69.1%	71.6%

It is important to notice that the tolerance value for evaluation has a significant influence. For instance, for the last experiment with boost + crf, raising the tolerance from 10s to 15s increases F-max from 71.6% to 78.4% (6.8% absolute gain). 15*s* is presented as the TDT standard in [1] but we kept 10s in order to be compliant with studies on French. Furthermore, 10*s* seems more reasonable from an applicative point of view.

Another advantage of supervised approaches is their ability to easily integrate new features. We have run a set of experiments simply adding the duration of BGs with both classifiers. In fact, the average duration of BGs at topic boundaries is 9.8*s* compared to an overall 5.9*s* for all BGs. Simply adding this information yielded 73.3% F-max.

We have performed separate evaluation on shows from the BigThree channels. As show in table 3 the last system performing 73.3% F-measure, yields 75.4% on the BigThree subset and 67.9% on the other subset.

The BigThree channels are easier to process. As mentioned in section 4.1, thematic segments for the other channels are shorter which it can explain that lexical cohesion doesn't perform as well as for the BigThree. A separate tuning for this type of shows could be achieved in future work. Furthermore, the proportion of topic boundaries that are not related to anchor is lower for the BigThree (3.8%) than for the others (11,8%).
Despite all this, our approach still yields improvement for all channels. Finally, our best system achieves 77.2% recall for 69.7% precision. We have tried to extend the paradigm of the unsupervised approach selection from anchor only to anchor+ reporter, allowing boundaries to be searched in reporter turns, but we were not able to improve the performance. Adding role information in the non-anchor BGs for training supervised models didn't improve performance either. More training data would be necessary.

Table 3. Performance of the best system on the different types of shows

	recall	precision	F-measure
BigThree	79.1%	72.1%	75.4%
Other	72.4%	63.9%	67.9%
ALL	72.2%	69.7%	73.3%

6 Related Work

Story segmentation has been studied in the domain of Broadcast News and, more recently for multi-party meeting segmentation (as in [17] with the CALO Meeting Assistant System or in [10] for leader detection in multi-party meetings) or lecture segmentation in [13]. In the domain of TVBN, most of the studies have been carried out on English and Mandarin, as well as Arabic [14]. Some works [7] [8] have focused on unsupervised approaches and proposed improved estimation of lexical cohesion for processing automatic transcriptions. Lexical cohesion has been used with the traditional Textiling approach [21] or exploited with more sophisticated algorithms (eg. with adaptive language modeling in [8], [18]). We have chosen to focus on lexical cohesion and to design efficient strategies to exploit linear lexical cohesion curve. Several studies have proposed to train supervised models to detect topic boundaries, exploiting lexical and multimodal features. Lexical feature extraction in [16] and [1] consists to analyse the transition words that are likely to introduce a new topic. We have not used this type of features but have relied on lexical cohesion instead as in [21], [20]. Several non-lexical features have been explored in the literature: prosodic [16], structural from audio [1] and multimodal [4] [20]. In this paper, we haven't included so far any multimodal information but we have focused on speaker role information. The authors of [20] and [21] also exploit anchor speaker information, but contrarily to our French TVBN shows, their shows are presented by two anchors. It is the anchor speaker change that is used as an indicator for topic boundary, which is more favorable situation compared to our situation with a single anchor speaker. In this context, [20] report a 79.8% F-measure on Chinese BN using CRFs with multimodal features, on a set of shows from two channels, with a 15s tolerance. In this study we have used a diverse corpus composed of 7 different TV channels and have proposed a robust approach, with an overall 73.3% Fmeasure, with audio only features and a 10s tolerance.

7 Conclusion

We have proposed an integrated system that exploits lexical cohesion and speaker role information. We have proposed both an improved boundary selection strategy in the unsupervised framework, and a supervised approach that exploits structural and lexical cohesion features. The system is robust to various TVBN show formats. It performs better on "traditional" shows but also provides good performance on less classical types of TVBN shows. Only considering anchor detection yields 57.5% F-measure, we were able to achieve an F-measure of 53.6% with lexical cohesion only, and 71.6% through the integration of both information sources. As a perspective to this work, several other structural features can be added to the supervised approach and the detection of these particular topic boundaries that are not linked with anchor speaker still need to be handled.

References

1. Amaral, R., Trancoso, I.: Exploring the structure of broadcast news for topic segmentation. In: Vetulani, Z., Uszkoreit, H. (eds.) LTC 2007. LNCS, vol. 5603, pp. 1–12. Springer, Heidelberg (2009)
2. Claveau, V., Lefèvre, S.: Topic segmentation of tv-streams by mathematical morphology and vectorization. In: INTERSPEECH, pp. 1105–1108 (2011)
3. Damnati, G., Charlet, D.: Multi-view approach for speaker turn role labeling in tv broadcast news shows. In: INTERSPEECH, pp. 1285–1288 (2011)
4. Dumont, E., Quénot, G.: Automatic story segmentation for tv news video using multiple modalities. Int. J. Digital Multimedia Broadcasting (2012)
5. Fayyad, U.M., Irani, K.B.: Multi-interval discretization of continuous-valued attributes for classification learning. In: International Joint Conference on Artificial Intelligence, pp. 1022–1029 (1993)
6. Gauvain, J.L., Lamel, L., Adda, G.: The limsi broadcast news transcription system. Speech Communication 37(1-2), 89–108 (2002)
7. Guinaudeau, C.: Structuration automatique de flux télévisuels. Thèse, INSA de Rennes (2011)
8. Guinaudeau, C., Gravier, G., Sébillot, P.: Enhancing lexical cohesion measure with confidence measures, semantic relations and language model interpolation for multimedia spoken content topic segmentation. Computer Speech and Language 26(2), 90–104 (2012)
9. Guinaudeau, C., Hirschberg, J.: Accounting for prosodic information to improve asr-based topic tracking for tv broadcast news. In: INTERSPEECH, pp. 1401–1404 (2011)
10. Hadsell, R., Kira, Z., Wang, W., Precoda, K.: Unsupervised topic modeling for leader detection in spoken discourse. In: ICASSP, pp. 5113–5116 (2012)
11. Hearst, M.A.: Texttiling: segmenting text into multi-paragraph subtopic passages. Comput. Linguist. 23(1), 33–64 (1997)
12. Lecorvé, G., Gravier, G., Sébillot, P.: An unsupervised web-based topic language model adaptation method. In: ICASSP, pp. 5081–5084 (2008)
13. Malioutov, I., Barzilay, R.: Minimum cut model for spoken lecture segmentation. In: International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics, pp. 25–32 (2006)
14. Rosenberg, A., Hirschberg, J.: Story segmentation of brodcast news in english, mandarin and arabic. In: Proceedings of the Human Language Technology Conference of the NAAC, pp. 125–128 (2006)
15. Sitbon, L., Bellot, P.: Topic segmentation using weighted lexical links (wll). In: SIGIR, pp. 737–738 (2007)
16. Tür, G., Hakkani-Tür, D.Z., Stolcke, A., Shriberg, E.: Integrating prosodic and lexical cues for automatic topic segmentation. Computational Linguistics 27(1), 31–57 (2001)

17. Tür, G., Stolcke, A., Voss, L.L., Peters, S., Hakkani-Tür, D., Dowding, J., Favre, B., Fernández, R., Frampton, M., Frandsen, M.W., Frederickson, C., Graciarena, M., Kintzing, D., Leveque, K., Mason, S., Niekrasz, J., Purver, M., Riedhammer, K., Shriberg, E., Tien, J., Vergyri, D., Yang, F.: The calo meeting assistant system. IEEE Transactions on Audio, Speech and Language Processing 18(6), 1601–1611 (2010)
18. Utiyama, M., Isahara, H.: A statistical model for domain-independent text segmentation. In: Proceedings of the 39th Annual Meeting on Association for Computational Linguistics, pp. 499–506 (2001)
19. Winston, H., Hsu, H.-M., Chang, S.-F.: A statistical framework for fusing mid-level perceptual features in news story segmentation. In: International Conference on Multimedia and Expo, pp. 413–416 (2003)
20. Xiaoxuan, W., Lei, X., Mimi, L., Bin, M., Chng, E.S., Haizhou, L.: Broadcast news story segmentation using conditional random fields and multimodal features. IEICE Transactions on Information and Systems 95(5), 1206–1215 (2012)
21. Xie, L., Yang, Y., Liu, Z.Q., Feng, W., Liu, Z.: Integrating acoustic and lexical features in topic segmentation of chinese broadcast news using maximum entropy approach. In: International Conference on Audio, Language and Image Processing, pp. 407–413 (2010)
22. Xie, L., Yang, Y., Zeng, J.: Subword lexical chaining for automatic story segmentation in chinese broadcast news. In: Huang, Y.-M.R., Xu, C., Cheng, K.-S., Yang, J.-F.K., Swamy, M.N.S., Li, S., Ding, J.-W. (eds.) PCM 2008. LNCS, vol. 5353, pp. 248–258. Springer, Heidelberg (2008)

An Investigation of Single-Pass ASR System Combination for Spoken Language Understanding

Fethi Bougares, Mickael Rouvier, Nathalie Camelin, Paul Deléglise, and Yannick Estève[⋆]

LIUM - University of Le Mans, France

Abstract. This paper studies the benefits provided by a single-pass Automatic Speech Recognition (ASR) exchange-based combination approach for spoken dialog system. Three famous open-source ASR systems are used to experiment this approach in the framework of Spoken Language Understanding (SLU). On the ASR side, single-pass ASR systems are used with an online acoustic model adaptation using the previous utterances said by a speaker. On the SLU side, a competitive CRF-based SLU system is applied on outputs of ASR system to obtain the semantic concepts. The evaluation is done on the French PORT-MEDIA test data in terms of both Word Error Rate (WER) and Concept Error Rate (CER). While the best single pass system used alone shows a CER of 29.8% for a WER of 22.8%, single-pass ASR exchange-based combination reaches a CER of 27.3% for a WER of 26%. This CER is only slightly higher than the one reached by a 5-passes ASR system which obtained a CER of 26.8% for a WER of 22.8% in better conditions, *i.e.* better acoustic model adaptation made on all the speech utterances said by a speaker, advanced feature extraction techniques and search graph rescoring using language model with higher order.

Keywords: Automatic speech recognition, spoken dialog understanding, ASR system combination.

1 Introduction

Automatic Speech Recognition (ASR) systems are usually based on a multi-passes framework. There is a consensus on the fact that a multi-passes architecture enables to recognize speech with a higher degree of accuracy than a single-pass architecture. However, these multi-passes ASR systems require an iterative decoding scheme which is time consuming and cannot be appropriate for applications such as human-machine spoken dialog systems.

[⋆] This research was supported by the ANR (Agence Nationale de la Recherche) under contracts number ANR-09-BLAN-0161 (ASH) and ANR-08-CORD-026 (PORT-MEDIA).

In a previous work [2], we proposed a low latency ASR combination of parallelized single-pass systems based on a local ROVER and a driven decoding algorithm using bags of n-grams. These bags of n-grams were extracted from recognition hypotheses provided by auxiliary ASR systems and were used by a primary ASR system to alter its linguistic scores. The major goal of such parallel combination was to couple the advantages (recognition accuracy and system robustness) provided by system combination techniques with the speed of a single pass system.

ASR combination was deeply studied these past 15 years, as summarized in the overview of ASR combination presented in [17]. The most frequent combination architecture is ROVER, introduced in [6], but other kinds of combination, based on cross adaptation of acoustic models [19] or language models [16], are also popular. Recently, in [11], a driven decoding algorithm was proposed to take into account automatic transcriptions of different ASR systems to modify in-the-fly n-gram scores of a ASR system. This permitted to significantly reduce the word error rate and inspired the ASR exchange-based combination approach presented in [2].

In this paper, we present a study which evaluates the benefits provided by a single-pass ASR exchange-based combination approach for spoken dialog system. Such application implies a fast system response while multi-passes ASR system do not seems as well appropriate as a single-pass ASR in terms of speed. It is known that a single-pass ASR system reaches significantly lower performances than an multi-passes one in terms of accuracy. Three famous open source ASR systems are used to experiment this approach in the framework of the French PORT-MEDIA data. Section 2 discusses about the applicative context of study, *i.e.* Spoken Language Understanding in the framework of the PORT-MEDIA project presented in section 4.1. Section 3 describes an exchange-based decoding paradigm, called BONG-EBD, for Bag Of N-Grams Exchange-Based Decoding. Before concluding, section 4 presents the experiments with the comparison of results obtained by the three state-of-the-art ASR systems: CMU Sphinx project [21], RWTH ASR toolkit [22], Kaldi ASR system [20] and their combination.

2 Spoken Language Understanding

In most of today's commercial applications based on speech recognition, the quality of the human-machine interaction is still far from being enjoyable and effective. To improve the usefulness and acceptability of automatic dialog systems, a good mean is to increase the level of intelligence of automatic systems up to SLU, but also to accelerate the system response.

SLU is the interpretation of words automatically transcribed from a speech signal: it aims to obtain a conceptual representation of natural language sentences. The ASR component has to be precise enough to facilitate the semantic interpretation process, and has to be fast in order to contribute to the speed of system response. Currently, SLU consists often in a slot sequence tagging for spoken language, and such SLU module are usually based on Conditional Random Fields (CRF) [4].

3 Exchange-Based ASR Combination

Exchange-based ASR combination is based on the Driven Decoding Algorithm (DDA) [13] which was first introduced to help LVCSR systems to process audio documents associated to imperfect manual transcripts (*i.e.* subtitles). Although they are inexact, these transcriptions represent useful information exploited to drive the search space pruning and allow system accuracy improvement. By using DDA, a DTW (dynamic time warping) alignment is performed between the system partial hypothesis and the imperfect transcription. This alignment is used to compute a matching score α which is integrated in the computation of the linguistic score L:

$$L(w_i|w_{i-2}, w_{i-1}) = P(w_i|w_{i-2}, w_{i-1})^{1-\alpha(w_i)}$$

where $P(w_i|w_{i-2}, w_{i-1})$ is the probability provided by the initial trigram model for the word sequence $w_{i-2}w_{i-1}w_i$ and $\alpha(w_i)$ is the estimated matching score. The matching score depends on the similarity between the system hypothesis w_i and the imperfect transcription. The linguistic score L replaces the n-gram probability as the linguistic part of the usual formula used in speech recognition. By switching imperfect manual transcriptions by ASR outputs coming from other systems (called auxiliary systems), DDA can be used to combine ASR systems, as presented in [12]: when ASR decoders are used in parallel, we call this approach an exchange-based ASR combination.

Improvements were recently introduced in [1]: for instance, to speedup the combination process, auxiliary transcriptions are presented as a bag-of-n-grams (BONG), and DTW alignment is replaced by a simple search in the corresponding bag. Using these modifications, the driven decoding is straightforward efficiently generalizable: integration of a new auxiliary system is simply done by adding its recognition hypotheses to the corresponding bag. For a small number of auxiliary systems (*e.g.* three), BONG-EBD combination outperforms ROVER [6], even if the primary system is guided by less accurate auxiliary systems. In this study, the BONG driven decoding is integrated in the primary system. Details to compute the BONG matching score are described in [1].

4 Experiments

Experiments were made to evaluate the impact of an ASR exchange-based combination approach for spoken dialog system. ASR systems are evaluated on the French PORT-MEDIA corpus in terms of Word Error Rate (WER) and Concept Error Rate (CER) reached by applying a competitive CRF-based SLU system on outputs of recognition systems.

4.1 PORT-MEDIA Corpus

PORT-MEDIA is a French corpus related to the domain of ticket reservation within the 2010 Festival d'Avignon [14]. 700 dialogues were recorded with a *wizard of Oz* technique (a human agent mimics an automatic system): these dialogues were conducted by telephone. The dataset contains 10k user utterances and 65,775 words, for a total of 1,920 distinct words. The corpus has been manually transcribed and semantically annotated. The semantic annotation uses 35 concepts (*e.g.* theatreName, nbTicket, answer, ...). Each concept is supported by a sequence of words, the concept support. The *null* concept is used to annotate every word segment that does not support any of the 34 other concepts. On average, a concept support contains 2.6 words and 2.5 concepts are included in a utterance. This corpus has been divided into three sub-corpora : the *TRAIN* corpus is made up of 500 dialogues, the *DEV* is made up of 100 and the last 200 dialogues constitute the *TEST*.

4.2 ASR Systems

All the ASR systems used in the experiments share the same dictionary (about 5K words) and the same language model trained on the PORT-MEDIA training corpus and some articles from "Le Monde" newspaper. In the following, some precisions are presented about ASR singularities, especially at the acoustic level. The three single-pass ASR systems are presented, in addition to a brief description of the competitive 5-passes system used as contrastive.

Sphinx. In these experiments, the single-pass Sphinx ASR system is used with a CMLLR [15] adaptation of acoustic models. For each user utterance, the CMLLR transformation matrix is computed on the automatic transcriptions of the previous user utterances in the same dialog: we call this *on-line* adaptation. This ASR system is trained using 39 dimensions acoustic features consisting of 13 static MFCCs and their first- and second-order derivatives. Since PORT-MEDIA task is phone calls for human-machine interaction, these features are computed corresponding to narrowband analysis. Acoustic models are trained using a set of data from distinct sources: ESTER-1, ESTER-2, EPAC [5] and PORT-MEDIA training data. Acoustic SAT-CMLLR models are composed of 7500 tied states and 28 gaussians for each state. These models are then adapted to speaker gender (male/female) using MAP [8] adaptation of means, covariances and weights.

Kaldi. Kaldi is an open-source toolkit based on Finite State Transducers (FST) framework. Experiments are carried out using version 1.0 of Kaldi toolkit [20]. Kaldi acoustic models are trained using all ESTER-1, ESTER-2, EPAC and PORT-MEDIA corpora. Acoustic features are based on mel-frequency cepstral coefficients, 13 MFCC-features coefficients are first extracted and then expanded with delta and double delta features. Acoustic models are composed of 5000 context-dependent states and 210.000 Gaussians. The states tying is performed using a decision tree based on automatically phonetic question. In addition, *on-line* fMLLR linear transformation acoustic adaptation is performed.

RASR. The RWTH ASR (short RASR) [22] system is an open-source speech recognition toolkit based on a Beam search decoding algorithm developed by the RWTH Aachen University. Acoustic models are trained using the same corpus used for Kaldi system. Acoustic models were trained using MFCC coefficients: first 16 MFCC static coefficients are extracted. A sliding window of size 9 frames is applied to previous features and 144-dimensional vectors were obtained. These feature vectors were then projected down to 45 components using LDA transformation. Acoustic models are composed of 6500 context-dependent states and 703.980 gaussians. Unlike Kaldi systems, the states tying is performed using a decision tree based on manual phonetic questions. The RASR *on-line* acoustic adaptation is performed using CMLLR and MLLR [7] linear transformations.

LASR. LASR, the LIUM ASR system, is an expansion of the best open-source ASR system participating in the ESTER 2 evaluation campaign [3]. The LASR system is based on a multi-passes decoding scheme using two types of acoustic features. The first set is composed by 39 dimensional PLP features (13 PLP with energy, delta, and double-delta). The second type is composed by probabilistic features produced by a Multi Layer Perceptron (MLP), trained using the ICSI QuickNet libraries [23]. The input speech representation of our MLP is a concatenation of nine frames of 39 MFCC coefficients (twelve MFCC features, energies, Δ and $\Delta\Delta$). The topology of the MLP is the following: the first hidden layer is composed of 4000 neurons, the second one, used as the decoding output, of 40 neurons and the third one, used for training, of 102 neurons (34 phonemes, 3 states per phoneme). The MLP features were decorrelated by a PCA transformation which allows an additional dimensionality reduction. The second feature vector has 79 parameters resulting from the concatenation of the MLP and PLP (39 PLP + 40 MLP).

Acoustic models for 33 phonemes and 5 kinds of fillers are trained using a set of data from distinct sources. The training corpus is composed of 511 hours of wide band and 60 hours of narrow band training data.

The decoding strategy is close to that used in LIUM'08 system [3]. The involved passes are as follows:

1. The first pass uses gender acoustic models and a 3-gram language model. Only PLP features are used.
2. The word-graphs provided by the first pass are used to compute a CMLLR transformation for each speaker on all the user utterances (*off-line* adaptation). This second pass is performed using SAT and Minimum Phone Error (MPE) acoustic models with CMLLR transformations. Only PLP features are used.
3. In the third pass, the word-graphs of previous pass are used to drive a graph-decoding with full 3-phone context with a better acoustic precision, particularly in inter-word areas. This pass generates new word-graphs. PLP features are used in association with MLP.

4. The linguistic scores of the third pass word-graphs are updated using a 4-gram language model.
5. The last pass generates a confusion network from the word-graphs and applies the consensus method to extract the final one-best hypothesis

4.3 SLU System

The PORT-MEDIA corpus is close to the MEDIA corpus [18]: they follow the same paradigms and specifications and differ only on the domain application. Therefore we developed a statistical semantic annotation system based on a Conditional Random Field (CRF) tagger [10]. Actually, this method has shown the best semantic annotation performance on the MEDIA corpus [9], on manual and automatic transcriptions. We apply the CRF++ toolkit on the *TRAIN* corpus to obtain a standard semantic model. Each word is represented on two levels: the word itself and its pre-defined semantic categories[1] (*e.g.*: TOWN for avignon, SURNAME for anne, ANIMAL for cat, ...). The two previous words and the two following ones are taken into account in a unigram or bigram to take the semantic label decision on the word.

In the same way of the WER computation, the CER is computed between the sequence of reference concepts (one per concept support) and the sequence of hypothesis concepts (in both case the *null* concept is discarded). As a result, we obtain a CER of 17.8% on the reference transcription of the *TEST* set. A comparative result (CER equals to 18.9%) had been obtained on PORT-MEDIA in [14].

4.4 Results

Table 1 presents the results obtained by each single-pass ASR system on the development and the test corpora in terms of word error rate. While the Sphinx decoder reached the best result on the development corpus with a WER of 28.1%, the Kaldi decoder reaches the best WER on the test data with 29.5% of WER.

Table 1. WER for each single-pass ASR system applied to the PORT-MEDIA data sets. Each ASR system uses an online acoustic model adaptation.

	single-pass ASR systems		
data	Sphinx	Kaldi	RASR
dev	**28.1%**	28.9%	30.4%
test	31.5%	**29.5%**	33.0%

In table 2, the performances reached by the SLU system applied to the recognition outputs from each single-pass ASR system is presented. The best results are reached on the Kaldi outputs with a CER of 30.5% on the development corpus, and 29.8% on the test corpus.

[1] The list of the different semantic categories has been built by Christian Raymond on the MEDIA Corpus.

Table 2. Concept Error Rate reached by applying a CRF-based SLU system on recognition outputs from the single-pass ASR systems

		Automatic transcripts		
data	Manual transcripts	Sphinx	Kaldi	RASR
dev	18.7%	31.3%	**30.5%**	35.2%
test	17.8%	31.4%	**29.8%**	34.2%

WER obtained by different combinations of single-pass ASR system are presented in table 3.

Table 3. Word Error Rate for each single-pass ASR system combination applied to the PORT-MEDIA data sets. WER of the 5-passes LASR system are also presented.

	Single-pass ASR systems			5-passes
data	ROVER	BONG	BONG+ROVER	LASR
dev	**23.9**%	25.7%	**23.9**%	*21.5%*
test	26.2%	29.8%	**26.0**%	*22.8%*

The ROVER combination includes the 3 single-pass decoders. The BONG-EBD approach is performed by using the Sphinx system as the primary ASR system. The primary system deals with the recognition hypotheses provided by Kaldi and RASR considered as auxiliary systems, and uses them to modify some n-gram scores. The ROVER+BONG combination consists in replacing in the ROVER combination the outputs produced by the Sphinx decoder used alone by the outputs produced by the Sphinx decoder used as a primary ASR system in the BONG-EBD framework. On the development and the test corpora, ROVER and ROVER+BONG single-pass ASR combinations have a WER greater than the WER obtained by the 5-passes LASR system: the WER of 5-passes system is 3.2 point lower than the ROVER+BONG on the test data. But table 4 shows that this difference is almost insignificant in terms of CER: the ROVER+BONG single-pass ASR combination allows to reach a CER of 27.3% while the 5-passes ASR system allows a CER of 26.9%.

Table 4. Concept Error Rate for each single-pass ASR system combination applied to the PORT-MEDIA data sets. CER for the 5-pass LASR system are also presented.

	Single-pass ASR systems			5-pass
data	ROVER	BONG	BONG+ROVER	LASR
dev	**28.3**%	30.1%	**28.3**%	*27.1%*
test	27.6%	30.7%	**27.3**%	*26.9%*

5 Conclusion

This study has evaluated the benefits provided by an ASR exchange-based combination approach for spoken dialog system. Three open source ASR systems were used to experiment this approach in the framework of SLU: on the French PORT-MEDIA test data, while the best single pass system used alone reaches a CER of 29.8%, single-pass ASR exchange-based combination permits to reach a CER of 27.3%. This CER is only slightly higher than the one reached by a 5-passes ASR system which obtained a CER of 26.8% in better conditions: more training data for acoustic models, a longer computing time and static acoustic model adaptation made on all the speech utterances said by a speaker.

It is shown that combination of parallelized single-pass ASR systems seems to be a competitive solution in the framework of SLU to accelerate the system response in comparison to the use of a multi-pass ASR system. More, such combination of single-pass systems significantly reduces the word error rate in comparison to the best single-pass ASR system used alone. A such solution is particularly relevant with the recent hardware evolution: multi-cores and multi-CPUs can facilitate and accelerate the parallelization of single-pass ASR systems. Last, this paper presents the first experiments on automatic speech processing applied to the PORT-MEDIA data, which will be soon easily available by contacting ELRA[2].

Acknowledgements. This research was supported by the ANR (Agence Nationale de la Recherche) under contracts number ANR-09-BLAN-0161 (ASH) and ANR-08-CORD-026 (PORT-MEDIA).

References

1. Bougares, F., Estève, Y., Deléglise, P., Linarès, G.: Bag Of N-Gram driven decoding for LVCSR system harnessing. In: IEEE Automatic Speech Recognition and Understanding Workshop, Hawaii, USA (December 2011)
2. Bougares, F., Rouvier, M., Estève, Y., Linarès, G.: Low latency combination of parallelized single-pass LVCSR systems. In: Interspeech, Portland, Oregon (USA), September 9-13 (2012)
3. Deléglise, P., Estève, Y., Meignier, S., Merlin, T.: Improvements to the LIUM French ASR system based on CMU Sphinx: what helps to significantly reduce the word error rate?. In: Interspeech, Brighton, UK (September 2009)
4. Deoras, A., Sarikaya, R., Tür, G., Hakkani-Tür, D.: Joint decoding for speech recognition and semantic tagging. In: Interspeech 2012, Portland, Oregon, USA (September 2012)
5. Estève, Y., Bazillon, T., Antoine, J.-Y., Béchet, F., Farinas, J.: The EPAC corpus: manual and automatic annotations of conversational speech in french broadcast news. In: LREC 2010, Malta, May 17-23 (2010)
6. Fiscus, J.: A post-processing system to yield reduced word error rates: recogniser output voting error reduction (ROVER). In: ASRU, pp. 347–354 (1997)
7. Gales, M.J.F.: Maximum likelihood linear transformations for hmm-based speech recognition. Computer Speech and Language 12, 75–98 (1998)

[2] `http://catalog.elra.info/`

8. Gauvain, J.-l., Chin-hui, L.: Maximum a posteriori estimation for multivariate gaussian mixture observations of markov chains. IEEE Transactions on Speech and Audio Processing 2, 291–298 (1994)
9. Hahn, S., Dinarelli, M., Raymond, C., Lefèvre, F., Lehnen, P., De Mori, R., Moschitti, A., Ney, H., Riccardi, G.: Comparing stochastic approaches to spoken language understanding in multiple languages. IEEE Transactions on Audio, Speech and Language Processing PP(99), 1 (2010)
10. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: International Conference on Machine Learning, Williamstown, MA, USA, pp. 282–289 (2001)
11. Lecouteux, B., Linarès, G., Estève, Y., Mauclair, J.: System combination by driven decoding. In: ICASSP (2007)
12. Lecouteux, B., Linarès, G., Estève, Y., Gravier, G.: Generalized driven decoding for speech recognition system combination. In: ICASSP, Las Vegas, Nevada, USA (2008)
13. Lecouteux, B., Linarès, G., Nocera, P., Bonastre, J.-F.: Imperfect transcript driven speech recognition. In: ICSLP /INTERSPEECH, Pittsburgh, Pennsylvania, USA (2006)
14. Lefèvre, F., Mostefa, D., Besacier, L., Estève, Y., Quignard, M., Camelin, N., Favre, B., Jabaian, B., Rojas-Barahona, L.M.: Leveraging study of robustness and portability of spoken language understanding systems across languages and domains: the Port-MEDIA corpora. In: LREC 2012, Istanbul, Turkey (2012)
15. Leggetter, C.J., Woodland, P.C.: Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models. Computer Speech & Language 9(2), 171–185 (1995)
16. Liu, X., Gales, M.J.F., Woodland, P.C.: Language Model Cross Adaptation For LVCSR System Combination. In: Interspeech (2010)
17. Ma, C., Kuo, H.-K.J., Soltau, H.: A comparative study on system combination schemes for LVCSR. In: ICASSP, pp. 4394–4397 (2010)
18. Maynard, H., Rosset, S., Ayache, C., Kuhn, A., Mostefa, D.: Semantic annotation of the MEDIA corpus for spoken dialog. In: Proceedings of Eurospeech, Lisbon, pp. 3457–3460 (2005)
19. Nguyen, L., Abdou, S., Afify, M., Makhoul, J., Matsoukas, S., Schwartz, R., Xiang, B., Lamel, L., Gauvain, J.-L., Adda, G., Schwenk, H.: The 2004 BBN/LIMSI 10xRT English Broadcast News Transcription System. In: 2004 Rich Transcriptions Workshop, Pallisades, NY (2004)
20. Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., Silovsky, J., Stemmer, G., Vesely, K.: The Kaldi speech recognition toolkit. In: IEEE 2011 Workshop on Automatic Speech Recognition and Understanding, Hilton Waikoloa Village, Big Island, Hawaii, US (December 2011)
21. Ravishankar, M., Singh, R., Raj, B., Stern, R.M.: The 1999 CMU 10x real time broadcast news transcription system. In: Proc. DARPA Workshop on Automatic Transcription of Broadcast News (2000)
22. Rybach, D., Hahn, S., Lehnen, P., Nolden, D., Sundermeyer, M., Tüske, Z., Wiesler, S., Schlüter, R., Ney, H.: RASR - the RWTH Aachen University open source speech recognition toolkit. In: IEEE Automatic Speech Recognition and Understanding Workshop, Hawaii, USA (December 2011)
23. Zhu, Q., Chen, B., Morgan, N., Stolcke, A.: On using mlp features in lvcsr. In: Proc. ICSLP, Jeju, Korea, pp. 921–924 (2004)

Predicting Part-of-Speech Tags and Morpho-Syntactic Relations Using Similarity-Based Technique

Samuel W.K. Chan and Mickey M.C. Chong

Dept. of Decision Sciences
The Chinese University of Hong Kong
{swkchan,mickey_chong}@cuhk.edu.hk

Abstract. This paper describes a similarity-based technique which produces a good estimate of part-of-speech tags and their morpho-syntactic relations of Chinese compound words before they are fed into a tagger. The technique relies on a set of features from Chinese morphemes as well as a set of collocation markers which provide hints on the syntactic categories of the compound words. The technique is trained with a compound words database with more than 53,500 disyllabic words. Experimental results show the tagger with the technique outperforms its counterpart.

Keywords: Part-of-speech tagging, Chinese morphemes, Chinese word structures, Machine learning.

1 Introduction

The importance of lexical semantic resources in all domains of natural language processing (NLP), ranging from word segmentation, shallow parsing, semantic role labeling to question answering and text classification, is well recognized. However, even large lexical databases, such as WordNet (Fellbaum, 1998), do not include all the words encountered in broad-coverage NLP applications. The quality of these resources depends certainly, to a large degree, on the considerable efforts of lexicographers, who must keep pace with both language evolution and knowledge development in all relevant domains. In Chinese, the situation is even more taxing, as each Chinese morpheme carries meaning, new words can be simply constructed by the concatenation of morphemes, and there is no delimiter between words. As a result, the number of out-of-vocabulary (OOV) words in Chinese is huge, and identifying their part-of-speech (POS) tags is one of the challenging tasks in Chinese NLP.

On the other hand, most Chinese words are compounds consisting of two or more morphemes. Most Chinese have a direct correspondence across syllable, character, and morpheme. That is, a Chinese character in print virtually always, with a few exceptional cases, represents one syllable which is most often a morpheme. The combinations of the morphemes in Chinese compound words are certainly not random, but exhibit several different morpho-syntactic relations. The relations include mainly the endocentric, coordinative, subject-predicate, verb-object, verb/adjective complement and others. Except some exocentric nominal compound words which are semantically opaque, such as 東西/dong1 xi5/(east west: *thing*), the canonical endocentric modifier-noun/verb relation and the coordinative relation are constructed from morphemes which contribute to word meaning. Packard (2000) describes the intimate relationship between Chinese words and their components, including how the identities of Chinese morphemes are word-driven. He argues that the morphological relations are not only important in understanding Chinese, but can be used to pinpoint the semantic head morpheme in Chinese. A semantic head is a part of the word which is a more general instance of what the entire word means, often defined in terms of the *is-a* relation. For example, in endocentric compounds, say 綠葉/lv4 ye4/(green leaf: *leaf*), the second morpheme葉/ye4/(*leaf*) is the semantic head that expresses the main meaning of the word. The role of the first morpheme is to modify the second one. Similarly, morphemes in the coordinative relation lean to have the same role in indicating the meaning of a given word. The component morphemes in coordinative compounds have the same lexical properties, and have either similar or opposite meanings. For example, the morphemes 跳 /tiao4/(*jump*), and 躍 /yue4/(*jump*) in the word 跳 躍 /tiao4 yue4/(jump jump: *leap*) share the same subject as well as the lexical properties.

In this research, we have designed and implemented a means of predicting the POS of Chinese compound words. While it is considered that the notion of the head is posited to be knowledge that is intrinsic to a native speaker's knowledge of words (Packard, 2000), another aspect of this research is to explore a means to have a good approximation of the morpho-syntactic relation *MSyR*(*W*) which is virtually the key to determine its head morpheme. A non-native speaker, who understands the right-headedness of nouns and their relevant morpho-syntactic relations, knows that綠葉 /lv4 ye4/(green leaf: *leaf*) is some kind of leaf, even though he/she never comes across the word. This awareness may be particularly important in Chinese for, at least, three reasons. *First*, the same head morpheme often appears in a large number of Chinese words. For example, the morpheme 葉/ye4/(*leaf*) forms many compound words including樹葉/shu4 ye4/ (tree leaf: *leaf*), and 嫩葉/nen4 rou4/(young leaf: *leaf bud*); *Second*, most Chinese compounds built hierarchically from other compounds, such as 綠葉樹/lv4 ye4 shu4/(green leaf tree: *tree with green leaves*). The morpho-syntactic relation of the compound provides a good hint on the sense estimation of the word; *Third*, a noun, that is not in endocentric relation, does not necessarily follow the Right-Headedness Principle for noun as advocated by Packard (2000: 194), such as in the word 根葉/gen1 ye4/(root leaf: *in all aspects*). The morpho-syntactic relation in a word governs the compositionality of the morphemes and imposes constraints in its possible meaning. While the semantic category of a compound word could be revealed by the head morpheme which, in turn, relies on its morpho-syntactic relation, it is imperative to explore an objective means to have a good approximation of their morpho-syntactic relation. In this research, all the predictions are based on two important types of features: *morpheme properties*, indicating the major structure of the word, and the *word neighbors in raw text*. The paper is organized as follows. In Section 2, we first provide a review of the related work. We then describe, in Section 3, a technique in predicting the POS tag of a word and its morpho-syntactic relation. The technique relies on a supervised ensemble machine learning technique which makes use of a set of linguistic features for the predictions. It shortlists the potential POS tags and morpho-syntactic relations by imposing necessary, even not sufficient, constraints on their features. In order to demonstrate the capability of the technique, a POS tagger is devised to tag a test corpus used in an open evaluation. The detailed results are given in Section 4, followed by a conclusion.

2 Related Work

One of the major strategies in tagging out-of-vocabulary (OOV) words is based on Frege's Principle of Compositionality, which states the sense of a complex can be compounded out of the senses of the constituents (Frege, 1948). The meaning of a complex, such as an OOV word, can be identified by combining or concatenating the meanings of the morphemes that make up the word. The study on the impact of Chinese morphemes on OOV words is throughout the literature. Chen & Bai (1998) study the distribution of OOV in the Sinica corpus, and find that there are 14 different methods of morpheme concatenation in Chinese. Other than proper nouns, composite and derived words constitute the majority of OOV words in the Chinese corpus. Inspired by the information content for similarity measure (Resnik, 1998; Lin, 1998), Chen & Chen (2000) describe a similarity-based model of learning morphological rules for Chinese compound nouns. Their model uses entropy to compute the similarity measure between an OOV word and the words in *Cilin* (Mei *et al.*, 1984). Tseng & Chen (2002) make use of the *k*-NN classifier to devise a morphological analyzer that can segment a word into a sequence of morphemes. The analyzer can also predict the morph-syntactic relationships between morphemes, such as modifier-head, verb-object, and resultative verb, and is based on the assumption that the morpho-syntactic relation of an OOV word reflects its sense. Kwong & Tsou (2003) suggest POS tagging should not only be theoretically valid but also sufficiently capture the extent of categorical fluidity as reflected by the corpus. Ng & Low (2004) suggest that character-based approach is better than a word-based approach for POS tagging in Chinese, simply because Chinese characters have well defined meanings. They also suggest all-at-once approach, that is, available information should be integrated into a unified framework to make the prediction. Gao *et al.* (2006) use a support vector machine (SVM) to estimate the likelihood that two adjacent characters will form a new word. Four linguistically motivated features are found to be indicative of word formation, namely, independent word probability, anti-word pairs, word formation analogy, and morphological productivity. Although they do not target POS tagging, they find that their SVM classifier fits well with 1+1[1] and 2+1 OOV identification. Chung & Chen (2010) analyze the morpho-syntactic behaviors of about 4,025 morphemes and classify them into 4 semantic types. They also propose constraint-based resolutions and a set of composition rules to predict the POS tags. Inspired by the work above, in this paper, we take one step further to propose and implement a mechanism to predict the POS tag and the morpho-syntactic relation of the OOV words based on the distributional similarity, which is based on a distributional hypothesis that words that occur within similar neighbors are semantically similar (Harris, 1968; Dagan *et al.*, 1999; Pereira, *et al.*, 1993). This hypothesis has already given rise to a large body of work on automatic thesaurus generation (Widdows, 2003; Curran & Moens, 2002;

[1] 1 = monosyllabic word 2 = disyllabic word.

Lin *et al.*, 2003), named-entity recognition (Ciaramita & Johnson, 2003), lexical entailment (Geffet & Dagan, 2005), and co-occurrence retrieval (Weeds & Weir, 2006). More specifically, researchers have attempted to assign OOV words automatically into WordNet using a corpus-based approach (Schütze, 1992; Curran, 2005). We have devised a set of collocation markers which uncover the co-occurrence between the word and some key linguistic clues in a corpus under the distributional hypothesis. Words that occur within similar contexts are syntactically similar. The detailed discussion of our approach is as follows.

3 Similarity-Based POS Tagger

The basic idea of the tagger in predicting the POS tags and the morpho-syntactic relations (*MSyR*) is based on two important types of features: *morpheme properties* and their *word neighbors in raw text*. In the POS tagging, the system architecture of the tagger is shown in Figure 1. Whenever there is an OOV word in the input sentence, the word is first subject to a similarity-based technique to unveil its potential POS tags before it is further processed in a base POS tagger. The fundamental rationale of our similarity-based technique is that whenever we have zero evidence for a higher-order, we "back off" to a lower-order. We approximate the POS of the OOV words by their constituent morphemes information, without becoming trapped in a subjective linguistic quagmire.

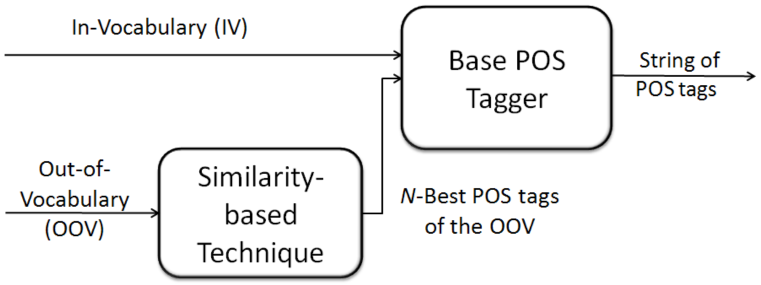


Fig. 1. Architecture of the POS tagger

To build a similarity-based technique for OOV words in the POS tagger, the following questions arise naturally: (i) Given a word *W* and a large corpus of raw Chinese text, is it possible to have a good estimation of the *POS*(*W*)? (ii) What features are important to determine its *POS*(*W*)? (iii) Do the features, that give a good prediction of *POS*(*W*), produce a good or even better estimation on its morpho-syntactic relation (*MSyR*)? To answer the above questions, for each disyllabic word, the following feature templates, shown in Table 1, are extracted for the training in a supervised machine learning algorithm in our similarity-based module. In the feature templates, *M*, *R*, and *MPOS* refer to the morpheme, the radical of the morpheme, and its morpheme POS respectively. We denote n =0 for the left morpheme, 1 for the right one.

Table 1. Categories of features used in the technique

	Feature Template
(a)	Morpheme features, M_n ($n = 0, 1$)
(b)	Radical features, R_n ($n = 0, 1$)
(c)	POS of the morphemes, $MPOS_n$ ($n = 0, 1$)
(d)	Phonetic components
(e)	Collocation marker features

For example, given a disyllabic word *W* in the database, say綠葉/lv4 ye4/(green leaf: *leaf*), templates (a)–(c) result in the following features, M_0=綠/lv4/(*green*), M_1=葉/ye4/(*leaf*), R_0=糸/mi4/(*silk*), R_1=艸/cao3/(*grass*), $MPOS_0$=Ag, $MPOS_1$=Ng where Ag, Ng represent an adjective and noun morpheme in the tagging convention of a lexicon respectively. We further decompose the pinyin of a Chinese morpheme into several phonetic components which involve consonants, vowels and tones. For example, the components for the morpheme綠/lv4/(*green*) are l, v (ü), 4. In addition, collocation marker features are used to uncover the co-occurrence between the disyllabic word *W* and some key linguistic clues in a corpus under the distributional hypothesis. Words that occur within similar contexts are syntactically similar.

Table 2. Sample of collocation markers

Major Category	Example of Collocation Markers
Function words	(把 ba3|被bei4|得de2)*W*
Tense	(已經yi3 jing1|現已xian4 yi3)*W*
Comparable	(很hen3|極ji2|較jue2) *W*
Negative adverb	L+"不 bu4"+ LR, where *W* = L+R
Complement	"又 you4"+ L+"又you4"+R, where *W* = L+R
"shi" phrases	L+"是shi"+R, where *W* = L+R
"di4" phrases	*W*+"地di4"

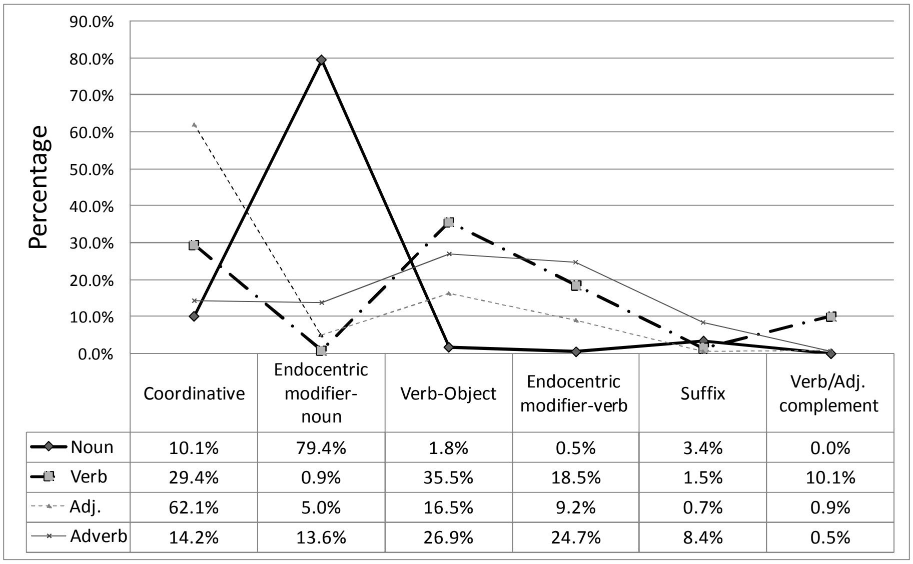
A Chinese corpus with more than 2 billion characters is first segmented by a segmenter[2] with an *F*-score of 98.5%. Similar to the work in Chinese Word Sketch developed in Academia Sinica (2006), these collocation markers reveal the syntactic context of the words and provide some hints on their part-of-speech. For example, the word *W* which follows the word 已經 /yi3 jing1/(already by: *have been*) is most likely to be a verb. Similarly, the word which is right after the word 的/de/(*of*) has a good chance to be a noun. More than 100 collocation marker features are designed and then deployed to extract from the corpus. Table 2 shows some of the markers and their major categories. All these extracted features are then presented to a supervised ensemble machine learning technique. The basic idea of ensemble techniques involves

[2] `http://HanMosaic.baf.cuhk.edu.hk`

considering several classification methods or multiple outputs to reach a decision. An ensemble of classifiers is a set of classifiers whose individual decisions are combined in some way, typically by weighted or un-weighted voting to classify new examples. They tend to yield better results than a single classifier in those situations when different classifiers have different error characteristics and their errors can compensate each other. Two questions need to be addressed when building and using an ensemble that integrates the predictions of several classifiers. *First*, what data are used to train the classifiers so that the errors made by one classifier could be remedied by the other? *Second*, how are the individual classifiers fused or integrated to produce a final ensemble prediction? We address the first question by introducing two heterogeneous and mutually independent attribute feature sets, namely the morpheme and its related features, i.e. (a)-(d) in Table 1 and the collocation marker features as discussed in Table 2. Instead of training all the features to form a single giant classifier, we produce two distinct, sometimes diversified, training sets of data to form two separate moderate classifiers, in the hope that they will produce a highly accurate prediction. The second question is addressed by employing the boosting algorithm. Boosting is an effective method that produces a very accurate prediction rule by combining rough and moderately inaccurate rules of thumb (Freund & Schapire, 1997). It generates the classifiers in an iterative way. At the early beginning, an initial base classifier using a set of training data with equal weight is first constructed. When the prediction of the base classifier differs from the expected outcome, the weight of the poorly predicted data is increased to an extent based on their misclassification rate on the preceding classifiers. As a result, the learning of the subsequent classifier will focus on learning the training data that are misclassified, or poorly predicted. This process continues until a specified number of iterations are reached or a predefined termination condition is met. As a result, the certainty factors, as discussed in Section 4, generated from all the classifiers will be fused to produce the final verdict for the prediction.

In the preparation of linguistic data as shown in Table 1, we first extend a Chinese compound words database which is developed at Peking University (PKU). The database originally contains more than 50,000 Chinese compound words with their word POS tags, morpheme POS tags, pinyin, as well as their morpho-syntactic relations (Liu *et al.*, 2000). It involves 21 different types of word POS tags and 23 types of morpheme POS tags. In addition, in our experiments, we target only at the disyllabic words in the database simply because of their pre-dominance. OOV words are usually generated in the process of disyllabification in Chinese. The further breakdown of the word POS tags for all 53,753 disyllabic words is shown in Table 3. In order to have a better overall picture of the Chinese compound words and of the variety of morpho-syntactic relations it presents, we study the distribution of six most popular morpho-syntactic relations found in four major POS tags, namely noun, verb, adjective and adverb. As shown in Table 3, about 80% of nouns are endocentric while verb-object and coordinative relations seem dominant in Chinese verbs. Among all 3,957 adjectives, more than 62% of them are coordinative compounds. This shows that the morpho-syntactic relations in each POS class are not random, at least demonstrated in more than 90% of our 53,753 disyllabic words. This provides further evidence that morphemes and their morpho-syntactic relations do hint on the part-of-speech of the Chinese words.

Table 3. Distribution, in percentage, of 53,753 disyllabic Chinese words in six major morpho-syntactic relations under four different POS tags



	Coordinative	Endocentric modifier-noun	Verb-Object	Endocentric modifier-verb	Suffix	Verb/Adj. complement
◆ Noun	10.1%	79.4%	1.8%	0.5%	3.4%	0.0%
■ Verb	29.4%	0.9%	35.5%	18.5%	1.5%	10.1%
▲ Adj.	62.1%	5.0%	16.5%	9.2%	0.7%	0.9%
× Adverb	14.2%	13.6%	26.9%	24.7%	8.4%	0.5%

4 Experiments and Results

We train and test the above feature templates in C5.0 decision tree which is a tree-based supervised machine learning model and plays an important role in many data analyses (Quinlan, 1993). A key advantage of the tree-based learning is its interpretability. The feature spaces partition is fully described by a single tree. Usually, if the training data exhibit regular patterns and are not random, classifiers that are in form of rulesets will be constructed after training. Otherwise, no classifiers will be generated. The rulesets consist of unordered collections of *if-then* rules, each associated with a value, called certainty factor *CF*, between 0 and 1. The factor indicates the confidence with which this prediction is made. During prediction, for the rulesets, a feature is used to classify a case if it is referenced by a condition of at least one rule that applies to that case. In addition, in this experiment, we apply the ensemble technique to enhance its predictive power. This technique creates a finite set of classifiers from random sets of training instances and then uses them together for the prediction. During the experiment, all the disyllabic words, except the ones which cannot be found in the segmented corpus, in the database are trained. Similar collocation markers features are bundled together. As a result, more than 60 different attributes, as described in Tables 1 & 2, are then subject to the training in the tree-based learning. Under the same attributes sets, we conduct two sets of experiments using two different targets, i.e., word POS tags, *POS*(*W*) and their morpho-syntactic relations, *MSyR*(*W*). There are more than 10 different morpho-syntactic relations in the experimental data. To evaluate the performance of our similarity-based technique, a 90/10 training/testing strategy is employed. Table 4 summarizes the experimental details.

Table 4. Summary of the experimental details

# of disyllabic words in the PKU database	53,753
# of disyllabic words found in the database, but not in the 2-billion characters segmented corpus	836
# of disyllabic words for training (90%)	47,626
# of disyllabic words for testing (10%)	5,291
Total # of attributes used	83
Total # of classifiers	10
Total # of cross-validation trials	10
Target 1	*POS*(*W*)
Target 2	*MSyR*(*W*)

At the same time, we winnow some attributes systematically in order to filter the useful attributes from the unhelpful ones. Table 5 demonstrates the outcomes of five experiments in which different combinations of features are winnowed.

Table 5. Training and testing of classifiers using different feature sets in five experiments

Feature Set	Exp01	Exp02	Exp03	Exp04	Exp05
Morpheme	√	√	√	√	√
Radical	√	√	√	√	X
POS (morpheme)	√	X	√	√	√
Phonetic Components	√	√	X	√	X
Collocation Markers	√	√	√	X	√
Target 1: *POS*(*W*)					
Training error (%)	10.5	Fail	5.9	11.3	6.5
Test error (%)	15.7	Fail	14.9	16.5	15.3
Target 2: *MSyR*(*W*)					
Training error (%)	7.1	Fail	1.2	9.3	7.5
Test error (%)	16.6	Fail	15.8	18.5	16.7

From experiment Exp02, it is clearly shown that the morpheme POS is a prominent feature set in predicting the word POS. Without the morpheme POS, the tree-based learning fails to produce any indicative classifier. That is, the training data exhibit no regular patterns and are apparently random. As a result, no classifiers will be generated and the learning terminates during the experiment. In other words, morpheme POS is an important attribute in training the classifiers both for *POS*(*W*) and *MSyR*(*W*). While the inclusion of the features from phonetic components produces some negative impacts, the devised collocation markers are significant. They improve the accuracy about 1.0% and 2.0% in learning *POS*(*W*) and *MSyR*(*W*) respectively, even though the two different classifiers rely on the collocation markers in different extent. For example, the collocation markers found helpful in learning the *POS*(*W*) include the tense markers, such as (已經yi3 jing1|現已xian4 yi3)*W*, "di4" phrases, such as

W+"地di4" as well as the comparable markers. Surprisingly, it is worthwhile to mention that the radical of the morphemes is also one of the important features in predicting the two targets. The learnt classifier shown in the experiment Exp03 is selected to implement our similarity-based technique. While it is unreasonable to assume the *POS* of the morphemes, we take a further approximation, if not expedient, by multiplying the conditional probability of the *POS s*, given the morpheme *m* at position *l*, to the certainty factor CF_i^j of the rules r_i generated from the classifiers c_j

$$CF_i^j = \begin{cases} CF_i^j \times \prod_{k=0,1} P(s_k | m_k, l_k) & \text{if } s_k \text{ is in } r_i \\ CF_i^j & \text{otherwise} \end{cases} \quad (1)$$

To evaluate the contribution of the similarity-based technique in our POS tagger, we test our tagger with the similarity-based technique using a corpus that is provided in a POS tagging track of an open evaluation CIPS-ParsEval-2009, held in Beijing. The corpus provides 389,170 and 92,687 words for the training and test data respectively. Our base tagger is based on the work described in Ng & Low (2004) as shown in Figure 2.

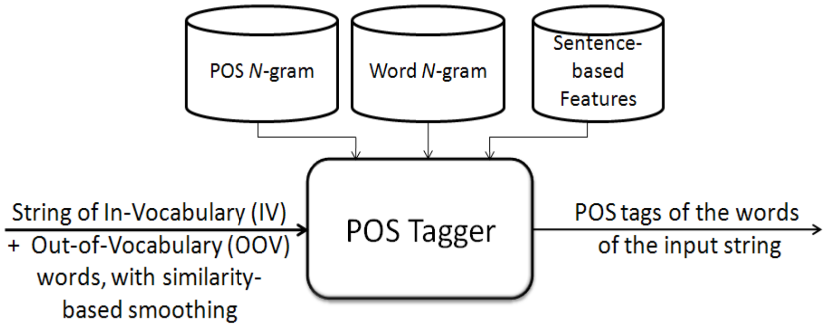

POS N-gram
Word N-gram
Sentence-based Features
String of In-Vocabulary (IV) + Out-of-Vocabulary (OOV) words, with similarity-based smoothing
POS Tagger
POS tags of the words of the input string

Fig. 2. Linguistic features used in our base POS tagger

While the accuracy of POS tagging may vary from parsers, corpus, or more importantly, the percentage of OOV in the test corpus, the overall tagging accuracy of 91.9% was reported in the Ng & Low tagger. In our base tagger, several additional features, such as the *N*-grams of the POS tags, word tags, and relative position of the target words in the sentences are used to enhance its performance. In particular, we introduce three pointwise mutual information (PMI) indicators to quantify the collocations of the POS tags next to the target word in our base tagger. These include PMI(POS2L, POS1L), PMI(POS1L, POS1R), and PMI(POS1R, POS2R), where POS2L stands for the POS tag of the second neighbor word suited at the left of the target word. Details can be found in Table 6.

Table 6. Additional features used in the base POS tagger. POS1R is the POS tag of the first neighbor word at the right of the target word.

POS2L	POS1L	Target Word	POS1R	POS2R		Additional features in the base POS tagger
√	√					PMI(POS2L, POS1L)
	√		√			PMI(POS1L, POS1R)
			√	√		PMI(POS1R, POS2R)

In addition, we adopt the performance of automatic POS tagging accuracy is equal to the ratio of sum of words with correct POS tags to sum of words in gold-standard sentences. Table 7 shows the performance of our tagger with and without the similarity-based technique. During the experiment, 3-best POS tags of the OOV, which are deduced by the technique, are fed into the base tagger as shown in Figure 1. In this research, we only implement the similarity-based technique for disyllabic compound words and leave trisyllabic words intact. At the same time, there is no special treatment for proper nouns. The accuracy of the POS tagging for OOV increases more than 24%, with a ripple effect of 1.1% increase for the IV. This raises the final accuracy to 95.3% while the state-of-the-art in the competition for the closed and open tracks are 93.41% and 93.40% respectively.

Table 7. Performance of our tagger with and (*without*) the similarity-based technique, where IV and OOV represent the in-vocabulary and out-of-vocabulary words respectively

	Total	IV	OOV
Correct	88,290 (*85,684*)	82,881 (*81,891*)	5,409 (*3,793*)
Incorrect	4,397 (*7,003*)	3,255 (*4,245*)	1,142 (*2,758*)
Total	92,687	86,136	6,551
Accuracy	95.26% (*92.44%*)	96.22% (*95.07%*)	82.57% (*57.90%*)

5 Conclusions

Unlike English, words in Chinese are not often associated with any apparent morphological features. As a result, porting an English POS tagger directly into a Chinese counterpart is usually ineffectual, if not impossible. Whereas previous research on Chinese OOV words mostly focuses on the identification of proper nouns, in this paper, the focus of interest is on Chinese compound words. We take advantage of the features from word structures as well as their neighbors in raw text, the linguistic hints from the collocation markers. We have devised a similarity-based technique to determine the part-of-speech into which an OOV word fits mostly and to study its impacts on the morpho-syntactic relations. While we have identified several major features that are helpful in the predictions, we apply the technique into a real POS tagger. Experiments show our technique renders a relatively good POS tagging accuracy. While the research on OOV tagging is still ongoing, one of the possible enhancements is to capture the senses of the morphemes into the technique. Focus should also be on a more detailed study on morpho-semantic constraints imposed by each morpheme.

Acknowledgement. The work described in this paper was partially supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK440609).

References

1. Chen, K.-J., Bai, M.-H.: Unknown word detection for Chinese by a corpus-based learning method. Computational Linguistics and Chinese Language Processing 3(1), 27–44 (1998)
2. Chen, K.-J., Chen, C.-J.: Automatic semantic classification for Chinese unknown compound nouns. In: COLING 2000, pp. 173–179 (2000)
3. Chinese Word Sketch (2006), `http://wordsketch.ling.sinica.edu.tw/`
4. Chung, Y.-S., Chen, K.-J.: Analysis of Chinese morphemes and its application to sense and part-of-speech prediction for Chinese compounds. In: Proceedings of the Joint Conference of 23rd International Conference on the Computer Processing of Oriental Languages (2010)
5. Ciaramita, M., Johnson, M.: Supersense tagging of unknown nouns in WordNet. In: Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing, pp. 168–175 (2003)
6. Curran, J.R., Moens, M.: Improvements in automatic thesaurus extraction. In: Proceedings of the ACL 2002 Workshop on Unsupervised Lexical Acquisition, Philadelphia, Pennsylvania, pp. 59–66 (2002)
7. Curran, J.R.: Supersense tagging of unknown nouns using semantic similarity. In: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics, Ann Arbor, pp. 26–33 (2005)
8. Dagan, I., Lee, L., Pereira, F.: Similarity-based models of word co-occurrence probabilities. Machine Learning Journal 34(1-3), 43–69 (1999)
9. Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press, Cambridge (1998)
10. Frege, G.: On sense and reference. The Philosophical Review 57, 207–230 (1948)
11. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences 55(1), 119–139 (1997)
12. Gao, J., Li, M., Wu, A., Huang, C.-N.: Chinese word segmentation and named entity recognition: A pragmatic approach. Computational Linguistics 31(4), 531–574 (2006)
13. Geffet, M., Dagan, I.: The distributional inclusion hypotheses and lexical entailment. In: Proceedings of the 43rd Annual Meeting of the ACL, pp. 107–114 (2005)
14. Harris, Z.: Mathematical Structures of Language. Wiley, NY (1968)
15. Kwong, O.Y., Tsou, B.K.: Categorical fluidity in Chinese and its implications for part-of-speech tagging. In: Proceedings of the Conference on European Chapter of the Association for Computational Linguistics, pp. 115–118 (2003)
16. Lin, D.: An information-theoretic definition of similarity. In: Proceedings of 15th International Conference on Machine Learning, pp. 296–304 (1998)
17. Lin, D., Zhou, S., Qin, L., Zhou, M.: Identifying synonyms among distributionally similar words. In: Proceedings of the 18th International Joint Conference on Artificial Intelligence, pp. 1492–1493 (2003)

18. Liu, Y., Yu, S., Zhu, X.: Construction of the contemporary Chinese compound words database and its application. In: Zhang, P. (ed.) The Contemporary Educational Techniques and Teaching Chinese as a Foreign Language, pp. 273–278. Guangxi Normal University Press (2000)
19. Mei, J., Zhu, Y., Gao, Y., Ying, H.: *Cilin* 《同 義 詞 詞 林》梅家駒等 商務印書館 (1984) (in Chinese)
20. Ng, H.T., Low, J.K.: Chinese part-of-speech tagging: One-at-a-time or all-at-once? Word-based or character-based? In: Proceedings of EMNLP, Barcelona, Spain (2004)
21. Packard, J.L.: The Morphology of Chinese: A Linguistic and Cognitive Approach. Cambridge University Press (2000)
22. Pereira, F., Tishby, N., Lee, L.: Distributional clustering of similar words. In: Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics, pp. 183–190 (1993)
23. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann (1993)
24. Resnik, P.: Semantic similarity in a taxonomy: An information-based measure and its application to problem of ambiguity in natural language. Journal of Artificial Intelligence Research 11, 95–130 (1998)
25. Schütze, H.: Automatic word sense discrimination. Computational Linguistics 24(1), 97–124 (1992)
26. Tseng, H., Chen, K.-J.: Design of Chinese morphological analyzer. In: Proceedings of the First SIGHAN Workshops on Chinese Language Processing (2002)
27. Weeds, J., Weir, D.: Co-occurrence retrieval: A flexible framework for lexical distributional similarity. Computational Linguistics 31(4), 439–475 (2006)
28. Widdows, D.: Unsupervised methods for developing taxonomies by combining syntactic and statistical information. In: Proceedings of the 2003 Conference of the North American Chapter of the Association For Computational Linguistics on Human Language Technology, Morristown, NJ, pp. 197–204 (2003)

Statistical Error Correction Methods for Domain-Specific ASR Systems

Horia Cucu[1], Andi Buzo[1], Laurent Besacier[2], and Corneliu Burileanu[1]

[1] University "Politehnica" of Bucharest, Romania
{horia.cucu,andi.buzo,corneliu.burileanu}@upb.ro
[2] LIG, University Joseph Fourier, Grenoble, France
laurent.besacier@imag.fr

Abstract. Whenever an ASR company promises to deliver error-proof transcripts to the end user, manual verification and correction of the raw ASR transcripts cannot be avoided. This manual post-editing process systematically generates new and correct domain-specific data which can be used to incrementally improve the original ASR system. This paper proposes a statistic, SMT-based ASR error correction method, which takes advantage of the past corrected ASR errors to automatically post-process its future transcripts. We show that the proposed method can bring more than 10% WER improvements using only 2000 user-corrected sentences.

Keywords: ASR, error correction, language modeling, statistical machine translation (SMT).

1 Introduction

There are many applications in which general, large vocabulary ASR (Automatic Speech Recognition) systems are the only choice, because a potentially unknown speaker can potentially speak about anything. In this kind of applications the ASR word error rate (WER) is generally around 15-20%. However, most ASR applications have a much more specific task (limited number of speakers, speaking domain, etc.) and, for these simpler scenarios, the customer asks for better performance. Several model adaptation methods have been proposed to increase the general ASR's performance for a specific task. These methods require task-specific acoustic data to adapt the ASR's acoustic model and task-specific textual data to adapt the ASR's language model. The drawback of these model adaptation methods is that they require access to the internals of the ASR system and therefore they cannot be used if the ASR system is purchased as a *black-box*.

Among the various ASR applications, there are cases in which the user can benefit from error-prone transcriptions (i.e. spoken document retrieval, on-line movie captioning, etc.), but there are also cases in which the transcription must be manually post-edited (corrected) to become useful (i.e. dictation, interviews/news transcription, etc.). In this second case, the ASR user will always post-edit the raw transcriptions to create error-proof transcripts (for the final reader), systematically generating new and correct domain-specific data.

This paper deals with the latter case, investigating the ways in which the user-generated data can be used to incrementally improve the ASR's output. Although this study also analyzes the scenario in which the ASR system itself can be improved (and shows that its language model can be adapted to obtain better transcriptions), our main focus is on the scenario in which *the ASR system is regarded as a black-box*. In this scenario, we show that the user-generated data can be used to train a statistical post-editor (SPE), which can be afterwards employed to correct the raw ASR transcriptions.

The remainder of the paper is organized in five sections. Section 2 presents the most relevant related works in ASR errors correction, section 3 describes our proposed correction methods and section 4 evaluates them. In section 5 we analyze and discuss the various types of corrections based on some particular examples and in section 6 we draw the conclusions of this work.

2 Related Work

ASR error correction methods can be broken down in two categories: a) ASR adaptation methods and b) transcripts post-editing methods. ASR adaptation methods have been proven to be very effective when the recognition task is strictly defined and when the user has access to the internals of the ASR system. In [1] the user corrected text is used to identify the most probable cause for the error: out-of-vocabulary (OOV) word, wrong pronunciation or language model (LM) probabilities. Based on the identified error, the ASR is automatically adjusted: OOVs are inserted in the lexicon, the probability of the corrected bigram or trigram is boosted, new word pronunciations are generated. In our former papers [2-3] we also showed that LM adaptation can be successfully used to adapt an ASR system to a specific domain, even if the textual data is only available in a different language.

In other scenarios the ASR system is purchased as a black-box and the user does not have any hooks to modify the ASR models. In that case the ASR error correction methods apply a post-editing block to correct the errors in the raw ASR transcripts. One of the first works on this subject [4] uses a fertility channel model to correct 1-to-1, 1-to-2 and 2-to-1 errors. The paper shows that, for a particular dataset, the post-editing correction can be combined with LM adaptation to obtain even better results (24% relative WER improvement). Jung, Jeong, and Lee [5] also apply this fertility channel model (using syllables instead of words) along with an improved LM – that incorporates statistical and other higher level linguistic knowledge. They obtain much better results (on a very domain-specific Korean speech database): 40% relative WER improvement using only 70 training utterances.

Another approach in ASR error correction uses statistical replacement rules extracted from a corpus of raw transcripts along with their manual corrections. Kaki, Sumita and Iida [6] report on using character co-occurrence rules and obtain an improvement of 8.5% relative WER, when 4300 utterances (from a Japanese travel specific speech database) were used for training. Brandow and Strzalkowski [7] propose a method based on word sequences replacement rules, but do not provide any evaluation figures. Mangu and Padmanabhan [8] use ASR confusion networks composed of several word confusion sets to define rules that specify when the second candidate in a confusion set should be preferred over the first one. This method's reported improvement is only 4.2% relative WER, when 4000 speech utterances (from the Switchboard database) are used for training. Sarma and Palmer [9] regard ASR error correction only from an information retrieval point of view. They compile co-occurrence statistics for each word in the vocabulary and finally use these statistics to identify possible ASR errors, but do not provide an ASR overall evaluation.

In this paper we first analyze the potential of (unsupervised and semi-supervised) LM adaptation to correct ASR errors and propose a *statistical post-editing error correction method* inspired from statistical machine translation (SMT). Our proposed method goes beyond the state-of-the art by employing *a new strategy for error detection and correction* which is based on SMT principles and tools (Moses SMT Toolkit [10]). The method resembles in some aspects with the works presented in [4] and [5] because the SMT system also incorporates a fertility model, but extends this idea by generalizing this type of model. Moreover, we are the first to use a regular SMT system for ASR transcription post-editing and in the end we describe how the SMT system's *phrase translation table* can be further used to deeply analyze the ASR errors. We evaluate the method on a Romanian domain-specific speech database (weather news) and obtain an improvement of 10.5% relative WER when 2000 utterances are used for training.

3 ASR Output Error Correction

Consider the scenario in which a *general ASR system* is used *daily* to transcribe domain specific speech, for example broadcast weather news. Due to the mismatch between the general ASR's language model and the recognition domain (weather news), the recognition accuracy will be relatively poor. The ideal case would be to have a domain-specific ASR system, but this is not always possible (due to the lack of domain-specific data). In our previous studies [2-3] we considered the same problem and showed that if we have domain-specific text data in a different language we can translate it to the target language and successfully use it to adapt the ASR system. Now the premises have changed: we have *domain-specific audio data* in the target language, but not *domain-specific text data* (the required media type is not available). Consequently, we propose employing a general ASR system to transform the audio data into text and then use it to incrementally improve the ASR's output.

3.1 The Unsupervised Scenario

In the fully unsupervised scenario the raw ASR transcriptions are directly used for LM adaptation. Apparently, the system would have no means of learning from its previous errors, because it does not know when it makes mistakes. However, adapting the language model with the raw transcriptions might be beneficial, because the LM probabilities for domain-specific words and word sequences will be boosted up.

3.2 The Semi-supervised Scenarios

In the scenario considered in this study (daily transcriptions of broadcasted weather news) the ASR user (i.e. a media company offering transcription services) cannot deliver the raw transcripts to the final user due to the fact that they contain many errors. This is an application where it is mandatory to have error-proof transcriptions; therefore the ASR output must be manually verified and post-edited (corrected) to become useful. In this case the ASR user will always post-edit the raw transcriptions, systematically generating new and useful domain-specific data. This user-generated data can be used in various ways to incrementally improve the ASR's output and the improvement would obviously return to the user as he will have to do less and less corrections over time.

If the user has the possibility to modify the internals of the ASR system, then a good choice would be to *adapt the language model* with the manually generated text. Doing so, the user will create a domain-specific ASR system, which will be much more adequate at recognizing domain-specific speech. This is the first semi-supervised scenario discussed in this paper.

However, in some cases, the ASR system is purchased as a *black-box* and the user can only attach an automatic post-editing correction block to the baseline transcription system. In this second scenario, we propose using a statistical post-editor (SPE) that is trained to correct the *systematic ASR errors*, just as Figure 1 illustrates.

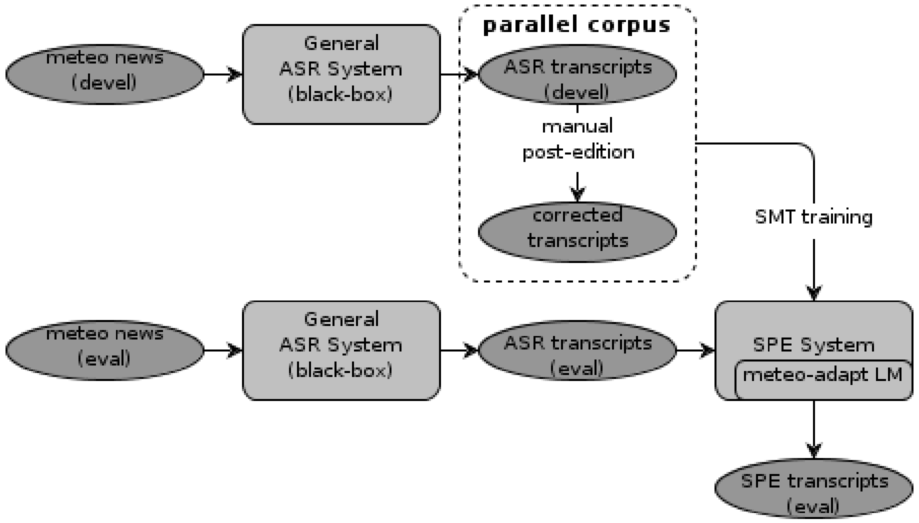


Fig. 1. Training and using the SPE system to correct ASR transcripts

In statistical machine translation (SMT), a parallel corpus (composed of sentences in the source language aligned with sentences in the target language) is used to train a translation model. After the training phase, the SMT system is able to translate source language sentences into the target language. The post-editing error correction method we propose in this paper uses an SMT system that regards the raw ASR transcripts as text in the source language and the manually corrected ASR transcripts as text in the target language. Consequently, this statistical post-editor is trained on a parallel corpus (composed of raw and corrected transcripts), thus learning how to *"translate" raw transcripts* *into corrected transcripts*. This error correction method is based on the observation that the post-editing task has quite a repetitive nature (the ASR usually makes systematical errors). The idea of using an SMT system to correct text was used before by Simard [11-12] and Lagarda [13], but only in the context where the raw text came from a rule-based SMT system and not from an ASR system (as in our scenario).

4 ASR Error Correction Experiments

4.1 Experimental Setup

For all the ASR experiments presented in this work we have used the same HMM-based acoustic model [2]. This system models the 36 phonemes in Romanian in a context-dependent manner with 4000 HMM senones and 16 Gaussian mixtures per senone state [14]. The acoustic model was previously created and optimized (using the CMU Sphinx Toolkit [15]) with a training database of about 54 hours of Romanian read speech. This speech database was progressively developed by our research group and now comprises isolated words, general newspaper articles and domain-specific (library) dialogues. The texts were recorded by 17 speakers (7 males and 10 females). The phonetic dictionary used in the experiments was created using a graphemes-to-phonemes conversion tool [14] and covers all the words in the LMs.

The general language model was previously created using a corpus of about 169M words collected from the Internet [2]. The domain specific language models were created using only the post-edited weather news transcripts. The domain adapted language models were created by interpolating the general language model with the domain specific language models. SRI-LM Toolkit [16] was used to create all the language models and to eventually interpolate them. The statistical machine translation system was trained using Moses SMT Toolkit [10]. The same toolkit was also used during system evaluation to automatically post-edit the raw transcriptions.

For the weather news experiments, we developed a new speech database by recording text news using three new speakers. One of these speakers recorded 2000 speech utterances to be further used for development (LM adaptation and SPE training) and all the speakers recorded 200 speech utterances each (a total of 600 utterances) to be further used for ASR evaluation. Although we admit that the evaluation database is quite small, we consider that the experimental results are conclusive and intend back them up with further experiments on a larger database.

4.2 Experimental Results

The general ASR system obtains a word error rate (WER) of 11.4% on the weather news evaluation database. This is considered to be the baseline which we want to improve by language model adaptation and statistical post-editing.

In the first experiment we consider the fully unsupervised scenario:

- the baseline ASR system is used to decode the 2000 utterances in the development database;
- the *raw transcripts* are used to adapt the general language model;
- the adapted ASR system is evaluated on the 600 utterances in the evaluation database.

Table 1 presents the adapted-ASR system results when 500, 1000, 1500 and respectively 2000 transcriptions are used for adaptation. The conclusion that emerges from this experiment is that even *unprocessed ASR data* can be successfully used to adapt the general ASR system. A relative improvement of 10.5% is significant and it is very encouraging given the relatively small amount of data used for adaptation (2000 raw sentences).

Table 1. Unsupervised LM adaptation

	# adaptation transcripts	**WER [%]**	**relative gain**
baseline ASR	0	11.4%	n/a
unsupervised adapted ASR	500	10.7%	6.1%
	1000	10.5%	7.9%
	1500	10.4%	8.8%
	2000	10.2%	10.5%

For the second experiment we exploit the errors made by the baseline system (on the development database) to create a statistical post-editing system, as follows:

- the baseline ASR system is used to decode the 2000 utterances in the development database;
- the raw transcripts are manually post-edited to create a set of corrected transcripts;
- the set of raw transcripts and the set of corrected transcripts are used to train a SPE system (as in Figure 1);
- the baseline ASR system + the additional SPE correction block is evaluated on the 600 utterances in the evaluation database (as in Figure 1).

Table 2 presents the results for this second experiment. The conclusion that emerges from this experiment is that we can obtain an important WER improvement even if we do not have access to the ASR internals, by attaching an automatic correction block as proposed above.

Table 2. Black-box ASR + SPE

	# corrected transcripts	**WER [%]**	**relative gain**
baseline ASR	0	11.4%	n/a
baseline ASR (black-box) + SPE block	500	10.7%	6.1%
	1000	10.4%	8.8%
	1500	10.2%	10.5%
	2000	10.2%	10.5%

In our third experiment we use the manually corrected transcripts to adapt the language model in the baseline ASR system. These transcripts do not contain any errors (as opposed to the unsupervised scenario) and consequently the adaptation is much more effective (see Table 3). The ASR improvement brought by this method is also more significant than the one obtained in the second experiment, but this error correction mechanism *can be applied only if the user has the means of changing the LM (the ASR is not a black-box).*

Table 3. Semi-supervised LM adaptation

	# adaptation transcripts	**WER [%]**	**relative gain**
baseline ASR	0	11.4%	n/a
semi-supervised adapted ASR	500	6.8%	40%
	1000	6.0%	47%
	1500	5.4%	53%
	2000	4.9%	57%

Finally, provided that we have access to the ASR internals, we combined the two semi-supervised methods presented above (LM adaptation and SPE correction). In this case, our experiments showed that the SPE block is left with almost nothing to correct and cannot bring any further improvements, but does not degrade the ASR performance either (same WER as in 3rd experiment).

5 Error Correction Analysis and Discussion

The ASR results obtained through unsupervised and semi-supervised LM adaption are quite easy to understand. *Unsupervised LM adaptation* boosts the probabilities of some domain-specific words and word sequences *which were correctly recognized* by the baseline ASR system. With increased LM probabilities, these items have a higher chance to be outputted in the future (and also in the ASR evaluation phase). This is in concordance with the reality: future weather news will also contain many weather terms and phrases.

Besides the above advantage, *semi-supervised LM adaptation* benefits from several other key features, deriving from the fact that the adaptation is done with correct ASR transcripts:

- all the words in the manually corrected transcripts get a LM probability boost,
- the wrongly recognized words and phrases do not get a LM probability boost,
- many OOV words for the baseline ASR can be detected in the development phase and recovered.

Our analysis showed that the baseline ASR system lacked 315 words among the ones *uttered in the development database* (315 OOVs). A few examples are: *climatologice* (climatologically), *burniță* (drizzle), *se înnorează* (it's getting cloudy), *tunete* (thunders), *lapovița* (sleet), *consistenți* (consistent), *aversele* (the showers), etc. The OOVs detected in the development transcripts can be automatically recovered: inserted in the adapted LM and in the ASR vocabulary. This improves the overall ASR system, because many of these words were also *uttered in the evaluation database*: the 600 evaluation utterances initially had 48 OOVs, among which almost 60% were recovered through the adaptation process.

The SPE correction block manages to improve the raw ASR transcription by replacing erroneous words and word sequences with their correct counterparts. We analyzed the replacements made by this SPE block and reached several interesting conclusions. First, some of the most frequent OOVs in the evaluation database were in part corrected (1-to-1 replacements):

- *masive muntoase* (mountains) → *masivele muntoase* (the mountains): 2 corrections out of 3,
- *averse* (showers) → *aversele* (the showers): 1 correction out of 3, etc.

In the same manner (1-to-1 replacements), many other wrongly recognized words (not only OOVs) were corrected by the SPE block:

- *ceața* (the fog) → *ceață* (fog): 6 corrections,
- *noi* (we) → *norii* (the clouds): 6 corrections,
- *continua* (will continue) → *continuă* (continues): 5 corrections,
- *sînt* ([they] are, old form) → *sunt* ([they] are): 7 corrections, etc.

A second conclusion that emerged from analyzing the replacements was that, besides the 1-to-1 replacements, the SPE block also performs many-to-many replacements, as follows:

- *climatul logica/logice* (logical climate) → *climatologice* (climatologically): 6 corrections,
- *va sta la dispoziție* (will be available)→ *vă stă la dispoziție* (is available): 7 corrections,
- *nori consistent şi* (consistent clouds and) → *nori consistenți* (consistents clouds): 3 corrections.

We further analyzed the SPE translation table and found that it has learnt many other replacement rules that are potentially useful for the weather news domain, but were not needed (and consequently not applied) in this evaluation:

- *bun găsit dantelă şi domnilor* (welcome, lace and gentlemen) → *bun găsit doamnelor şi domnilor* (welcome, ladies and gentlemen)
- *vor găsi tuturor* ([they] will find everyone) → *bun găsit tuturor* (welcome everyone)
- *teama de peste o vreme la această* (the fear over a while at this [time]) → *cam atât despre vreme la această* (that's all about the weather at this [time])
- *valori la prânz între opt şi* (values at noon between eight and...) → *valori cuprinse între opt şi* (values in the interval eigth and...)
- *noi taxe vreme* (new taxes weather) → *nu uitați vremea* (don't forget the weather)

One last important conclusion is that the SPE usually learns replacement rules for phrases which have similar pronunciations. This is important, because the SPE should not change the acoustical context. However the phrase translation table also contains (wrong) rules, for which the acoustical context is not preserved at all:

- *ce* (what) → *aceste* (these)
- *cinci* (five) → *în jur de* (around)

The above observation leaves room for an important improvement for this system: the SPE rules could be filtered based on the acoustical similarity of the replacement pairs.

6 Conclusion and Future Work

Several ASR error correction methods have been proposed in the past 10-15 years and were shown to improve the speech-to-text transcription process. Most of these methods regard the ASR system as a *black-box* and propose a correction block to post-process the raw transcripts.

This paper also proposed such an ASR correction block which uses SMT principles and tools to "translate" the raw transcripts into corrected transcripts. This SPE block takes advantage of the user generated corrections, in a scenario in which the ASR user must verify and correct the transcripts to be able to actually use them. We showed that the proposed method is scalable and gets better WER results as more user-generated data is used. We also deeply analyzed the ASR errors and the SPE corrections based on the output transcriptions and the SMT phrase translation table. A key conclusion which emerged from this analysis was that the correction block makes 1-to-1 replacements in particular word contexts, but also *many-to-many replacements*.

In the near future we plan to use the N-best raw transcripts along with the manually corrected transcripts to train the SPE system (increasing the size of the training parallel corpus). Moreover, we intend to introduce another factor in the factored translation model (of the SPE system) in order to weight the translation rules based on the *acoustical similarity* of the replacement pairs. This could be done by aligning the phonetic transcription of the replacement pairs and will provide an acoustical basis for our particular "translation" scenario.

A second research direction is to evaluate the ASR correction method on various domains with different characteristics in order to see if the method is effective for broader domains, how the vocabulary richness of the domain correlates with the amount of transcriptions that need to be post-processed (corrected), etc. In the same context, changing the speech domain could also imply changing the language, because the proposed approach could be easily adapted to other languages by changing the black-box ASR system.

Finally, we also intend to investigate whether the selection of the corrected transcriptions used for SPE training has any effect on the system performance. There might be transcription subsets which generate a more effective SPE and consequently, our research goal would be to find the best selection procedure.

References

1. Yu, D., Hwang, M., Mau, P., Acero, A., Deng, L.: Unsupervised Learning from Users' Error Correction in Speech Dictation. In: 8th International Conference on Spoken Language Processing (Interspeech), Jeju Island, Korea, pp. 1969–1972 (2004)

2. Cucu, H., Besacier, L., Burileanu, C., Buzo, A.: Investigating the Role of Machine Translated Text in ASR Domain Adaptation: Unsupervised and Semi-supervised Methods. In: The 2011 Automatic Speech Recognition and Understanding Workshop (ASRU 2011), Hawaii, USA, pp. 260–265 (2011)
3. Cucu, H., Besacier, L., Burileanu, C., Buzo, A.: ASR Domain Adaptation Methods for Low-Resourced Languages: Application to Romanian Language. In: 20 th European Signal Processing Conference (EUSIPCO), Bucharest, Romania (2012)
4. Ringger, E.K., Allen, J.F.: A Fertility Channel Model for Post-Correction of Continuous Speech Recognition. In: 4th International Conference on Spoken Language Processing, Philadelphia, USA, vol. 2, pp. 897–900 (1996)
5. Jung, S., Jeong, M., Lee, G.G.: Speech recognition error correction using maximum entropy language model. In: 8th International Conference on Spoken Language Processing (Interspeech), Jeju Island, Korea, pp. 2137–2140 (2004)
6. Kaki, S., Sumita, E., Iida, H.: A method for correcting errors in speech recognition, using the statistical features of character co-occurrence. In: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (COLING-ACL), Montreal, Canada, pp. 653–657 (1998)
7. Brandow, R.L., Strzalkowski, T.: Improving speech recognition through text-based linguistic post-processing. United States Patent 6064957 (2000)
8. Mangu, L., Padmanabhan, M.: Error corrective mechanisms for speech recognition. In: 27th International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Salt Lake City, USA, vol. 1, pp. 29–32 (2001)
9. Sarma, A., Palmer, D.D.: Context-based speech recognition error detection and correction. In: Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2004), Boston, USA, pp. 85–88 (2004)
10. Moses Statistical Machine Translation Toolkit, `http://www.statmt.org/moses`
11. Simard, M., Ueffing, N., Isabelle, P., Kuhn, R.: Rule-Based Translation with Statistical Phrase-Based Post-Editing. In: 2nd Workshop on Statistical Machine Translation, Prague, Czech Republic, pp. 203–206 (2007)
12. Simard, M., Goutte, C., Isabelle, P.: Statistical Phrase-based Post-editing. In: Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2007), Rochester, NY, USA, pp. 508–515 (2007)
13. Lagarda, A.L., Alabau, V., Casacuberta, F., Silva, R., Díaz-de-Liaño, E.: Statistical Post-Editing of a Rule-Based Machine Translation System. In: Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2009), Boulder, CO, USA, pp. 217–220 (2009)
14. Cucu, H., Besacier, L., Burileanu, C., Buzo, A.: Enhancing Automatic Speech Recognition for Romanian by Using Machine Translated and Web-based Text Corpora. In: The 15th International Conference SPECOM, Kazan, Russia (2011)
15. CMU-Sphinx Speech Recognition Toolkit, `http://cmusphinx.sourceforge.net`
16. SRI Language Modeling Toolkit, `http://www.speech.sri.com/projects/srilm`

Reward Shaping for Statistical Optimisation of Dialogue Management

Layla El Asri[1,2], Romain Laroche[1], and Olivier Pietquin[2]

[1] Orange Labs, 38-40 rue du Général Leclerc 92794 Issy-les-Moulineaux, France
[2] SUPELEC Metz Campus, IMS-MaLIS Research Group, UMI 2958 (CNRS - GeorgiaTech), 2 rue Edouard Belin 57070 Metz, France
{layla.elasri,romain.laroche}@orange.com, olivier.pietquin@supelec.fr

Abstract. This paper investigates the impact of reward shaping on a reinforcement learning-based spoken dialogue system's learning.

A diffuse reward function gives a reward after each transition between two dialogue states. A sparse function only gives a reward at the end of the dialogue. Reward shaping consists of learning a diffuse function without modifying the optimal policy compared to a sparse one.

Two reward shaping methods are applied to a corpus of dialogues evaluated with numerical performance scores. Learning with these functions is compared to the sparse case and it is shown, on simulated dialogues, that the policies learnt after reward shaping lead to higher performance.

Keywords: Spoken Dialogue Systems, Evaluation, Reinforcement Learning.

1 Introduction

Dialogue management is one of the core functionalities of a Spoken Dialogue System (SDS) along with automatic speech recognition, natural language understanding, natural language generation and speech synthesis. The Dialogue Manager (DM) sequences the interaction with the user. It chooses the action to perform according to its beliefs about the current state of the dialogue. Actions the DM can perform might be: asking for a piece of information, asking the user to confirm a statement, *etc.* Hand-coding the behaviour of the DM is time consuming and results in a specific implementation difficult to transfer to other domains. Therefore, statistical learning of the DM's behaviour through Reinforcement Learning (RL) [22] has become a popular technique: the DM is modelled as a sequential decision making agent and it selects actions in order to maximise a numerical return. This return is computed from a reward function provided by the SDS designer [10]. Ideally, the reward function is to be conceived as the most succinct, robust and transferable representation of the system's task [19].

However, it is common to define this function based on SDS designer intuition and experience, not relying on any data. Only a few studies have been conducted to learn a reward function from data. Among them, Walker *et al* [24] proposed a PARAdigm for DIalogue System Evaluation (PARADISE), modelling system performance as a linear function of task completion and dialogue costs (duration of the dialogue, number of speech recognition rejections...). Walker *et al.* [23] as well as Rieser and Lemon [18] evaluated the performance of different systems using the PARADISE framework and used this evaluation as a reward function. Yet, PARADISE requires to automatically compute task completion, which is not always possible. Besides, the linear representation of system performance has been criticised for not having a strong theoretical nor experimental grounding [7]. Another methodology was proposed, which consists of learning, from examples of expert behaviour, the reward function that describes best the task being completed by that expert. This approach is known as Inverse Reinforcement Learning (IRL) [19]. It was first suggested for dialogue management by Paek and Pieraccini [15] who thought of using IRL on Human-Human dialogues to learn a reward function enabling the SDS to mimic human operators behaviour. Following this idea, Boularias *et al.* [2] learnt a reward function from dialogues collected in a Wizard-of-Oz (WOZ) setting where a human expert replaces the DM. However, it is not always possible to learn from a human expert. For example, a DM could have to choose between different speech styles and these choices can only be made statistically. Besides, it is difficult to transpose speech recognition issues to WOZ experiments. IRL has also been used to model user behaviour for dialogue simulations [4].

In previous work, we proposed two algorithms using a corpus of manually evaluated dialogues (with numerical performance scores) to compute a reward function [5]. It was shown that a reward function that predicts accurately the subjective performance for a given dialogue could be learnt, even from a corpus of small size. Sample efficiency has been an important subject of research in the field of dialogue management [11,16]. A learning algorithm is said to be sample efficient if it can learn a near-optimal policy with only a few dialogues, meaning that it optimises data exploitation. It is costly to conduct evaluation campaigns on an SDS and most of the time, only a few number of evaluated dialogues can be collected, hence the importance of optimal data exploitation.

The reward function learnt with the methods we previously proposed gives a reward after each transition between two dialogue states while keeping the optimal policy unchanged compared to a sparser reward. We will call such a function *diffuse* in contrast with the sparse case where a reward is only received at the end of the dialogue. In this paper, it is shown, on a simulated corpus with user behaviour inferred from real dialogues, that the policy learnt with diffuse rewards entails higher performance than the one learnt with sparse rewards.

2 Reinforcement Learning for Dialogue Management

Dialogue management is cast as a sequential decision making problem, modelled by a Markov Decision Process (MDP) (S, A, T, R, γ) where S is the state space, A the action space, T the transition probabilities: $\forall\ (s = s_t, a = a_t, s' = s_{t+1}),\ T(s, a, s') = P(s' \mid s, a) \in [0, 1]$, R the reward function: $\forall(s = s_t, s' =$

$s_{t+1})$, $R(s, s') \in \mathbb{R}$ and $\gamma \in]0, 1[$ a discount factor. A similar MDP without a reward function is denoted MDP$\backslash R$.

A deterministic policy π is a function mapping each state to a unique action: $\forall\, s \in S, \pi(s) = a \in A$. The immediate reward received after a transition (s_t, s_{t+1}) is $R_t = R(s_t, s_{t+1}) \in \mathbb{R}$. The cumulative reward (or return) at time t is the discounted sum of immediate rewards: $r_t = \sum_{k \geq 0} \gamma^k R_{t+k}$. Given a policy π, V^π is the state value function, the value $V^\pi(s)$ of a state s being the expected return $E[r_t \mid s_t = s, \pi]$ over all possible trajectories starting in state s_t and following π. Likewise, Q^π is the state-action value function, the value $Q^\pi(s, a)$ of a state-action couple (s, a) being $E[r_t \mid s_t = s, a_t = a, \pi]$. The dialogue manager aims to find an optimal policy: a mapping selecting actions maximising the expected return for every state. An optimal policy π^* is thus such that $\forall\, \pi, \forall\, s, V^{\pi^*}(s) \geq V^\pi(s)$. Although uniqueness of the optimal policy is not guaranteed, all optimal policies share the same state and state-action value functions noted V^* and Q^* and thus perform comparatively. In the context of this paper, time is measured in number of dialogue turns, each dialogue turn occurring in between two results of automatic speech recognition.

If many dialogue parameters are taken into account, the state space can become computationally intractable so designers usually define a summary state space instead. A summary state is an agglomerate of states with similar features. For example, for the SDS described in [8] which provides information about local restaurants, the current state can be summed up in terms of *empty*, *filled* and *confirmed* items (location, price range, type of food) instead of listing the current values of all items (*e.g.* location=city center, price range=cheap, type=Italian).

The reward function is hard to define *ex nihilo* as one should be able to numerically translate and appropriately distribute qualitative requirements. For instance, one has to decide which prevails between task completion and speech recognition rejections, when should the rewards be given during the dialogue, which numerical range, and so forth. Our approach to this problem is given in the following section which briefly presents two algorithms computing, from a corpus of manually evaluated dialogues, a diffuse reward function, defined over a summary state space $\tilde{S}$ [5]. These algorithms solve the problem introduced in Definition 1. The manual evaluations in question are based on dialogue features representative of system usability such as dialogue length, task completion,... They can be computed from user answers to a Likert-scale questionnaire. In this paper, dialogues are simulated and the scores are a linear combination of such dialogue features.

Definition 1 (Reward inference problem). *Infer a reward function from a corpus of N dialogues* $(D_i)_{i \in 1..N}$ *among which p dialogues have been manually evaluated with a numerical performance score* $P_i \in \mathbb{R}$.

3 Learning Rewards from Data

This section presents three different approaches to the problem issued in Definition 1. The first two algorithms infer diffuse reward functions. Details can be found in [5]. The third one, which will serve as a baseline, gives a reward equal to system performance at the end of the dialogue.

3.1 Reward Shaping

This first algorithm is named Reward Shaping in reference to the line of research which aims to include, without modification of the optimal policy, immediate rewards as progress estimators instead of having to wait until the end of an episode to receive a reward [12]. Ng *et al.* [14] proved that the optimal policy of an MDP would not be changed by adding to the reward function a potential-based reward function U such that $U(s,s') = \gamma\Phi(s') - \Phi(s)$ and experimentally validated that, if Φ is equal to the value function, learning speed is increased. In our case, reward shaping consists of using performance scores to evaluate each state and then defining the reward associated to a given transition as the difference of potential between the arrival and the initial state.

The value function V^π is estimated according to the performance scores: the return used to estimate the value of each summary state $V^\pi(\tilde{s})$ is $\forall\ D_i,\ r_t = \gamma^{-t}P_i$. Thus, the global return r_0 is equal to P_i.

The reward function (denoted R_{RS}) is then defined as $\forall\ (\tilde{s},\tilde{s}'),\ R_{RS}(\tilde{s},\tilde{s}') = \gamma V^\pi(\tilde{s}') - V^\pi(\tilde{s}) + \delta_{\tilde{s}=\tilde{s}_0} V^\pi(\tilde{s}_0)$ with δ the Kronecker symbol ($\delta_{\tilde{s}=\tilde{s}_0} = 1$ if $\tilde{s} = \tilde{s}_0$ and 0 otherwise). In other words, the reward function is modelled as the sum of an offset $C_0 = V^\pi(\tilde{s}_0)$ and the potential-based function $U(\tilde{s},\tilde{s}') = \gamma V^\pi(\tilde{s}') - V^\pi(\tilde{s})$. With R_{RS}, the global return r_0 for a given dialogue D (lasting from turns 0 to t_f) is: $r_0 = \hat{P} = \gamma^{t_f} V^\pi(\tilde{s}_{t_f})$. Since V^π is estimated according to the returns $r_t = \gamma^{-t}P_i$, $\gamma^{t_f}V^\pi(\tilde{s}_{t_f})$ is an estimation of the performance of the dialogues ending with state $\tilde{s}_{t_f}$ and r_0 is an estimation of the performance of the system during D.

3.2 Distance Minimisation

Instead of evaluating states, distance minimisation evaluates transitions. This algorithm directly aims to cut the performance evaluations into local rewards over the transition space. The distance minimisation problem is formalised in Definition 2.

Definition 2. *Let an MDP$\backslash$R. Let $\phi = [\phi_i]_{i=1,\dots,m}$ be a vector of features over the transition space ($\forall\ i \in [1,m], \forall\ (\tilde{s},\tilde{s}')$, $\phi_i(\tilde{s},\tilde{s}') \in [0,1]$). The immediate reward R_t following transition $(\tilde{s}_t,\tilde{s}'_t)$ is modelled as a linear sum of these features: $R_t = \sum_{i=1}^m \omega_i\phi_i(\tilde{s}_t,\tilde{s}'_t)$. Let $P = [P_i]_{i=1,\dots,p}$ be a performance score vector such that each dialogue D_i is associated with a performance P_i, and let d_P be a distance measure between P and the return vector r_0[1]. The distance minimisation problem consists of finding the weight vector ω^* such that $\omega^* = \operatorname{argmin}_\omega d_P(\omega)$.*

[1] $(r_0)_{1\leq i\leq p}$, $\forall i$, $r_{0i} = \sum_{t\geq 0}\gamma^t R_t = \sum_{\tilde{s}_t,\tilde{s}'_t}\gamma^t\sum_{j=1}^m \omega_j\phi_j(\tilde{s}_t,\tilde{s}'_t)$.

Here, Euclidean distance minimisation is solved. The problem issued in Definition 2 can be cast as a quadratic optimisation problem and solved with well-known direct or iterative methods (resolution details can be found in [5]). The resulting reward function is denoted R_{DM}.

3.3 Performance Scores

R_{RS} and R_{DM} are compared to the sparse reward function which gives the performance score at the end of each dialogue. This function (denoted R_{PS}) is defined as follows: $\forall\ D_i, \forall\ (\tilde{s}, \tilde{s}'),\ R_{PS}(\tilde{s}, \tilde{s}') = 0$ if $\tilde{s}' \neq \tilde{s}_f$ and $R_{PS}(\tilde{s}, \tilde{s}') = \gamma^{-t_f-1} P_i$ otherwise[2].

In the following section, the performance of the policies learnt with R_{PS}, R_{RS} and R_{DM} on the same corpus of dialogues are compared.

4 Experimental Setting

4.1 System Overview

We used the TownInfo system, based on the DIPPER architecture [1]; it provides informations about restaurants in a given city depending on three criteria: location, price range and type of food [9]. At each time step, a slot corresponding to one of these criteria can either be empty, filled or confirmed. We defined a summarised state space which counts the number of empty, filled and confirmed slots. We also defined a summary action space which does not differentiate the actions according to the position of the slot involved (for instance, AskSlot1, AskSlot2 and AskSlot3 are summarised into AskASlot). Nevertheless, to assure dialogue coherence and avoid *e.g.* asking for a slot that has already been confirmed, when an action is chosen and has to be mapped to a slot, for example, AskASlot, we first check the current value of the slots and then force this action to be mapped only to empty slots. The state and action spaces were voluntarily made simple as the main objective of this paper is to validate the diffuse rewards approach.

4.2 Dialogue Simulations

The three reward functions were applied to a corpus of 600 simulated dialogues. User was simulated according to the Bayesian method proposed in [17]. It consists of modelling user behaviour as a Bayesian network to simulate dialogues at the intention level, including grounding behaviours. The parameters of the Bayesian network were trained on the 1234 human-machine dialogues which are described in [25].

As for system policy, it was set to be uniform to collect as much information as possible for every state-action pair.

After each dialogue, a performance score was computed according to dialogue features, in a PARADISE-like manner. Once again, since our aim was to validate

[2] So that $r_0 = P_i$.

the diffuse rewards approach, a simple, automatically computed scoring function was sufficient. With nbEmpty the number of empty slots, nbRight, the number of slots that were correctly filled, nbWrong, the number of incorrectly filled slots and nbTurns, the number of dialogue turns, the score was:

$$\begin{aligned} \text{score} = & -3 \times \text{nbEmpty} + 0.25 \times \text{nbRight} \\ & -0.75 \times \text{nbWrong} - 0.015 \times \text{nbTurns} \end{aligned} \tag{1}$$

4.3 Learning a Diffuse Reward Function

We simulated 2300 dialogues in total but we only used 600 dialogues to learn R_{RS} and R_{DM} since it is difficult, in real-life experiments to obtain as many as 2300 dialogues. The proximity between R_{RS}, R_{DM} and the simulated performance scores (see Equation 1) was assessed by Spearman's rank correlation coefficient [20]. The closer to 1 the correlation coefficient, the stronger the relationship between the corresponding rankings.

We used the remaining 1700 dialogues to measure this proximity. We drew 100 times 600 dialogues from the corpus of simulations and computed the mean correlation coefficient on these runs for both R_{RS} and R_{DM}. The mean correlation coefficient was equal to 0.81 for R_{RS} and 0.84 for R_{DM}. Here, the coefficients are high because the scoring function in equation 1 can be approximated on the state space presented in Section 4.1 as the only non-observable parameter is the number of correctly filled slots.

4.4 Learning a Near-Optimal Policy

Policies were learnt on the 600 dialogues with R_{RS}, R_{DM} and R_{PS} using Least-Squares Policy Iteration (LSPI, [6]). LSPI is an approximate policy iteration algorithm involving LSTDQ[3] which learns an approximate state-action value function for a given policy from a fixed data set. After a policy was learnt with LSPI, 200 new dialogues were generated with this policy and the dialogues were automatically evaluated according to Equation 1. We also applied LSPI to the whole corpus of 2300 dialogues and compared the three resulting policies on 200 dialogues. Our aim is to show that learning with R_{RS} and R_{DM} leads to higher performance no matter the size of the training corpus.

5 Results

Table 1 shows that though the policy learnt with R_{RS} leads to longer dialogues, it has the best evaluation. This can be explained by the fact that this policy has a better success at confirming slots than the other two. A great number of confirmed slots implies a limited risk of getting one value wrong and since filling and having the right value for each slot have a greater weight in the scoring

[3] An extension to control problems of Least-Squares Temporal Differences, LSTD [3].

Table 1. 95% confidence interval for the mean performance, mean number of dialogue turns and mean number of empty and confirmed slots on 200 dialogues simulated with the policies learnt with R_{RS}, R_{DM} and R_{PS} after 600 and 2300 dialogues

Learning on 600 dialogues	Performance	Turns	Empty	confirmed
R_{RS}	0.13 ± 0.09	11.6	0	3
R_{DM}	0.062 ± 0.10	7.72	0	0
R_{PS}	0.007 ± 0.10	8.55	0	0.8
Learning on 2300 dialogues	Performance	Turns	Empty	confirmed
R_{RS}	0.13 ± 0.09	11.6	0	3
R_{DM}	0.08 ± 0.10	7.28	0	0.23
R_{PS}	0.04 ± 0.11	7.95	0	1.22

function (see Equation 1) than having short dialogues, the policy learnt with R_{RS} achieves better performance than the ones learnt with R_{DM} and R_{PS}. For R_{RS}, the results are the same after 600 and 2300 dialogues because the policies learnt with LSPI on these two corpora are similar.

R_{PS} gives the exact scores as rewards which makes it more accurate than R_{RS} and R_{DM} but this accuracy is counterbalanced by the fact that the rewards are only given at the end of each dialogue. The policy learnt with R_{PS} is the least competitive because it more poorly balances the trade-off between the number of confirmed slots and dialogue length than the other two policies.

6 Relation to Prior Work

Walker *et al.* [23] used performance evaluation to learn a policy for an SDS with Q-Learning [22], giving a reward equal to the evaluation at the end of each dialogue. This SDS granted a vocal access to the user's e-mail account and could summarise and read messages. Walker *et al.* showed that about hundred dialogues were sufficient to learn the best strategy between system and mixed initiative yet it was not enough for the summary strategy to achieve convergence. We showed, on a different dialogue task, that it is possible to shape a reward function based on performance evaluation in order to optimise corpus exploitation. We believe that reward shaping is a promising method for statistical dialogue management optimisation as it is often difficult to obtain corpora of great size.

Meguro *et al.* [13] designed a listening-oriented dialogue system and inferred a reward function from third-party evaluation of user satisfaction. In order to counter inter-annotators ambiguity concerning the interpretation of the Likert scale, Sugiyama *et al.* [21] introduced Preference-based Inverse Reinforcement Learning (PIRL): performance scores are used to deduce the best of two dialogues and then a reward function that classifies dialogues respecting the same order is learnt. Contrary to our reward inference algorithms, this method does not enable to use directly performance scores given by users. Indeed, each user would have to interact several times with the system for us to infer a ranking from their evaluation, otherwise a third-party annotator would be required.

7 Conclusion

This paper provided some empirical results on the issue of reward function design for spoken dialogue systems. A diffuse reward function was learnt from a corpus of evaluated dialogues. It was shown that diffuse rewards enabled to learn a policy leading to a better performance on new dialogues.

Future work will include defining a compatible active learning framework and proposing a method to optimise the conception of the summary state space. We will also compare our reward inference methods to Preference-Based Inverse Reinforcement Learning on dialogues evaluated by a third-party annotator.

Acknowledgements. The authors would like to thank Heriot-Watt University's *Interaction lab* for providing help with DIPPER. The authors would also like to thank PhD student Edouard Klein from the IMS-MaLIS research group in Supélec for providing a C implementation of LSPI and Olivier Pietquin thanks Région de Lorraine for financial support.

References

1. Bos, J., Klein, E., Lemon, O., Oka, T.: DIPPER: Description and Formalisation of an Information-State Update Dialogue System Architecture. In: Proceedings of SIGdial Workshop on Discourse and Dialogue (2003)
2. Boularias, A., Chinaei, H.R., Chaib-draa, B.: Learning the reward model of dialogue pomdps from data. In: Proceedings of NIPS (2010)
3. Bradtke, S.J., Barto, A.G.: Linear least-squares algorithms for temporal difference learning. Machine Learning 22, 33–57 (1996)
4. Chandramohan, S., Geist, M., Lefèvre, F., Pietquin, O.: User simulation in dialogue systems using inverse reinforcement learning. In: Proceedings of Interspeech (2011)
5. El-Asri, L., Laroche, R., Pietquin, O.: Reward function learning for dialogue management. In: Proceedings of STAIRS (2012)
6. Lagoudakis, M.G., Parr, R.: Least-squares policy iteration. Journal of Machine Learning Research 4, 1107–1149 (2003)
7. Larsen, L.B.: Issues in the evaluation of spoken dialogue systems using objective and subjective measures. In: Proceedings of IEEE ASRU, pp. 209–214 (2003)
8. Lemon, O., Georgila, K., Henderson, J., Stuttle, M.: An ISU dialogue system exhibiting reinforcement learning of dialogue policies: Generic slot-filling in the talk in-car system. In: Proceedings of EACL (2006)
9. Lemon, O., Georgila, K., Henderson, J., Stuttle, M.: An ISU dialogue system exhibiting reinforcement learning of dialogue policies: generic slot-filling in the talk in-car system. In: Proceedings of EACL (2006)
10. Lemon, O., Pietquin, O.: Machine learning for spoken dialogue systems. In: Proceedings of Interspeech, pp. 2685–2688 (2007)
11. Li, L., Williams, J.D., Balakrishnan, S.: Reinforcement learning for dialog management using least-squares policy iteration and fast feature selection. In: Proceedings of Interspeech (2009)
12. Mataric, M.J.: Reward functions for accelerated learning. In: Proceedings of ICML, pp. 181–189 (1994)

13. Meguro, T., Higashinaka, R., Minami, Y., Dohsaka, K.: Controlling listening-oriented dialogue using partially observable markov decision processes. In: Proceedings of Coling (2010)
14. Ng, A.Y., Harada, D., Russell, S.: Policy invariance under reward transformations: Theory and application to reward shaping. In: Proceedings of ICML, pp. 278–287 (1999)
15. Paek, T., Pieraccini, R.: Automating spoken dialogue management design using machine learning: An industry perspective. Speech Communication 50, 716–729 (2008)
16. Pietquin, O., Geist, M., Chandramohan, S., Frezza-Buet, H.: Sample-efficient batch reinforcement learning for dialogue management optimization. ACM Transaction on Speech and Language Processing 7(3), 1–21 (2011)
17. Pietquin, O., Rossignol, S., Ianotto, M.: Training Bayesian networks for realistic man-machine spoken dialogue simulation. In: Proceedings of IWSDS 2009 (2009)
18. Rieser, V., Lemon, O.: Learning and evaluation of dialogue strategies for new applications: Empirical methods for optimization from small data sets. Computational Linguistics 37 (2011)
19. Russell, S.: Learning agents for uncertain environments (extended abstract). In: Proceedings of COLT (1998)
20. Spearman, C.: The proof and measurement of association between two things. American Journal of Psychology 15, 72–101 (1904)
21. Sugiyama, H., Meguro, T., Minami, Y.: Preference-learning based Inverse Reinforcement Learning for Dialog Control. In: Proceedings of Interspeech (2012)
22. Sutton, R.S., Barto, A.G.: Reinforcement Learning. An introduction, pp. 56–57. MIT Press (1998)
23. Walker, M.A., Fromer, J.C., Narayanan, S.: Learning optimal dialogue strategies: A case study of a spoken dialogue agent for email. In: Proceedings of COLING/ACL, pp. 1345–1352 (1998)
24. Walker, M.A., Litman, D.J., Kamm, C.A., Abella, A.: PARADISE: a framework for evaluating spoken dialogue agents. In: Proceedings of EACL, pp. 271–280 (1997)
25. Williams, J.D., Young, S.: Partially observable markov decision processes for spoken dialog systems. Computer Speech and Language 21, 231–422 (2007)

Discriminative Framework for Spoken Tunisian Dialect Understanding

Marwa Graja, Maher Jaoua, and Lamia Hadrich Belguith

MIRACL Laboratory, Arabic Natural Language Processing Research Group (ANLP-RG)
University of Sfax, Tunisia
{marwa.graja,maher.jaoua,l.belguith}@fsegs.rnu.tn

Abstract. In this paper, we propose to evaluate the performance of a discriminative model to semantically label spoken Tunisian dialect turns which are not segmented into utterances. We evaluate discriminative algorithm based on Conditional Random Fields (CRF). We check the performance of the CRF model to concept labeling on raw data in Tunisian dialect which are not analyzed in advance. We compared its performance with different types of preprocessing data until arriving to well treated data. CRF model showed the ability to ameliorate the accuracy of labeling task for spoken language understanding of not segmented and not treated speech in Tunisian dialect.

Keywords: concept labeling, discriminative model, speech understanding, Tunisian dialect.

1 Introduction

Spoken Language Understanding is an important component in spoken dialogue systems. It aims to clarify meaning from spontaneous speech [1]. The first level of spoken language understanding is concept labeling which consists in extracting concepts and their relation from transcribed speech. In fact, the concept labeling task is semantic labeling of transcribed words as input and concepts as output labels.

To perform semantic labeling, many statistical methods have been used, from generative to discriminative models [2]. Conditional random fields (CRF) model is the best of generative and discriminative models [3]. However, this model has been usually applied to semantic labeling of spontaneous speech for Latin languages such as English, French or Spanish [4]. However, speech understanding of Arabic dialect is still quite processed in the scientific research. Despite the importance of semantic analysis for the implementation of any dialogue system, there are only a few works which are interested in the automatic understanding of spoken standard Arabic [5][6] and not Arabic dialect. In this paper, we propose to evaluate a discriminative model on the spoken Tunisian dialect in the context of a definite task to semantically label oral utterances. In fact, we are interested in evaluating discriminative algorithm based on CRF model to semantic label raw data in Tunisian dialect which are not analyzed in advance. Then, we try to improve the raw data by applying several levels of treatments to evaluate the performance of the CRF model to label spoken Tunisian dialect.

We are interested in this paper in semantic labeling of client turns which are not segmented into utterances. This idea has been used by [4] and represents the more realistic situation of spontaneous speech. In fact, it is easier to segment dialogues into turns than into utterances.

According to our knowledge, spontaneous-speech dialogue corpora in Tunisian dialect are not very current. So, possibilities to test dialogue technologies on Tunisian dialect are also not common. In this article, the corpus selected for testing the proposed model is the TUDICOI corpus [7], which is a spoken task-oriented dialogue corpus in Tunisian dialect. The TUDICOI task consists of the request information about railway services recorded in train station.

This paper is organized as follows: the next section presents the Tunisian dialect. Section 3 describes the TUDICOI corpus in Tunisian Dialect used for experiments. The section 4 presents the CRF discriminative model used to perform semantic labeling. Experimental results are shown in section 5. Conclusion is drawing in the last section.

2 Tunisian Dialect

Arabic language is known by three main collections: the Classical Arabic, standard Arabic and dialectal Arabic. The dialectal Arabic is the real form of the language [8] since it is used in all informal communications. They are generally limited in use for request information and everyday communication. The main characteristic is that they are mainly spoken and not written. So it is so important to consider dialects in spoken dialogue system. Tunisian dialect is a subset of Arabic dialects related to the Arab Maghreb (western Arab world). Like all Arabic dialects, it is characterized by morphology, syntax, phonology and lexicon which have similarities and differences compared to the standard Arabic, and even to other Arabic dialects. Tunisian dialect is strongly influenced by the Berber and also by other languages such as Turkish, Italian and French. It has several large regional varieties, but the variety of Tunis (used in the capital of Tunisia) is the most understood by all Tunisians [9].

3 Spoken Dialogue Corpus for Tunisian Dialect

The construction of a dialogue corpus represents a big challenge especially when we deal with Arabic dialects which suffer from lack of resources [10] [11]. In fact and according to our knowledge, there is no spoken dialogue corpus in Tunisian dialect dealing with a limited task. In this context, we have produced an initial corpus of real spoken dialogue corresponding to the task of railway request information in collaboration with the National Company of Railway in Tunisia (SNCFT)[1]. This corpus is called TUDICOI as TUnisian DIalect COrpus Interlocutor.

The main task of the TUDICOI corpus is request information in Tunisian dialect about the railway services in the train station. These requests are about train schedule consultation, train type, train destination, train path, ticket price and ticket booking. Based on these requests, several requests can be combined together during a dialogue

[1] http://www.sncft.com.tn/

between the staff and the client about railway services in the train station. An example of a real dialogue in Tunisian dialect between a client and a staff is shown below in Table 1. Two speaker types have participated in this dialogue who are the clients (C) and the staffs (S).

Table 1. A sample of real dialogue in Tunisian dialect between a client (C) and a staff (S)

Turn	*Transliteration* Translation	Transcription
C:	*sAmHny wqtA$ yxrj EttrAn ltwns* Excuse me when the train leaves to Tunis	سامحني وقتاش يخرج التران لتونس
S:	*mADy sAEh wrbEh OdrAj* One hour past twenty minutes	ماضي ساعه وربعه أدراج
C:	*bqdA$ hwA Ettkyh* How much the ticket	بقداش هوا التكيه
S:	*vnA$ nlf wxmsmyh ltwns* Twelve dinars and five hundred to Tunis	ثناش نلف وخمسميه لتونس

The TUDICOI corpus consists of 1825 dialogues from 1831 users. These dialogues represent 12182 utterances. The most important characteristics are shown in Table 2.

Table 2. Main characteristics of the TUDICOI Corpus

# Dialogues	1825
# Speakers	1831
# Client turns	6533
# Staff turns	5649
# Words in client turns	21682
# Vocabulary size in client turns	108

The 1825 dialogues are composed of 6533 client turns and 5649 staff turns. On average, each dialogue consists of three turns for client and three turns for staff. In addition each client turn is composed of an average of 3.3 words. It's so important to note that the average of words per client turn is very low. This is due to key words used by clients to request for information about railway services.

It's so important to notice that we are interested in client turns. That's why we have manually labeled only transcribed speech data of clients based on semantic point of view in order to build a language model for client utterances. In fact, we have established a well defined semantic annotation scheme for client utterances for the railway request information task to cover all aspects of client utterances in the studied task.

The annotation scheme for concept labels respects many of the principles used in other speech annotated corpus with a structure which covers the more specific details of the task. Semantic concept labels used to label all versions of the annotated corpus are shown in Table 3.

Table 3. Semantic Concept labels

Domain concepts	
Train	Ticket_Numbers
Train_Type	Ticket
Departure_hour	Hour_Cpt
Arrival_hour	Departure_Cpt
Day	Arrival_Cpt
Origin	Price_Cpt
Destination	Class_Cpt
Fare	Trip_time
Class	Ticket_type
Link concepts	
Choice	
Coordination	

Requests concepts	
Path_Req	Existence_Req
Hour_Req	Trip_timeReq
Booking_Req	Clarification_Req
Price_Req	
Dialogue concepts	
Rejection	
Acceptance	
Politeness	
Salutation (Begin)	
Salutation (End)	
Out of vocabulary	
Out	

Generally, the most of works dealing with speech corpus perform some automatic treatments on the corpus before annotation. These treatments are done to reduce the complexity of the corpus and the structures [4]. In our case, we have two versions of annotated corpus. In the first version, we did not perform any treatments on the corpus before annotation. Indeed, we annotated the raw version of the client turns to look for results of discriminative models when we deal with very raw quality of the corpus. This can give important results about the robustness of discriminative models against deteriorated data. In the second version of the annotated corpus, we have performed some automatic treatments before annotation to improve the turn's structure. These treatments include the following points:

- Lexical normalization: since the transcription was done manually, any word in the TUDICOI corpus can be written in different orthographic ways. For example, we noticed that the word "رزرفسيون" "*Reservation*" is written in four different forms: "رزرفسيون", "رازرفسيون", "رازارفسيون", "ريزرفسيون". That's why we have performed automatic lexical normalization.
- Morphological analysis and lemmatization: in this analysis, we have performed verbs and nouns. Verbs treatment consists in determining the canonical form of the verb. For example, we replace the word "خارج" "*is going*" and "يخرج" "*goes*" by the following canonical form "خرج" "*go*". However, nouns treatment consists of two steps. The first step is returning to the singular form of the noun. The second step is replacing the definite form by the undefined form of the noun. As an example, the word "الترينوات" "*trains*" is transformed into "تران" "*train*".
- Synonyms treatment: this treatment consists in replacing each word by its synonym.

Given the lack of standard orthography and dictionaries for the Tunisian dialect, we have performed the lexical normalization and synonyms treatment by creating a lexicon dictionary for the railway request information domain. This dictionary helps us to correct the orthography and replace each word by its synonym.

Due to the absence of automatic analyzer for the Tunisian dialect, we have performed an automatic shallow morphological analysis for verbs and nouns by using the complete storage method [12]. Indeed, we have built a morphological base of possible changes for verbs and nouns which helps us to perform nouns and verbs treatments.

These two versions of the same annotated corpus are investigated to perform experiments and evaluate the CRF model on raw data and on very well treated data.

Given the complexity and time-consuming of the manual annotation task, we have only annotated 623 dialogues. These dialogues represent 2352 client turns. The most important characteristics of the annotated corpus are shown in Table 4.

Table 4. Main characteristics of annotated corpus

# Annotated dialogues	623
# Annotated client turns	2352
# Annotated words in client turns	7814

4 Statistical Method for Semantic Labeling

4.1 Sequence Labeling Task

The shallow parsing of spoken language understanding is sequence labeling which aims to attribute label sequences to a set of observation sequences of transcribed speech [13]. For example, consider the labeling task of words in an utterance with their corresponding concepts in the field of railway request information. In this task, each word is labeled with a concept indicating its appropriate semantic concept.

[مع Out] [وقتاش Hour_Req] [إي Out] [إي Out] [التران Train] [يمشي Departure_Cpt]

Fig. 1. Example of semantic labeling

4.2 Statistical Semantic Labeling

Previous works have dealt with sequence labeling task by means of statistical models. These models have been extremely investigated from generative to discriminative models. Raymond and al. [2] have compared SFST generative model to CRF discriminative model and [14] has shown that discriminative models are able to incorporate correlated features in conditional random fields (CRF). This advantage has reduced error rate for spoken language understanding compared to the generative model [15]. Based on its advantage, we choose the CRF model as a representative discriminative model to evaluate its performance on transcribed speech in Tunisian dialect which is not segmented into utterances and with different levels of treatment.

4.3 CRF Based Model

Conditional random fields (CRF) are undirected graphical models trained to maximize a conditional probability [16].

Lafferty et al. [16] define the conditional probability of a label sequence $y = y_1 \ldots y_N$ given an observation sequence $x = x_1 \ldots x_N$ as:

$$p(y/x) = \frac{1}{Z(x)} \exp \left(\sum_j \lambda_j t_j(y_{i-1}, y_i, x, i) + \sum_k \mu_k s_k(y_i, x, i)\right) \tag{1}$$

With:

$$\mathrm{Z(x)} = \sum_y \exp \left(\sum_j \lambda_j t_j(y_{i-1}, y_i, x, i) + \sum_k \mu_k s_k(y_i, x, i)\right) \tag{2}$$

$Z(x)$ is the normalization factor that makes the sum of all probabilities equals to one. $t_j(y_{i-1}, y_i, x, i)$ represents transition feature function of the entire observation sequence and the labels at positions i and $i-1$ in the label sequence. $s_k(y_i, x, i)$ represents state feature function of the label at position i in the observation sequence. λ_j and μ_k are parameters which are estimated from training data.

Given the model as defined in Equation (1), the most probable labeling sequence y^* for an input x, is:

$$y^* = arg\ max_y\ P(y/x) \tag{3}$$

We used in our experiments the CRF++ toolkit [17]. It is a simple, customizable and open source implementation of Conditional Random Fields (CFRs) for segmentation and labeling sequence data.

5 Experiments and Results

Multiple of works have dealt with the labeling problem of the spoken language understanding using several statistical methods for different languages. But these works are usually interested on segmented dialogues into utterances. Our proposal in these experiments is to test the performance of CRF model on spoken Tunisian dialect to label unsegmented turns. This situation appears more realistic since the speech recognition component provides as output turns for each speaker which are unsegmented into utterances. This idea is inspired from the works of Martinez and al. [5] who perform the labeling in the more realistic situation where the segmentation of turns into utterances is not available.

We evaluate the CRF model on the TUDICOI corpus for Tunisian dialect manually transcribed with unsegmented dialogue turns. We have used the same training corpus but treated differently. In fact, we have used two sets of annotated corpus to perform CRF model training. The first set (Set 1) is the annotated raw data version without any treatment in advance. The second set (Set 2) is the gold data of the annotated corpus which is analyzed morphologically, with lexical standardization and synonyms treatment. We learn the CRF model based on these two data sets to check the performance of CRF model in different levels of annotated data. Each set consists of 5710 words used as training corpus. To evaluate the performance of CRF model learned from these two sets, we prepared a test corpus which consists of 2104 words. The test corpus is used on its raw version for the first test. Then, it is performed with the same treatment done on Set 2.

The evaluation of the concepts sequence labeling is given in term of F-measure and concept error rate (CER). The CER is computed as incorrect prediction of the model by reference labels. Results are reported in Table 5.

$$F - mesure = \frac{2 \times precision \times recall}{precision + recall} \ and \ CER = \frac{\#incorrect\ concept\ predcition}{\#concept\_ref}$$

Table 5. CRF labeling results with different sets of annotated data

	Set 1	**Set 2**
#Training words	5710	5710
#Test words	2104	2104
Correct prediction	1243	1332
Incorrect prediction	223	198
Reference concepts	1800	1800
CER	**12%**	**11%**
Precision	**84%**	**87%**
Recall	**69%**	**74%**
F-measure	**75%**	**80%**

The model learned from the Set 2 has reduced the error rate compared to the CRF model learned from Set 1 (from 12% to 11%). This proves the importance of treatment for learning CRF model to simplify the utterance structures.

Despite the raw quality of the first set, CRF performs well in comparison with automatic labeling task using knowledgebase method. In fact, CRF performs better compared to result reported in [18] which uses domain ontology as knowledgebase with well performed data prepared in advance. This is a very important result since CRF model performs well with minimal preprocessing on the training data. This shows the robustness of such a model with noisy data by comparing it with a method based on knowledge which requires robust preprocessing to the training data. By manual examination of automatic labeling result using CRF, we found that the CRF have the ability to detect composed token specific to the task and label them correctly. CRF semantic labeling has failed in the case where labeled word in the training data depends on the client intension which is an important feature in the TUDICOI corpus. Indeed, it should be noted that the TUDICOI corpus is a pilot corpus for direct communications between clients and agents. So the client bases on gestures and intensions to request for information.

6 Conclusion

In this paper, we have evaluated the performance of the CRF model to perform semantic labeling task for spontaneous speech in Tunisian dialect in the context of spoken language understanding. These evaluations are done in the more realistic situation where the segmentation of turns into utterances is not available. We have performed experiments based on different levels of treated data, from raw to very well performed transcribed speech. Discriminative model showed robustness against raw turns which are not treated in advance and not segmented into utterances. CRF model has the ability to improve semantic labeling with minimal data treatment in comparison with knowledgebase method which requires very well treated data in advance. So, CRF model is very adequate for languages which suffer from luck of resources such as the Tunisian dialect.

Acknowledgment. We thank Frédéric Béchet; a researcher in the field of Speech and Natural Language processing at the LIF laboratory of Marseille; for all helpful and enlightening discussions to develop this work.

References

1. Aust, H., Oerder, M., Seide, F., Steinbiss, V.: The Philips automatic train timetable information system. Speech Communication 17, 249–263 (1995)
2. Raymond, C., Riccardi, G.: Generative and discriminative algorithms for spoken language understanding. In: Proceedings of Interspeech, Antwerp, Belgium (2007)
3. Sha, F., Pereira, F.: Shallow parsing with Conditional random fields. In: Proceedings of HLT-NAACL, pp. 134–141 (2003)
4. Martínez-Hinarejos, C.D., Benedí, J.M., Granell, R.: Statistical framework for a Spanish spoken dialogue corpus. Speech Communication (2008)
5. Zouaghi, A., Zrigui, M., Ben Ahmed, M.: Évaluation des performances d'un modèle de langage stochastique pour la compréhension de la parole arabe spontanée. TALN (2007)
6. Bahou, Y., Hadrich Belguith, L., Ben Hamadou, A.: Towards a Human-Machine Spoken Dialogue in Arabic. In: LREC (2008)
7. Graja, M., Jaoua, M., Hadrich Belguith, L.: Lexical Study of A Spoken Dialogue Corpus in Tunisian Dialect. In: The International Arab Conference on Information Technology (ACIT), Benghazi – Libya (2010)
8. Diab, M., Habash, N.: Arabic Dialect Processing Tutorial. In: Proceedings of the Human Language Technology Conference of the North American, Rochester (2007)
9. Khalfaoui, A.: A cognitive approach to analyzing demonstratives in Tunisian Arabic. In PhD thesis of university of Minnesota (November 2009)
10. Habash, N., Diab, M., Rambow, O.: Conventional Orthography for Dialectal Arabic. In: Proceedings of the Language Resources and Evaluation Conference (LREC), Istanbul (2012)
11. Zbib, R., Malchiodi, E., Devlin, J., Stallard, D., Matsoukas, S., Schwartz, R., Makhoul, J., Zaidany, O.F., Callison-Burch, C.: Machine Translation of Arabic Dialects. In: HLT-NAACL, pp. 49–59 (2012)
12. Clavier, V., Lallich-Boidin, G.: Modélisation linguistique de la suffixation en vue de l'analyse automatique. Traitement Automatique des Langues 35(2), 129–144 (1994)
13. Minker, W.: Stochastic versus rule-based speech understanding for information retrieval. Speech Communication, 223–247 (1998)
14. Wang, Y.-Y., Acero, A.: Discriminative models for spoken language understanding. In: ICSLP (2006)

15. Wang, Y., Acero, A., Mahajan, M., Lee, J.: Combining statistical and knowledge-based spoken language understanding in conditional models. In: Proceeding COLING/ACL, Sydney, Australia (2006)
16. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: International Conference on Machine Learning (ICML), pp. 282–289 (2001)
17. Kudo, T.: crf++, `http://chasen.org/~taku/software/CRF++/`
18. Graja, M., Jaoua, M., Belguith, L.H.: Towards Understanding Spoken Tunisian Dialect. In: Lu, B.-L., Zhang, L., Kwok, J. (eds.) ICONIP 2011, Part III. LNCS, vol. 7064, pp. 131–138. Springer, Heidelberg (2011)

Finding the Most Likely Upper Level State Sequence for Hierarchical HMMs

Akira Hayashi, Kazunori Iwata, and Nobuo Suematsu

Faculty of Information Sciences, Hiroshima City University, Japan
akira@hiroshima-cu.ac.jp

Abstract. Computing the most likely state sequence from an observation sequence is an important problem with many applications. The generalized Viterbi algorithm, a direct extension of the Viterbi algorithm for hidden Markov models (HMMs), has been used to find the most likely state sequence for hierarchical HMMs. However, the generalized Viterbi algorithm finds the most likely whole level state sequence rather than the most likely upper level state sequence. In this paper, we propose a marginalized Viterbi algorithm, which finds the most likely upper level state sequence by marginalizing lower level state sequences. We show experimentally that the marginalized Viterbi algorithm is more accurate than the generalized Viterbi algorithm in terms of upper level state sequence estimation.

1 Introduction

Hidden Markov models (HMMs) [6], known for their success in voice recognition, have been widely used to analyze time series data. Fine et al. [1] proposed hierarchical hidden Markov models (HHMMs) as a generalization of HMMs with a hierarchical state space. An HHMM may be represented using a tree structure, where each state at a non-leaf node, called an internal state, is itself a dynamical probabilistic model. Therefore, the internal states of an HHMM emit sequences rather than a single symbol. An HHMM generates sequences by recursive activation of a substate of an internal state, until a leaf node state, called a production state, is reached. Production states are the only states that actually output symbols through the usual HMM mechanism. The original inference algorithm for HHMMs is not efficient, taking $O(T^3)$ time where T is the length of the observation sequence. Murphy et al. [3] devised a dynamic Bayesian network (DBN) representation for HHMMs, thanks to which a linear time ($O(T)$) inference algorithm is now available.

HHMMs can naturally represent the multiple time scale structure of many time series data (for example, voice has three time scale structures: word sequence, phone sequence, and sub-phone sequence), and are gaining much attention in the research community. Some of the applications of HHMMs are hand written character recognition [1], information extraction from texts [8], and video analysis [5,4].

The problem of finding the most likely state sequence from an observation sequence [6] is important and has many applications. To find the most likely state sequence for HHMMs, the generalized Viterbi algorithm (GVA) [1,3], a direct extension of the Viterbi algorithm for HMMs [9,6], has been used. However, GVA finds the most likely whole level state sequence, but not the most likely upper level state sequence.

In this paper, we propose a marginalized Viterbi algorithm (MVA) to overcome the problem associated with GVA. MVA finds the most likely upper level state sequence by marginalizing lower level state sequences. For example, MVA will find the most likely sequence of "word" states in speech recognition by marginalizing the irrelevant "phone" and "sub-phone" state sequences, thus avoiding the problems associated with words having several pronunciations [2][1].

To explain our motivation for marginalizing irrelevant lower level states, consider the simple two level <u>static</u> hierarchical model in Fig. 1. The model can be seen as a Gaussian mixture speaker model for speaker identification [7], where the top level state, q^1, stands for a speaker s, and the second level state, q^2, stands for a component c of the Gaussian mixture model:

$$\begin{cases} p(q^2 = c | q^1 = s) = \pi_c^s \\ p(o = \mathbf{x} | q^1 = s, q^2 = c) = \mathcal{N}(\mathbf{x} | \mu_c^s, \mathbf{\Sigma}_c^s) \end{cases}, \tag{1}$$

where $\pi_c^s \geq 0$ is the weight of c and satisfies $\sum_c \pi_c^s = 1.0$, and $\mathcal{N}(\mathbf{x}|\mu_c^s, \mathbf{\Sigma}_c^s)$ is a Gaussian density with mean vector μ_c^s and covariance matrix $\mathbf{\Sigma}_c^s$. Given an observation $o = \mathbf{x}$, the most likely estimation for the speaker identification is

$$\hat{s} = \underset{s}{\text{argmax}}\, p(q^1 = s | o = \mathbf{x}), \tag{2}$$

where $p(q^1 = s|o = \mathbf{x})$ is obtained by

$$p(q^1 = s | o = \mathbf{x}) = \sum_c p(q^1 = s, q^2 = c | o = \mathbf{x}), \tag{3}$$

that is, by <u>marginalizing</u> q^2, an irrelevant second level state.

Our paper is organized as follows. We explain HHMMs in Section 2. We then explain GVA and MVA in Section 3. In Section 4, we compare the performances of GVA and MVA through experiments. We summarize the paper in Section 5.

2 HHMMs

2.1 Overview of HHMMs

An HHMM is represented as a tree structure as shown in Fig. 2. The circles, trapezoids, and rectangles in the figure stand for internal states, production states, and end states, respectively. The arrows connecting the states represent

[1] MVA cannot, however, find the most likely word sequence since it does not marginalize over word segmentation boundaries.

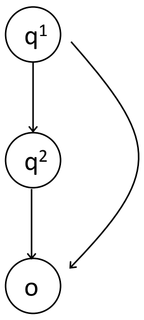


Fig. 1. Two level static hierarchical model

state transitions. A solid line indicates a horizontal transition to a state within the same level, a broken line indicates a vertical transition to a child state in the next level, and a dotted line indicates a transition to an end state, after which control is returned to the calling parent state. The state at the top of the hierarchy is called the root node. The level for the root node is 0, and a sequence of state transitions starts at the root state.

An HHMM generates a sequence of observations as follows.

(Step 1) Start: we start from the root node at time $t = 1$.
(Step 2) Vertical transition : a transition occurs from the current state (an internal state) to a child state in the lower level. If the destination is an internal state, further transitions to lower level states occur until a production state is reached.
(Step 3) Output symbol emission: the production state emits an output symbol o_t. Time t is incremented by 1.
(Step 4) Horizontal transition: a transition to a state within the same level occurs. If the destination is an internal state, we go back to Step 2, and if the destination is a production state, we go back to Step 3. If the destination is an end state, we proceed to Step 5.
(Step 5) Forced transition: A forced transition occurs to the upper level parent state which has initiated the current level state transitions, and we go back to Step 4.

Fine et al. [1], as well as proposing HHMMs, developed an algorithm for state estimation on the basis of the inside-outside algorithm. This algorithm is not efficient, however, and the time for state estimation and also for the most likely state sequence estimation is $O(T^3)$, where T is the length of the observation sequence.

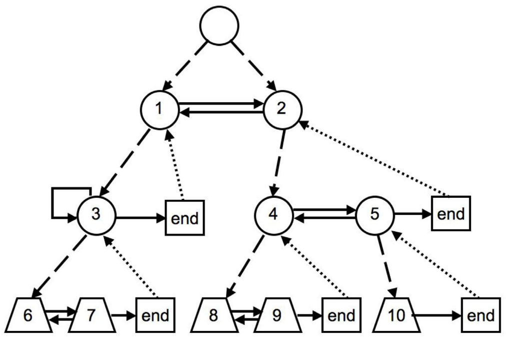


Fig. 2. Example of an HHMM with a three-level hierarchy

2.2 Representing HHMMs as DBNs

Murphy et al. [3] devised a dynamic Bayesian network (DBN) representation for HHMMs. A Bayesian network (BN) is a directed acyclic graph representing conditional independence relationships between random variables, and a DBN is an extension of a BN to a random process, where the random variables are dependent on time t. Thanks to the DBN representation, linear time ($O(T)$) algorithms for state estimation and the most likely state sequence estimation have become available.

We show a DBN representation of a three-level HHMM in Fig. 3. (We assume for simplicity that all production states are in the bottom level of the hierarchy.) The random variable o_t in the figure stands for the output from a production state at time t ($t = 1, \dots, T$). The output of an HHMM can be either discrete or continuous, but we consider the case of discrete symbol output in this paper. The state of the HHMM in level d and at time t is denoted by q_t^d ($d \in \{1, \dots, D\}$), where d is the hierarchy index: the top level has $d = 1$, and the bottom level has $d = D$.

f_t^d is an indicator variable which is equal to 1 if q_t^d has transitioned to its end state, and is 0 otherwise. The indicator variables play an important role in representing an HHMM as a DBN. As we explained in the previous subsection, a transition to an end state leads to a state transition in the upper level. In other words, $f_t^d = 1$ implies a possible state change in level $d-1$. In addition, if $f_t^d = 1$ then $f_t^{d'} = 1$ for all $d' > d$; hence the number of indicator variables that equal 0 denotes the level of the hierarchy we are currently in.

We now explain the state transition probabilities and the discrete output probabilities of an HHMM. The set of these probabilities constitutes the model parameters of an HHMM and completely defines the HHMM. Note that $f_t^d = 1$ implies not only that q_{t+1}^{d-1}, the state in level $d-1$ at time $t+1$, may change from q_t^{d-1} as mentioned above, but also that the value for q_{t+1}^d, the state in level d at time $t+1$, is determined by a vertical transition. We show below the state transition probabilities and the discrete output probabilities of an HHMM:

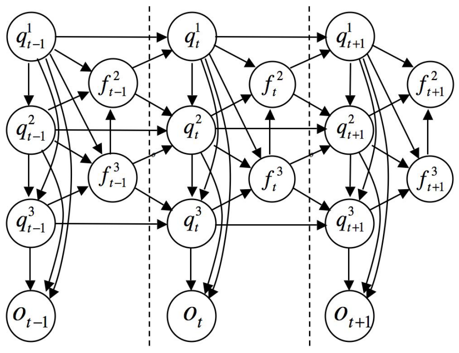


Fig. 3. DBN representation of a three-level HHMM, which draws state-transitions from time $t-1$ to $t+1$

$$p(q_t^d = j' | q_{t-1}^d = j, f_{t-1}^{d+1} = b, f_{t-1}^d = f, q_t^{1:d-1} = \boldsymbol{i}) = \begin{cases} \delta(j = j') & \text{if } b = 0, \\ A_{\boldsymbol{i}}^d(j, j') & \text{if } b = 1 \text{ and } f = 0, \\ \pi_{\boldsymbol{i}}^d(j') & \text{if } b = 1 \text{ and } f = 1, \end{cases} \tag{4}$$

$$p(f_t^d = 1 | q_t^d = j, q_t^{1:d-1} = \boldsymbol{i}, f_t^{d+1} = b) = \begin{cases} 0 & \text{if } b = 0, \\ Ae_{\boldsymbol{i}}^d(j) & \text{if } b = 1, \end{cases} \tag{5}$$

$$p(o_t = k | q_t^{1:D} = \boldsymbol{i}) = B_{\boldsymbol{i}}(k), \quad 1 \le k \le K, \tag{6}$$

where $q_t^{1:d} = (q_t^1, \ldots, q_t^d)$ is a vector consisting of the states in levels 1 through d at time t, and is denoted by $\boldsymbol{i}$ [2]. In Eq.(5), $d \ge 2$ is assumed. We assume that $f_0^1 = 1$ so that a state transition occurs at time $t = 1$. We also assume that $f_t^{D+1} = 1$ so that a state transition occurs in the bottom level at each time point.

$\delta(j = j')$ in Eq.(4) is 1 if $j = j'$, and is 0 if $j \ne j'$. If we assume that the vector of higher-up state variables at time t, $q_t^{1:d-1}$, is $\boldsymbol{i}$, then $A_{\boldsymbol{i}}^d(j, j')$ is the horizontal transition probability from state j to state j' in level d, $\pi_{\boldsymbol{i}}^d(j')$ is the vertical transition probability to state j' in level d, and $Ae_{\boldsymbol{i}}^d(j)$ is the horizontal transition probability in level d from state j to an end state. $B_{\boldsymbol{i}}(k)$ is the probability to output the k-th symbol when $q_t^{1:D}$ is $\boldsymbol{i}$. In Eq.(6), K is the number of output symbols.

3 Finding the Most Likely State Sequence

3.1 GVA

GVA is a direct extension of the Viterbi algorithm to HHMMs that finds the most likely sequence of states in all levels, including those in the lower levels.

[2] We suppose that $q_t^{1:d-1} = \boldsymbol{i}$ stands for the root node in level 0, when $d = 1$ in Eq.(4).

Let us define the symbols we use in explaining GVA. Let $Q^{1:D} = (q_1^{1:D}, \ldots, q_T^{1:D})$ be a sequence of state vectors, where $q_t^{1:D}(1 \le t \le T)$ is the vector of state variables from level 1 to level D at time t, and let $F^{2:D} = (f_1^{2:D}, ..., f_T^{2:D})$ be a sequence of indicator vectors, where $f_t^{2:D}(1 \le t \le T)$ is the vector of indicator variables from level 2 to level D at time t. Let $O = (o_1, ..., o_T)$ be a sequence of observations, where $o_t(1 \le t \le T)$ is the observation symbol at time t.

The most likely state sequence which GVA finds, $(\hat{Q}^{1:D}, \hat{F}^{2:D})$ is defined as follows:

$$(\hat{Q}^{1:D}, \hat{F}^{2:D}) \triangleq \operatorname*{argmax}_{Q^{1:D}, F^{2:D}} P(Q^{1:D}, F^{2:D} | O). \tag{7}$$

To find the most likely state sequence, $(\hat{Q}^{1:D}, \hat{F}^{2:D})$, given an observation sequence $O = \{o_1, ..., o_T\}$, we define $\delta_t(\boldsymbol{q}, \boldsymbol{f})$ as follows:

$$\delta_t(\boldsymbol{q}, \boldsymbol{f}) \triangleq \max_{q_{1:t-1}^{1:D}, f_{1:t-1}^{2:D}} \log P\big(q_{1:t-1}^{1:D}, f_{1:t-1}^{2:D}, q_t^{1:D} = \boldsymbol{q}, f_t^{2:D} = \boldsymbol{f}, o_1, o_2, \cdots, o_t\big). \tag{8}$$

$\delta_t(\boldsymbol{q}, \boldsymbol{f})$ is the log probability of the most likely state sequence that starts from the initial state, emits $o_1, ..., o_t$, and ends at time t in state $(\boldsymbol{q}, \boldsymbol{f})$. By induction, we can rewrite Eq.(8) as

$$\delta_t(\boldsymbol{q}, \boldsymbol{f}) = \max_{\boldsymbol{q}', \boldsymbol{f}'} \{ \delta_{t-1}(\boldsymbol{q}', \boldsymbol{f}') + \log P(q_t^{1:D} = \boldsymbol{q}, f_t^{2:D} = \boldsymbol{f} \mid q_{t-1}^{1:D} = \boldsymbol{q}', f_{t-1}^{2:D} = \boldsymbol{f}') \} + \log P(o_t \mid q_t^{1:D} = \boldsymbol{q}). \tag{9}$$

To actually retrieve the state sequence, we keep track of $\boldsymbol{q}', \boldsymbol{f}'$ which maximizes the right hand side of Eq.(9) for each state $(\boldsymbol{q}, \boldsymbol{f})$ at each time $t \ge 2$. We do this via the array $\phi_t(\boldsymbol{q}, \boldsymbol{f})$. The whole procedure for finding the most likely state sequence can now be stated as follows.

(Step 1) **Initialization:** for $t = 1$,

$$\delta_1(\boldsymbol{q}, \boldsymbol{f}) = \log P(q_1^{1:D} = \boldsymbol{q}, f_1^{2:D} = \boldsymbol{f}) + \log P(o_1 | q_1^{1:D} = \boldsymbol{q}), \qquad \forall \boldsymbol{q}, \forall \boldsymbol{f}. \tag{10}$$

(Step 2) **Recursion:** for $t = 2, ..., T$,

$$\delta_t(\boldsymbol{q}, \boldsymbol{f}) = \max_{\boldsymbol{q}', \boldsymbol{f}'} \{ \delta_{t-1}(\boldsymbol{q}', \boldsymbol{f}') + \log P(q_t^{1:D} = \boldsymbol{q}, f_t^{2:D} = \boldsymbol{f} \mid q_{t-1}^{1:D} = \boldsymbol{q}', f_{t-1}^{2:D} = \boldsymbol{f}') \} + \log P(o_t \mid q_t^{1:D} = \boldsymbol{q}), \quad \forall \boldsymbol{q}, \forall \boldsymbol{f}, \tag{11}$$

$$\phi_t(\boldsymbol{q}, \boldsymbol{f}) = \operatorname*{argmax}_{\boldsymbol{q}', \boldsymbol{f}'} \{ \delta_{t-1}(\boldsymbol{q}', \boldsymbol{f}') + \log P(q_t^{1:D} = \boldsymbol{q}, f_t^{2:D} = \boldsymbol{f} \mid q_{t-1}^{1:D} = \boldsymbol{q}', f_{t-1}^{2:D} = \boldsymbol{f}') \} + \log P(o_t \mid q_t^{1:D} = \boldsymbol{q}), \quad \forall \boldsymbol{q}, \forall \boldsymbol{f}. \tag{12}$$

(Step 3) **Termination:**

$$\log \hat{P} = \max_{\boldsymbol{q}, \boldsymbol{f}} \delta_T(\boldsymbol{q}, \boldsymbol{f}), \tag{13}$$

$$(\hat{q}_T^{1:D}, \hat{f}_T^{2:D}) = \operatorname*{argmax}_{\boldsymbol{q}, \boldsymbol{f}} \delta_T(\boldsymbol{q}, \boldsymbol{f}). \tag{14}$$

(Step 4) **Path (state sequence) backtracking:**
for $t = T-1, T-2, ..., 1$,

$$(\hat{q}_t^{1:D}, \hat{f}_t^{2:D}) = \phi_{t+1}(\hat{q}_{t+1}^{1:D}, \hat{f}_{t+1}^{2:D}). \tag{15}$$

The time complexity of GVA is $O(T)$.

3.2 MVA for Two-Level HHMMs

In hierarchical models, upper level states usually convey more important information. MVA finds the most likely upper level state sequence by marginalizing the lower level state sequences. In this subsection, we explain MVA for two-level HHMMs for simplicity. We skip explaining MVA for general D-level HHMMs because of the space limitation.

Let $Q^1 = q^1_{1:T}$, $Q^2 = q^2_{1:T}$, and let $F^2 = f^2_{1:T}$. The most likely upper level state sequence, $(\hat{Q}^1, \hat{F}^2)$, is defined as follows [3]:

$$(\hat{Q}^1, \hat{F}^2) \triangleq \operatorname*{argmax}_{Q^1, F^2} P(Q^1, F^2, O) = \operatorname*{argmax}_{Q^1, F^2} \sum_{Q^2} P(Q^{1:2}, F^2, O). \tag{16}$$

Before explaining MVA, let us define segments. Suppose we are given $\{Q^1 = q^1_{1:T}, F^2 = f^2_{1:T}, Q^2 = q^2_{1:T}\}$, a state sequence from time 1 to time T. We call $\{q^1_{t_1:t_2}, f^2_{t_1:t_2}, q^2_{t_1:t_2}\}$, which is a partial sequence of states between time t_1 and t_2, a segment with t_1, t_2 as the start time and the end time for the segment, when $\{q^1_{t_1:t_2}, f^2_{t_1:t_2}, q^2_{t_1:t_2}\}$ satisfies the following conditions:

$$\begin{cases} t_1 = 1 \text{ or } f^2_{t_1-1} = 1, \\ f^2_{t_1:t_2-1} = 0 \text{ if } \ t_2 - t_1 \geq 2, \\ f^2_{t_2} = 1. \end{cases} \tag{17}$$

Simply speaking, a segment is a partial sequence of states between the time just after the state in level 2 transitions to its end state and the time when the state in level 2 transitions to its end state again. See Fig. 4. Note that level 1 state transitions occur only at segment boundaries, and that the state in level 1 does not change within a segment (i.e., between the start and the end time of a segment). Therefore, the level 1 state sequence from time 1 to time T can be specified by the end times (or the start times) of all the segments and the level 1 state for each of the segments.

Computing $\{\delta_t(i) \mid 1 \leq i \leq N^1, 1 \leq t \leq T\}$ defined below plays a central role in MVA:

$$\begin{aligned} \delta_t(i) &\triangleq \max_{q^1_{1:t-1}, f^2_{1:t-1}} \log P(q^1_{1:t-1}, f^2_{1:t-1}, q^1_t = i, f^2_t = 1, o_{1:t}) \\ &= \max_{q^1_{1:t-1}, f^2_{1:t-1}} \log \sum_{q^2_{1:t}} P(q^1_{1:t-1}, f^2_{1:t-1}, q^2_{1:t-1}, q^1_t = i, f^2_t = 1, q^2_t, o_{1:t}), \end{aligned} \tag{18}$$

[3] For some applications, we may want to marginalize the indicator variable sequences, F^2, as well.

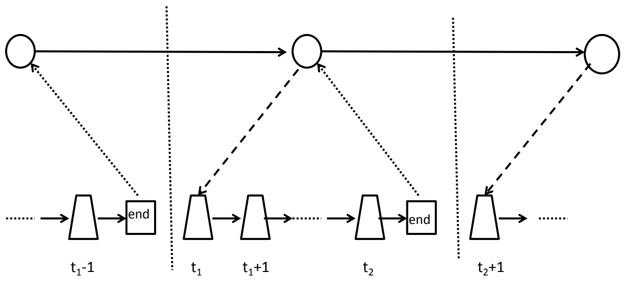


Fig. 4. Segment that starts at t_1 and ends at t_2

where N^1 is the total number of level 1 states. $\delta_t(i)$ is the log probability of the most likely level 1 state sequence which starts from the initial state, emits $o_1, ..., o_t$, and ends at time t, when the level 1 state is i and the level 2 state transitions to its end state (i.e. t is an end time of a segment).

Given an observation sequence, O, and the indicator variable at time T, $f_T^2 = \tilde{f}_T^2$, the most likely level 1 state sequence can be found by the following dynamic program. We assume $\tilde{f}_T^2 = 1$ for simplicity.

(Step 1) **Initialization:** for $t = 1$,

$$\delta_1(i) = \log P(q_1^1 = i, f_1^2 = 1, o_1), \qquad \forall i, \tag{19}$$

$$\phi_1(i) = (0, 0), \qquad \forall i. \tag{20}$$

(Step 2) **Recursion:** for $t = 2, ..., T$,

$$\delta_t(i) = \max\left\{\alpha_{0,t}(i), \max_{j,\tau:1\le\tau<t}\left(\delta_\tau(j) + \log A_0^1(j,i) + \alpha_{\tau,t}(i)\right)\right\}, \ \forall i, \tag{21}$$

$$\phi_t(i) = \begin{cases} (0,0) & \text{if } \delta_t(i) = \alpha_{0,t}(i), \\ (j^*, \tau^*) = \mathrm{argmax}_{j,\tau:1\le\tau<t}(\delta_\tau(j) + \log A_0^1(j,i) + \alpha_{\tau,t}(i)) & \text{otherwise}, \end{cases}$$
$$\forall i, \tag{22}$$

where

$$\alpha_{\tau,t}(i) = \log P(f_{\tau+1:t-1}^2 = 0, f_t^2 = 1, o_{\tau+1:t} | f_\tau^2 = 1, q_{\tau+1}^1 = i)$$
$$= \log \sum_{q_{\tau+1:t}^2} P(f_{\tau+1:t-1}^2 = 0, f_t^2 = 1, q_{\tau+1:t}^2, o_{\tau+1:t} \mid f_\tau^2 = 1, q_{\tau+1}^1 = i),$$
$$\tau \ge 1,$$
$$\alpha_{0,t}(i) = \log P(q_{1:t}^1 = i, f_{1:t-1}^2 = 0, f_t^2 = 1, o_{1:t}).$$

(Step 3) **Backtracking**

$$\hat{q}_T^1 = \operatorname*{argmax}_i \delta_T(i) \ , \hat{f}_T^2 = 1, t = T. \tag{23}$$

```
while $t > 0$ do
1 ) $(\hat{q}^1_\tau, \tau) = \phi_t(\hat{q}^1_t), \hat{f}^2_\tau = 1$
$\hat{q}^1_{\tau+1:t-1} = \hat{q}^1_t, \hat{f}^2_{\tau+1:t-1} = 0$  if  $\tau + 1 \leq t - 1$
2 ) $t \leftarrow \tau$
endwhile
```

In the above, $A^1_0(\cdot,\cdot)$ is the horizontal transition probability for the level 1 states, the subscript τ in $\alpha_{\tau,t}(i)$ is the end time of the segment just before the segment which ends at time t, and $\alpha_{\tau,t}(i)$ is the log probability that the subsequence of observations $o_{\tau+1:t}$ is emitted by any segment whose level 1 state is i. That is, $\alpha_{\tau,t}(i)$ is the marginalization over the lower level state sequence of the joint log probability that $o_{\tau+1:t}$ is emitted by a lower level state sequence generated by the upper level state i. All of $\{\alpha_{\tau,t}(i) | 1 \leq i \leq N^1, 1 \leq \tau < t\}$ can be computed by the backward procedure for HMMs in $O(t)$ total time[6]. $\phi_t(i)$ is a variable used for backtracking, containing both the level 1 state, j, of the previous segment and the end time, τ, for the previous segment.

While the time complexity of GVA is $O(T)$, the time complexity of MVA is $O(T^2)$. This is the cost we have to pay to find the most likely level 1 state sequence.

4 Experiments

We carry out two experiments. In Experiment 1, we compare GVA and MVA in terms of accuracy. In Experiment 2, we determine when MVA is much more accurate than GVA.

4.1 Overview of the Experiments

Experimental Data. Artificial data to be used in the experiments are randomly generated from two-level HHMMs. The initial state probabilities, $\{\pi^1_0(i) \mid 1 \leq i \leq N^1\}$ (N^1 is the total number of states at level 1), the state transition probabilities, $\{A^1_0(i,i') \mid 1 \leq i,i' \leq N^1\}$, for level 1, the initial state probabilities, $\{\pi^2_i(j) \mid 1 \leq i \leq N^1, 1 \leq j \leq N^2\}$ (N^2 is the total number of states in level 2), and the state transition probabilities, $\{A^2_i(j,j') \mid 1 \leq i \leq N^1, 1 \leq j,j' \leq N^2\}$, are all sampled from Dirichlet distributions with concentration parameters α_k equal to 1.0. The state ending probabilities, $\{A^2_i(j, end) \mid 1 \leq i \leq N^1, 1 \leq j \leq N^2\}$, are all 0.1. The output probabilities, $\{B_{(i,j)}(k) \mid 1 \leq i \leq N^1, 1 \leq j \leq N^2, 1 \leq k \leq K\}$, are also sampled from Dirichlet distributions with concentration parameters α_k equal to 1.0.

A concentration parameter of 1.0 in the Dirichlet distribution results in all sets of probabilities being equally likely (i.e. a uniform distribution). The state ending probabilities are set to 0.1 to make the segments longer and make upper level state estimation easier.

Performance Evaluation We estimate the upper level state sequences[4] using GVA and MVA. The performance of the algorithms is evaluated in terms of two accuracy rates:

- Accuracy rate 1 is computed from the sequences:

$$\text{accuracy rate 1} \triangleq \frac{N_{seq}^{corr}}{N_{seq}},$$

  where N_{seq} is the total number of sequences, and N_{seq}^{corr} is the total number of sequences for which the upper level states are always correctly estimated.
- Accuracy rate 2 is computed from the times:

$$\text{accuracy rate 2} \triangleq \frac{N_{times}^{corr}}{N_{times}},$$

  where N_{times} is the sum of the sequence lengths, and N_{times}^{corr} is the sum of the times at which the upper level states are correctly estimated.

4.2 Experiment 1

We set $N^1 = 2$. Let λ and λ' be the lower level models (considered as HMMs) whose parent states are states 1 and 2 in level 1, respectively. In Experiment 1, we consider the case where λ and λ' have the same number of states.

- N_λ and $N_{\lambda'}$, the total number of states in λ and λ', are both four.
- K is $2, 4, 6, 8$ or 10.
- We generate 100 sequences of length $T = 10$, $T = 20$, and $T = 50$ for each trial.
- The accuracy rates are averaged over 100 trials.

We show the results in Table 1. MVA outperforms GVA in terms of both accuracy rate 1 and accuracy rate 2. For both algorithms, accuracy rate 1 decreases as T becomes longer. Accuracy rates 1 and 2 both increase as K becomes larger. The reason is as follows. Since the symbol output probabilities are sampled from Dirichlet distributions whose concentration parameters are 0.1, the probabilities are close to 0 or 1, and therefore it becomes easier to estimate states from observations when there are more output symbols.

4.3 Experiment 2

We set $N^1 = 2$, as in Experiment 1. In Experiment 2, we consider the case where λ and λ' have different numbers of states. N_λ and $N_{\lambda'}$ are eight and four, respectively. All other settings are the same as in Experiment 1.

[4] An upper level state sequence can be either $q^1_{1:T}$ or $(q^1_{1:T}, f^2_{1:T})$, but we use the latter, $(q^1_{1:T}, f^2_{1:T})$, as an upper level state sequence.

Table 1. Results of Experiment 1 ($N^1 = 2, N_\lambda = N_{\lambda'} = 4$, the numbers in parentheses are standard deviations)

		Accuracy Rate 1 (%)				Accuracy Rate 2 (%)			
K	T	GVA		MVA		GVA		MVA	
2	10	30.6	(7.2)	**32.7**	(6.2)	64.8	(11.9)	**69.1**	(8.9)
	20	9.4	(4.3)	**11.5**	(3.4)	56.0	(15.4)	**64.7**	(11.3)
	50	0.5	(0.7)	**0.6**	(0.8)	58.1	(16.6)	**65.1**	(12.7)
4	10	31.7	(6.5)	**34.0**	(6.4)	68.3	(8.8)	**71.1**	(8.1)
	20	11.9	(4.2)	**13.4**	(3.7)	66.6	(12.8)	**72.1**	(9.2)
	50	0.4	(0.6)	**0.7**	(0.8)	64.6	(12.6)	**71.0**	(8.2)
6	10	32.1	(5.9)	**34.2**	(5.4)	68.7	(8.2)	**71.2**	(7.3)
	20	12.3	(3.5)	**13.3**	(3.7)	70.0	(9.3)	**72.6**	(8.2)
	50	0.8	(1.0)	**0.9**	(0.9)	69.4	(12.1)	**73.8**	(8.0)
8	10	34.1	(6.2)	**36.0**	(5.6)	70.4	(7.8)	**73.2**	(6.7)
	20	12.3	(3.9)	**13.7**	(3.6)	69.4	(8.0)	**72.7**	(6.1)
	50	0.6	(0.7)	**0.7**	(0.9)	70.0	(9.0)	**73.8**	(7.5)
10	10	33.5	(5.1)	**35.3**	(5.0)	71.2	(7.3)	**73.5**	(6.5)
	20	12.4	(3.7)	**13.3**	(3.9)	71.1	(7.6)	**73.3**	(6.6)
	50	0.7	(0.9)	**0.8**	(1.0)	72.3	(8.4)	**75.9**	(6.7)

Table 2. Results of Experiment 2 ($N_\lambda = 8, N_{\lambda'} = 4$, the numbers in parentheses are standard deviations)

		Accuracy Rate 1 (%)					
K	T	GVA			MVA		
2	10	25.3	(10.2)	-5.3	**32.3**	(7.0)	-0.4
	20	7.5	(4.4)	-1.9	**11.4**	(3.7)	-0.2
	50	0.4	(0.6)	-0.1	**0.6**	(0.8)	0.0
4	10	26.5	(8.0)	-5.3	**33.8**	(5.1)	-0.2
	20	9.1	(4.7)	-2.7	**13.0**	(3.8)	-0.4
	50	0.3	(0.6)	-0.1	**0.5**	(0.7)	-0.2
6	10	28.4	(8.1)	-3.8	**34.5**	(5.7)	+0.3
	20	9.2	(4.4)	-3.0	**13.0**	(3.7)	-0.4
	50	0.4	(0.7)	-0.4	**0.7**	(0.8)	-0.2
8	10	29.8	(7.7)	-4.3	**35.1**	(4.9)	-0.9
	20	10.2	(4.7)	-2.1	**14.5**	(4.5)	+0.8
	50	0.4	(0.6)	-0.2	**0.7**	(0.8)	0.0
10	10	28.6	(7.1)	-4.9	**34.4**	(5.7)	-0.9
	20	9.9	(4.1)	-2.5	**13.3**	(3.3)	0.0
	50	0.4	(0.7)	-0.3	**0.9**	(0.9)	+0.1

We show the results (accuracy rate 1) of Experiment 2 in Table 2. The last figures in each boxes show the differences from the accuracy rates in Experiment 1. When we compare the results with those from Experiment 1, we notice that while the accuracy rates of GVA are lower, the accuracy rates of MVA do not change very much. We conjecture that the reason is as follows. Since GVA finds the most likely whole level state sequence including the lower level states, a model which has fewer states (λ'), i.e. the upper level state 2, is more likely to be chosen by the algorithm because of the (possibly) larger transition probabilities in level 2. On the other hand, MVA, which marginalizes the lower level state, does not have this problem. This is why there is a big difference in the accuracy rates of the two algorithms.

5 Conclusions

The generalized Viterbi algorithm, a direct extension of the Viterbi algorithm for HMMs, has been used to find the most likely state sequence for hierarchical HMMs. However, the generalized Viterbi algorithm finds the most likely whole level state sequence, but not the most likely upper level state sequence. In this paper, we have proposed a marginalized Viterbi algorithm that finds the most likely upper level state sequence by marginalizing the lower level state sequences. Using experiments, we have shown that the marginalized Viterbi algorithm is more accurate than the generalized Viterbi algorithm in terms of upper level state sequence estimation.

References

1. Fine, S., Singer, Y., Tishby, N.: The hierarchical hidden Markov model: Analysis and applications. Machine Learning 32(1), 41–62 (1998)
2. Jurafsky, D., Martin, J.H.: Speeach and Language Processing, 2nd edn. Prentice Hall (2008)
3. Murphy, K., Paskin, M.: Linear time inference in hierarchical HMMs. Advances in Neural Information Processing Systems 14, 833–840 (2001)
4. Nguyen, N.T., Phung, D.Q., Venkatesh, S., Bui, H.: Learning and detecting activities from movement trajectories using the hierarchical hidden Markov models. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 955–960 (2005)
5. Phung, D.Q., Duong, T.V., Venkatesh, S., Bui, H.H.: Topic transition detection using hierarchical hidden Markov and semi-Markov models. In: Proceedings of 13th ACM International Conference on Multimedia, pp. 11–20 (2005)
6. Rabiner, L., Juang, B.H.: Fundamentals of Speech Recognition. Prentice Hall (1993)
7. Reynolds, D.A.: Speaker identification and verification using gaussian mixture speaker models. Speech Communication 17, 91–108 (1995)
8. Skounakis, M., Craven, M., Ray, S.: Hierarchical hidden Markov models for information extraction. In: Proceedings of 18th International Joint Conference on Artificial Intelligence, pp. 427–433 (2003)
9. Viterbi, A.: Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. IEEE Transactions on Information Theory 13(2), 260–269 (1967)

Generalization of Discriminative Approaches for Speech Language Understanding in a Multilingual Context

Bassam Jabaian[1], Fabrice Lefèvre[1], and Laurent Besacier[2]

[1] LIA, University of Avignon, Avignon, France
{bassam.jabaian,fabrice.lefevre}@univ-avignon.fr
[2] LIG, University Joseph Fourrier, Grenoble, France
laurent.besacier@imag.fr

Abstract. Probabilistic approaches are now widespread in the various applications of natural language processing and elicitation of a particular approach usually depends on the task at hand. Targeting multilingual interpretation of speech, this paper presents a comparison between the state-of-the-art methods used for machine translation and speech understanding. This comparison justifies our proposition of a unified framework to perform a joint decoding which translates a sentence and assigns semantic tags to this translation in the same process. The decoding is achieved using a cascade of finite-state transducers allowing to compose translation and understanding hypothesis graphs. This representation is favorable as it can be generalized to allow rich transmission of information between the components of a human-machine vocal interface.

Keywords: Multilingual Understanding, CRF, Hypothesis Graphs.

1 Introduction

Today, probabilistic approaches are widely used in all applications of automatic language processing (speech recognition, machine translation, syntactic parsing, POS or semantic tagging etc.). The performance of an approach depends greatly on the targeted task. Considering various tasks, the best performing approach is not always the same.

For instance, for the Spoken Language Understanding (SLU) task, Conditional Random Fields (CRF) [15] have been shown to be the most efficient so far [8]. Whereas for machine translation, Log-Linear Stochastic Phrase-Based Machine Translation (LLPB-SMT) [14] are the most commonly used and have shown their potential for many language pairs.

However, despite the initial formulations between the probabilistic approaches, distinctions tend to fade away when confronted with practical considerations. Also some works proposed the use of discriminative approaches, such as CRF, for automatic translation [23,18,17] while at the same time the phrase-based translation pipeline was also investigated for other natural language processing tasks such as grapheme-phoneme conversion [25], Part-of-Speech tagging [7] etc.

In this paper, our overall objective is to develop performing approaches for speech understanding in a multilingual context. For that matter two state-of-the-art approaches are investigated for each of the underlying issues: CRF-SLU for speech understanding and LLPB-SMT for translation. In a first step, to evaluate the practical interest of each method, we propose to use and optimize the LLPB-SMT approach for speech understanding, and also to integrate a CRF-based model in a machine translation module. This study allows to highlight the specificities of each task and to evaluate the performance of the respective approaches for these tasks.

On the other hand, we showed in a previous work [11,12] that the use of machine translation is an effective solution for the portability of an understanding system from a language to another. For instance, this portability can be obtained by cascading a translation module with an understanding system. In few words, the idea here is to translate the inputs of a user into a language for which we already have a (hopefully) performing understanding system.

In some cases, the best translation hypothesis is not the one for which the system generates the best understanding (from our experience it is most often caused by bad word reordering). Therefore the selection of the best translation does not optimize the overall system in a scenario of multilingual understanding.

Based on the comparison made between both tasks we propose a model that can jointly decode the inputs in terms of translation and understanding hypotheses. This joint decoding will select a translation taking into account the semantic tagging generated for this translation. In this line of thought, we no longer seek the best possible translation, but the translation that can be semantically labelled in the best possible way.

Our experiments are based on the French dialogue corpus MEDIA based on which an initial understanding system for French can be built. In order to use this system for Italian entry tagging, an Italian to French translation system is also trained with few manual data. This system will be used during decoding to translate Italian entries into French in order to provide French inputs to the understanding system. Then these models will be merged in a single decoding loop by means of hypothesis graphs.

The paper is organized as follows: Section 2 presents the use of a machine translation approach for speech understanding. Then Section 3 describes the use of CRF for machine translation. Our proposal for a joint decoding between translation and understanding is presented in Section 4. Finally, Section 5 gives an overview of the experimental study and results.

2 Using a Machine Translation Approach for Spoken Language Understanding

The understanding of a user utterance can be seen as a translation from a sequence of words (source language) to a sequence of concepts (target language). [20,19] showed that statistical machine translation approaches can be used in a task of speech understanding. This approach assumes that the sequences of concepts are the translations of the original sequences of words.

Despite this similarity between the tasks of understanding and translation, understanding has its own characteristics that must be taken into consideration in order to improve the performance obtained by a LLPB-SMT approach.

The differences between a classical translation task (from a natural language into another) and the use of translation for understanding (translation from a language into semantic tags) can be summarized as follows:

- the semantics of a sentence generally follows the order in which words occur unlike a translation task in which translated words may have a different order between target and source depending on the language pair considered and their syntactic proximity;
- in a translation task, a source word can be aligned to no target word (null fertility), whereas for understanding every word must be aligned to a concept, even though words that do not contribute to the meaning of the sentence considering the domain of the task are labelled with a specific `NULL` concept;
- also, evaluation measures are different between both tasks (BLEU [24] for translation vs. CER - Concept Error Rate - for understanding), therefore the tools used for optimizing translation systems should be adapted to optimize CER score instead of BLEU.

Following the assumption that the semantics of a sentence follows the order in which words occurred, we propose to introduce a constraint of monotony, which conducts the decoder to follow strictly the order of words to generate concepts.

A major difficulty of the translation task is that it requires to automatically align a word from the source language to its corresponding word in the target language. Since corpora used for training translation systems are usually aligned at the sentence level, an automatic alignment step is necessary to obtain word alignment. However, most of understanding corpora are labelled (aligned) at the segment level and therefore the use of alignment information can be beneficial to help the alignment process. In this respect, the use of the BIO tagging (Begin Inside Outside) [26] ensures that each word in the source sentence is aligned to its corresponding concept and therefore no additional automatic alignment is required. In that way, the extraction of the phrase table is obtained from a corpus with a perfect alignment (no alignment errors).

Finally, since we want to evaluate the hypotheses generated by this approach from an understanding perspective (evaluating the CER and not the BLEU score) we propose to modify the MERT - Minimum Error rate Training [22] - algorithm to maximize the CER directly.

3 Using a Spoken Language Understanding Method for Machine Translation

In this approach, the translation is viewed as a tagging of the source words sequences, the possible tags being the words of the target language themselves. Training a tagger based on a CRF approach requires an annotated corpus (translated corpus) at word level. The application of IBM models [4] provides automatic word alignments from a bilingual corpus originally aligned at the sentence level. As for the understanding task, where many words may be associated with a single concept, several source words can be aligned with only one target word. Thus, we proposed to handle tags as it is done for understanding using the BIO formalism. For example, the French sequence "Je voudrais" aligned to the Italian word "vorrei" can be represented as: "je, B_vorrei" "voudrais, I_vorrei".

The main difficulty for training CRF models for translation is related to the high number of tags (corresponding to the target language vocabulary size). Though some solutions exist, [27] proposed to use the RPROP algorithm for feature optimization to deal with a large number of features. This algorithm reduces the memory requirements compared to other optimization algorithms [32].

Another limitation to the use of CRF for translation is that it does not take into account word reordering and that the target language model is limited by the computational complexity of the decoding. To obtain an efficient CRF-based translation system, [17] have proposed a model based on finite state transducers in which different stages of the translation process are composed. This model will be called CRFPB-SMT because it embeds a mechanism for modelling a translation table by sub-sentential segments (called tuples, but analogous to phrases) and uses CRF as the probabilistic models providing the hypothesis scores.

The proposed decoder is a composition of Weighted Finite State Transducers (WFST) representing the following steps: reordering and segmentation of the source sentence according to the words tuples , application of the translation model with hypotheses evaluation based on CRF, and composition with a target language model.

This architecture allows to consider the translation of a sentence as a composition of transducers in the following order:

$$\lambda_{translation} = \lambda_S \circ \lambda_R \circ \lambda_T \circ \lambda_F \circ \lambda_L$$

where λ_S is the acceptor of the source sentence s, λ_R implements segmentation and reordering roles, λ_T is a dictionary of tuples, combining sequences of the source language and their possible translation based on the tuples inventory, λ_F is a feature matcher, which assigns probability scores to tuples using a CRF model and λ_L is a language model of the target language.

4 Language Portability Scenario: Joint Decoding for Translation and Understanding

4.1 Our Language Portability Scenario

Our study of the relations between the different approaches was mainly aimed at being able to combine them for multilingual portability of an understanding system. In a previous work [11], we have shown that the best way to port an understanding system to a new language is also the simplest: to translate users utterances of the new language back into the language of the existing understanding system and then to label the translation hypothesis with this system.

This proposition is based on a pipeline of a translation system (LLPB-SMT) and an understanding system (CRF-SLU). The best hypothesis generated by the translation system is conveyed as input to the understanding system. However, other hypotheses can be better interpreted by the tagger. So, the selection of the best translation does not necessarily optimize the behavior of the overall system.

A joint decoding between translation and understanding can be an efficient solution to address this problem. The joint decoding has the advantage of being able to optimize the selection of the translation taking into account the tags that can be assigned to the best possible translations.

The previously presented CRFPB-SMT approach can be generalized to the understanding task. Therefore an understanding system $\lambda_{understanding}$ can be obtained in the same way as proposed in Section 3. This representation allows to obtain a graph of understanding hypotheses similar to that obtained for translation. Since the translation output vocabulary is the same as for the understanding input, these two graphs can be composed to derive a joint graph:

$$\lambda_{joint} = \lambda_{translation} \circ \lambda_{understanding}$$

This composition takes a sentence in the target language as input and assigns a sequence of concepts to that sentence using a semantic tagger available for the source language. This consists in a joint decoding between the translation and the understanding since the probabilities of the two models are taken into account to determine the best overall hypothesis.

4.2 Related Works

The joint decoding issue has already been addressed in the past, mostly when system component pipelines are involved for human-machine interaction systems. In a standard architecture, the system transmits the best transcription hypothesis from the automatic speech recognition system to the speech understanding module. Considering that this hypothesis is noisy, it is not necessarily the best labelled one.

Several studies have proposed a joint decoding between speech recognition and understanding to take into account the n-best recognition hypotheses during the semantic tagging. These early works [31,29,9] have proposed to produce a confusion network out of several recognition graph outputs. The understanding system was represented as a WFST, which weights were obtained by maximum likelihood estimates on the training data. Then joint decoding is obtained as the composition of the recognition graph with the understanding graph.

The positive results obtained by these proposals have encouraged further work in the same line. Given that the most successful models in the SLU state-of-the-art are CRF, [2] proposed to use them instead of WFST for understanding. In the same line, our proposal seeks a joint decoding for translation and understanding. Since the two systems are different by nature, their joint optimization is made difficult, this is why we try to standardize systems over considered tasks.

5 Experiments and Results

All experiments presented in this paper are based on the MEDIA French dialogue corpus. MEDIA (described in [3]) covers the domain of hotel reservation and tourist information. This corpus is annotated with 99 semantic labels that represent the domain semantics.

The corpus is composed of 1257 dialogues grouped into 3 parts: a training set (13k sentences), a development set (1.3k sentences) and a test set (3.5k sentences). A subset of the training data (about 5.6k sentences), as well as the test set and the development set, are manually translated into Italian.

A LLPB-SMT based model is used to train an understanding system on the French corpus, and the manually translated subset of this corpus is used as a parallel corpus to train a translation model based on CRF. Then the CRFPB-SMT approach based on transducers is evaluated separately for translation and understanding before being used in the context of joint decoding.

The Concept Error Rate (CER) is the evaluation criterion used to evaluate the understanding task. CER can be defined as the ratio of the sum of concepts deleted, substituted and inserted on the number of concepts in the reference. On the other hand the BLEU score [24], based on the accounts of n-grams shared between hypothesis and reference, is used to evaluate the translation task.

5.1 Evaluation of the LLPB-SMT Model for Understanding

The MOSES toolkit [13] is used to train a LLPB-SMT for French language understanding. The first attempts showed clearly inferior performance to those obtained by a CRF-SLU baseline model (CER 23.2% after MERT tuning for LLPB-SMT compared to 12.9% for CRF-SLU [1]). Incremental improvements of the model as proposed in Section 2 are evaluated in Table 1.

Using a monotone decoding allows a reduction of 0.5% absolute in CER. Rewriting the training data into the BIO formalism reduces significantly the CER (2.4% absolute). Optimizing the CER instead of BLEU reduces the CER of an extra 0.4%. Finally, adding a list of cities in the training set considerably addresses the OOV word issue and provides a final CER reduction of 0.8%.

Results show that despite all the improvements on the LLPB-SMT approach, the CRF based approach still performs better for the understanding task (CER 12.9% for CRF-SLU vs 18.3% for LLPB-SMT). The analysis of the errors made by each model shows that CRF have a high level of deletions compared to the other types of errors, while the LLPB-SMT method presents a better trade-off between deletion and insertion errors, even though it ends up with a higher CER.

5.2 Evaluation of the CRF Model for Translation

In order to evaluate our approach using a CRF-SLU model for the translation we use the manually translated part of MEDIA as parallel corpus to train the translation model (French to Italian). The GIZA++ toolkit (available with MOSES)

[1] Please refer to [12] for more details on the CRF-SLU module.

Table 1. Incremental improvements of LLPB-SMT model for French understanding

Models	**Sub**	**Del**	**Ins**	**CER**
Initial	5.4	4.1	14.6	24.1
+MERT (BLEU)	5.6	8.4	9.2	23.2
+Monotone decoding	6.2	7.8	8.7	22.7
+BIO formalism	5.7	9.3	5.3	20.3
MERT (CER)	5.3	9.2	4.6	19.1
OOV list	5.8	7.4	5.1	18.3

was used to automatically obtain word-to-word alignment between the two languages and WAPITI [16] was used to train the CRF model.

We trained a CRF-SLU model for translation using the RPROP algorithm as proposed in Section 3. Characteristic functions of 5-grams on the observations and bi-grams on labels are used to train the model. The performances obtained are presented in Table 2. The results show that the performance of the CRF-SLU model (BLEU 42.5) is significantly worse than the performance obtained by the LLPB-SMT method using MOSES with basic settings (BLEU 47.2) [2].

To make a fair comparison between the two methods, we evaluate the LLPB-SMT approach in the same conditions as the CRF-SLU approach. The LLPB-SMT method uses a reordering model while CRF-SLU, dedicated to sequential labelling, does not include such model. For that we added a monotony constraint during decoding for the LLPB-SMT model. The performance of the LLPB-SMT baseline model are obtained using a trigram language model (commonly used in translation systems). However, the computational complexity of the CRF-SLU approach does not allows the use of such a language in the label side. In order to evaluate the CRF-SLU approach and the LLPB-SMT under the same conditions, and since we can not increase the size of the characteristic functions of the CRF model, we propose to use a bigram language model for the LLPB-SMT model.

Furthermore, while using the CRF-SLU model, OOV words are translated by other words in the corpus according to the context of the sentence, unlike with the LLPB-SMT approach which tends to project untouched the OOV words in the translated sentence. These OOV words, being in most cases city names or places, their translation does not change from one language to another, and therefore their projection in the translated output is advantageous for LLPB-SMT models. In that purpose, we introduced a pre-processing step for OOV words in the source sentence to retrieve outputs in CRF-SLU approach.

The results presented in Table 2 show that the monotone decoding decreases the performance of the LLPB-SMT model by 0.91% absolute. The use of a bigram language model increases the loss of an extra 0.3%. OOV word processing allows the CRF-SLU model to recover 1.0% of BLEU score compared to the CRF-SLU baseline model. Despite downgrading of the LLPB-SMT model and improvements of the CRF-SLU model, the performance of the latter is still lower than the LLPB-SMT model (BLEU 43.5% for CRF vs. 46.0% for LLPB-SMT).

[2] Please refer to [12] for more details on the LLPB-SMT model.

Table 2. Comparaison of the LLPB-SMT and CRF-SLU models for Italian to French translation (BLEU %)

	CRF-SLU	**LLPB-SMT**
Baseline	42.5	47.2
Monotone decoding	42.5	46.3
Bigrams	42.5	46.0
OOV processing	**43.5**	**46.0**

5.3 Evaluation of the CRFPB-SMT Model for Translation and Understanding

A CRFPB-SMT model has been constructed for translation as described in Section 3. This model was built using the N-CODE tool [6], implemented to train translation models based on n-grams [21].

This tool uses the OpenFst library [1] to built a translation graph by composition of several transducers. The difference between the model implemented by this tool and the model we aim to develop lies in the parameters of so-called translation model. So we adapted the tool to use a CRF model to estimate the translation probabilities and a normalization of the probability scores obtained with the model is done over the different paths in the graph (as proposed in [17]).

In N-CODE, the reordering model (proposed by [5]) is based on a set of rules extracted automatically from training data. This approach requires a grammatical labelling of the source training sentences and a word alignment between the source and target sentences to train the λ_R model. The TreeTagger tool [28] was used for grammatical labelling and GIZA++ for word alignment. The language model used in our experiments is a trigram model trained on the target side of our training corpus using the SRILM tool [30].

Table 3 presents a comparison between three models: CRFPB-SMT, LLPB-SMT (baseline) and the CRF-SLU (presented in the previous section). The results show that the CRFPB-SMT approach based on transducers gives comparable performance to those obtained by the LLPB-SMT approach.

Despite a loss of 3.1% absolute, the performance is fairly high for a translation task (despite a limited size training set), which in our context is explained by the limited vocabulary of the domain. On the other hand the results show that the use of graphs in CRFPB-SMT model is doubly advantageous compared to the use of a basic CRF approach. Besides the fact that it allows to process input graphs (eg coming from the ASR module) this approach allows to increase the system performance by about 1% absolute.

The mechanism used to obtain a translation graph can be used for understanding. In a first step, the graph of concept hypotheses is obtained by composing all models $\lambda_S \circ \lambda_R \circ \lambda_T \circ \lambda_F \circ \lambda_L$ as for the translation. This approach gives a CER of 15.3%. To take into account the specificities of understanding (which does not require reordering model or long-range language model in the target side), we propose to obtain the output graph by combining only $\lambda_S \circ \lambda_T \circ \lambda_F$. This allowed

Table 3. Comparison between the different approaches (LLPB-SMT, CRF-SLU, CRFPB-SMT) for Italian to French translation

Model	**Language**	**BLEU**
LLPB-SMT		47.2
CRF-SLU	IT – FR	43.5
CRFPB-SMT		44.1

Table 4. Evaluation of CRF-based approaches for French understanding

Model	**Sub**	**Del**	**Ins**	**CER**
CRF-SLU	3.1	8.1	1.8	12.9
CRFPB-SMT ($\lambda_S \circ \lambda_R \circ \lambda_T \circ \lambda_F \circ \lambda_L$)	4.2	8.8	2.3	15.3
CRFPB-SMT (simplified) ($\lambda_S \circ \lambda_T \circ \lambda_F$)	3.5	7.6	2.0	13.1

to increase the performance of this approach by 2.2% absolute (15.3% vs 13.1%) to find almost the same performance as CRF-SLU (12.9%). A comparison between the performance of the different versions is given in Table 4. Subsequently, simplified CRFPB-SMT is used for all reported experiments.

5.4 Joint Decoding for Multilingual Understanding

A joint decoding for translation and understanding has been applied as proposed in section 4. This decoding will label Italian sentences by combining an Italian into French translation system and a French understanding system. For that, we adapted the acceptor of the French understanding model (given in the last row of Table 4 and described in Section 5.3) to take graphs as input (instead of a single hypothesis). This transducer generates a weighted understanding graph that takes into account translation scores.

The two scores (translation and understanding) are considered in the decoding. We propose that the final score for each path of the graph is the addition of the translation score and the understanding on this path [3]. The best path is then selected among all possible paths of the graph. This path represents a joint decoding between translation and understanding (marginalization of random variability caused by intermediate translation).

We propose to perform the joint decoding in two modes: in the first, the translation system used is a LLPB-SMT model (using the MOSES toolkit) while we used a CRFPB-SMT (as described in 5.3) in the second. In both cases the performance are compared with or without taking into account the hypotheses graph in the joint decoding. In the first case, the 1-best translation of the graph is transmitted to the understanding system. In the second case, the oracle of the translation graph is given as input for understanding. Oracle scores represent an evaluation based on the closest path to the translation reference. That allows

[3] An experiment to assess the impact of scores weighting is presented in [10].

Table 5. Evaluation of different configurations of multilingual understanding, depending on the communication channel (1-best, oracle or graph)

Translation			Understanding (CRF)		
Model	**Output**	**BLEU/Oracle**	**Input**	**CER/Oracle**	**BLEU**
LLPB-SMT	1-best	47.2/47.2	1-best	19.9/19.9	47.2
	graph	46.9/47.9	1-best(graph)	19.9/19.4	46.9
	graph	46.9/47.9	oracle(graph)	19.8/19.3	47.9
	graph	46.9/47.9	graph	19.7/19.1	46.3
CRFPB-SMT	graph	44.1/44.9	1-best(graph)	21.7/21.1	44.1
	graph	44.1/44.9	oracle(graph)	21.6/21.1	44.9
	graph	44.1/44.9	graph	21.3/20.6	43.9

to evaluate the impact of translation quality on understanding performance (see table 5). We also evaluate the oracle scores (for translation and understanding) on the outputs of the different combination, and we calculated the BLEU score on the selected translation by the joint decoding (last column of Table 5).

The first line of this table is the baseline combination (without the use of a graph) wherein the output of MOSES is given as input to a CRF model. The results show that the graph of translation improves the performance of the system compared to the system using the 1-best (CER 19.7% vs. 19.9% for LLPB-SMT and 21.3% vs. 21.7% for CRFPB-SMT). Using a translation graph also gives better performance compared to the combination with the translation oracle (CER 19.7% vs. 19.8% for LLPB-SMT and 21.3 vs. 21.6 for CRF).

The difference between the performance obtained by the joint decoding using LLPB-SMT model for translation and this obtained using a CRFPB-SMT model (CER 19.7% vs. 21.3%) can be explained by the difference between the performance of these two models (BLEU 46.9% vs 44.1%).

It is important to mention that only combinations taking an input graph for understanding allow to select the translation according to the labelling that will be applied. In other cases the selection of the translation is done independently. We note that the BLEU score of the translation selected by joint decoding is lower than the best translation (46.3 vs. 47.2 for LLPB-SMT and 43.9 vs. 44.1 for CRFPB-SMT) despite that the former is better in terms of CER. This proves the interest of the joint graph based method.

6 Conclusion

In this paper we evaluated and compared stochastic approaches for both speech understanding and automatic translation. We showed that the discriminative CRF approach is the best approach for speech understanding, despite all the adaptations of LLPB-SMT approach for the task. Using a CRF approach for translation has several limitations and the performance of this approach can be improved by using a model based on transducers allowing the integration of appropriates models (reordering, segmentation, language model).

We proposed and evaluated an approach for joint decoding between the translation and the understanding in the context of a multilingual understanding system. We have shown that with such a decoding we can achieve good performance while providing a homogeneous system for two underlying tasks.

In the context of a human-machine dialogue system, a joint decoding between speech recognition and translation can be added. That allows the recognition system to transmit richer information to the understanding system and understanding system in turn will transmit rich information to the dialogue manager which might positively influence the overall system performance.

References

1. Allauzen, C., Riley, M., Schalkwyk, J., Skut, W., Mohri, M.: OpenFst: A general and efficient weighted finite-state transducer library. In: Holub, J., Žd̆árek, J. (eds.) CIAA 2007. LNCS, vol. 4783, pp. 11–23. Springer, Heidelberg (2007)
2. Anoop Deoras, R.S.G.T., Hakkani-Tur, D.: Joint decoding for speech recognition and semantic tagging. In: INTERSPEECH (2012)
3. Bonneau-Maynard, H., Rosset, S., Ayache, C., Kuhn, A., Mostefa, D.: Semantic annotation of the french media dialog corpus. In: EUROSPEECH (2005)
4. Brown, P.F., Pietra, S.D., Pietra, V.J.D., Mercer, R.L.: The mathematics of statistical machine translation: Parameter estimation. Computational Linguistics 19(2), 263–311 (1993)
5. Crego, J.M., Mariño, J.B.: Improving statistical mt by coupling reordering and decoding. Machine Translation 20(3), 199–215 (2006)
6. Crego, J.M., Yvon, F., Mariño, J.B.: Ncode: an open source bilingual n-gram smt toolkit. The Prague Bulletin of Mathematical Linguistics 96, 49–58 (2011)
7. Gascó i Mora, G., Sánchez Peiró, J.A.: Part-of-speech tagging based on machine translation techniques. In: Martí, J., Benedí, J.M., Mendonça, A.M., Serrat, J. (eds.) IbPRIA 2007. LNCS, vol. 4477, pp. 257–264. Springer, Heidelberg (2007)
8. Hahn, S., Dinarelli, M., Raymond, C., Lefèvre, F., Lehnen, P., De Mori, R., Moschitti, A., Ney, H., Riccardi, G.: Comparing stochastic approaches to spoken language understanding in multiple languages. IEEE Transactions in Audio, Speech and Language Processing 19(6), 1569–1583 (2010)
9. Hakkani-Tür, D.Z., Béchet, F., Riccardi, G., Tür, G.: Beyond asr 1-best: Using word confusion networks in spoken language understanding. In: Computer Speech and Language, pp. 495–514 (2006)
10. Jabaian, B.: Systèmes de compréhension et de traduction de la parole: vers une approche unifiée dans le cadre de la portabilité multilingue des systèmes de dialogue. Ph.D. thesis, CERI - Universitré d'Avignon, Avignon (2012)
11. Jabaian, B., Besacier, L., Lefèvre, F.: Investigating multiple approaches for slu portability to a new language. In: INTERSPEECH (2010)
12. Jabaian, B., Besacier, L., Lefèvre, F.: Combination of stochastic understanding and machine translation systems for language portability of dialogue systems. In: ICASSP (2011)
13. Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., et al.: Moses: Open source toolkit for statistical machine translation. In: ACL (2007)
14. Koehn, P., Och, F.J., Marcu, D.: Statistical phrase-based translation. In: HLT-NAACL (2003)

15. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: ICML (2001)
16. Lavergne, T., Cappé, O., Yvon, F.: Practical very large scale CRFs. In: ACL (2010)
17. Lavergne, T., Crego, J.M., Allauzen, A., Yvon, F.: From n-gram-based to crf-based translation models. In: WSMT (2011)
18. Liang, P., Taskar, B., Klein, D.: Alignment by agreement. In: HLT-NAACL (2006)
19. Macherey, K., Bender, O., Ney, H.: Application of statistical machine translation approaches to spoken language understanding. In: IEEE ICASSP (2009)
20. Macherey, K., Och, F.J., Ney, H.: Natural language understanding using statistical machine translation. In: INTERSPEECH (2001)
21. Mariño, J.B., Banchs, R.E., Crego, J.M., de Gispert, A., Lambert, P., Fonollosa, J.A.R., Costa-jussà, M.R.: N-gram-based machine translation. Computational Linguistic 32(4), 527–549 (2006)
22. Och, F.: Minimum error rate training in statistical machine translation. In: ACL (2003)
23. Och, F.J., Ney, H.: Discriminative training and maximum entropy models for statistical machine translation. In: ACL (2002)
24. Papineni, K., Roukos, S., Ward, T., Zhu, W.: Bleu: a method for automatic evaluation of machine translation. In: ACL (2002)
25. Rama, T., Singh, A., Kolachina, S.: Modeling letter-to-phoneme conversion as a phrase based statistical machine translation problem with minimum error rate training. In: HLT-NAACL (2009)
26. Ramshaw, L., Marcus, M.: Text chunking using transformation-based learning. In: The Workshop on Very Large Corpora (1995)
27. Riedmiller, M., Braun, H.: A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In: ICNN (1993)
28. Schmid, H.: Probabilistic part-of-speech tagging using decision trees. In: NMLP (1994)
29. Servan, C., Raymond, C., Béchet, F., Nocera, P.: Conceptual decoding from word lattices: application to the spoken dialogue corpus MEDIA. In: INTERSPEECH (2006)
30. Stolcke, A.: Srilm-an extensible language modeling toolkit. In: ICASSP (2002)
31. Tür, G., Wright, J.H., Gorin, A.L., Riccardi, G., Hakkani-Tür, D.Z.: Improving spoken language understanding using word confusion networks. In: INTERSPEECH (2002)
32. Turian, J.P., Wellington, B., Melamed, I.D.: Scalable discriminative learning for natural language parsing and translation. In: NIPS (2006)

Speech/Music Discrimination via Energy Density Analysis

Stanisław Kacprzak and Mariusz Ziółko

Department of Electronics, AGH University of Science and Technology
al. Mickiewicza 30, Kraków, Poland
{skacprza,ziolko}@agh.edu.pl

Abstract. In this paper we suggest to apply a new feature, called Minimum Energy Density (MED), in discrimination of audio signals between speech and music. Our method is based on the analysis of local energy for 1 or 2.5 seconds audio signals. An elementary analysis of the probability for the power distribution is an effective tool supporting the decision making system. We compare our feature with Percentage of Low Energy Frames (LEF), Modified Low Energy Ratio (MLER) and examine their efficiency for two separate speech/music corpora.

Keywords: speech/music discrimination, sound classification, audio content analysis.

1 Introduction

Discrimination between speech and music has applications in different areas of speech processing, such as voice activity detection (VAD), automatic corpus creation [11] and as part of modern hearing aids [1]. For the purpose of this discrimination many features, in time as well as in frequency domain, have been proposed [2], [9]. The most common are 4 Hz modulation energy, entropy modulation, spectral centroid, spectral flux, zero-crossing rate and cepstral coefficients, but more complex parameters like wavelet-based parameters [3] are also explored. Recognition rate over 98% [8], [9], has been reported for subsets of these features and their variations. Current research is focused on achieving high recognition rate with aspect of minimizing required computations. In this paper we focus on speech/music discrimination based on energy features. We analyse energy distribution in speech and music signals and upon this analysis we introduce a new feature Minimum Energy Density (MED). We compare this feature with Percentage of Low Energy Frames (LEF), Modified Low Energy Ratio (MLER) and examine their efficiency for corpus collected by Scheirer and Slaney [9] and a second one, created by us.

2 Energy Features

It is very intuitive to try to discriminate speech and music based on shape of signal's energy envelope. As Fig. 1 shows, speech signal has characteristic high and low amplitude parts, which represent voiced and unvoiced speech, respectively. On the other hand, the envelope of music signal is more steady. Moreover, we know that speech has a characteristic 4 Hz energy modulation, which matches the syllabic rate [9].

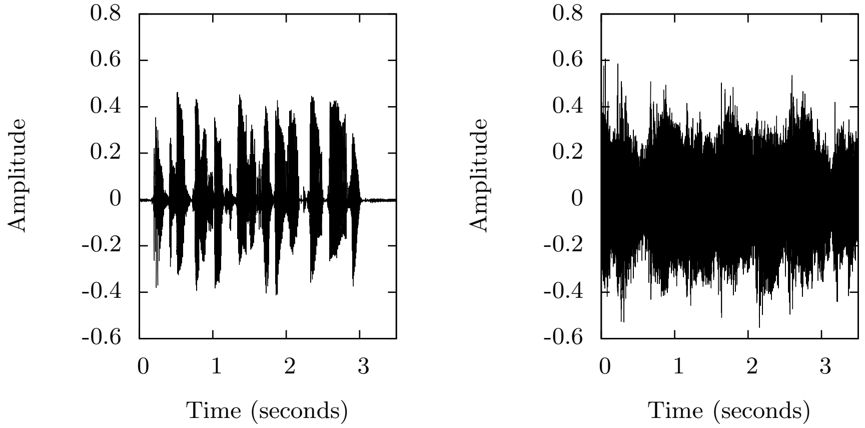


Fig. 1. Speech (left) and music (right) samples

Saunders [8] stated: "The energy contour is well known to be capable of separating speech from music." His algorithm however, was based on zero-crossings rate features and 90% accuracy was reported. It is interesting that after adding a new feature, which was a measure of energy minima below some threshold relative to peak energy, accuracy rose to 98%. Results based only on this energy feature were not presented. Measure of rapid changes in speech signal was the base of speech/music discrimination in hardware device described in patent [4].

In [9] authors define Percentage of Low Energy Frames (LEF) feature as percentage of frames within 1 s window with root mean square (RMS) power below 50% of window mean RMS power. This feature alone provides 14% error rate and was the fastest one in the sense of computational time. Similar feature was proposed in [5], but authors used short term energy instead of RMS. Wang, Gao and Ying in [10] explore this idea by introducing Modified Low Energy Ratio (MLER) which is different from LEF in the fact that percentage of the window mean short term energy is not fixed to 50%, but its value is subject to change. The formal definition of MLER [10] is

$$MLER = \frac{1}{2N} \sum_{n=1}^{N} \left[\operatorname{sgn}\left(lowthres - E(n)\right) + 1\right], \tag{1}$$

where

$$lowthres = \delta \cdot \sum_{n=1}^{N} E(n) \tag{2}$$

and N is the total number of frames in a window and $E(n)$ is frame short time energy.

These features take under consideration skewness of energy distribution in speech (Fig. 2), caused by the fact that there are many low energy or quiet frames in speech, also more than in music. However, these features ignore energy distribution within a window. Thus, they will fail in the presence of speech window with low mean energy, that can appear for example when a fricative is followed by a pause, or in case of whole silent window, which may occur if the person is speaking slowly. Moreover, because of relative character of this feature, MLER can fail in the presence of additive noise, since it would be necessary to increase δ with the decrease of SNR.

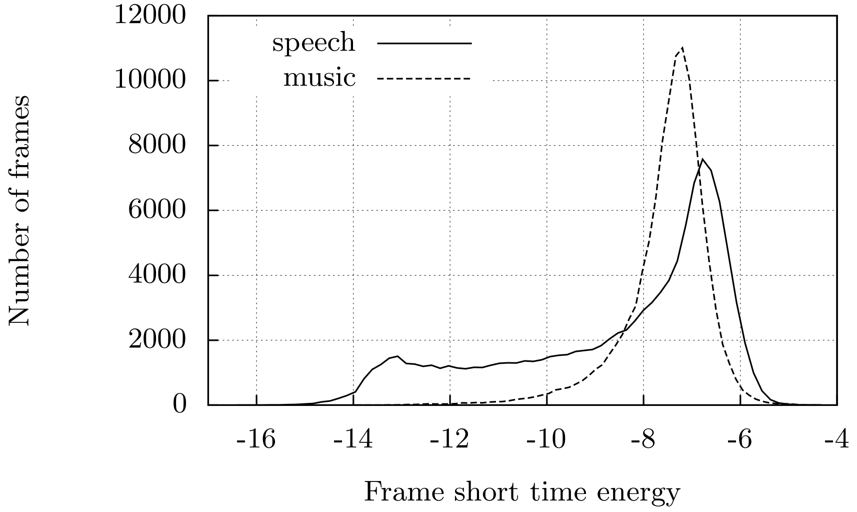


Fig. 2. Histogram outlines of normalized short time energy calculated for audio samples used in [9]. Values of energy have been log-transformed.

The number of energy dips below the value of threshold, which is little above noise level, was used as a feature in [7], where 86% accuracy was reported for 5 s windows, but tests where performed on very rigorous music data which contained single instrument music. Our feature explores idea of classification based on energy dips.

3 Minimum Energy Density Feature

We know from energy distribution (Fig. 2) that speech has more low energy frames than music. We also know that speech has 4 Hz energy modulation, which implies four energy minima in 1 s window. These facts allow us to suspect that the presence of the frame with energy below some calculated threshold will be sufficient to distinguish between speech and music. The disadvantage of this approach is inability to rely on some fixed threshold value, due to differences in signals power. To overcome that, we calculate probability distribution of short time frame energy inside some time window, which we refer to as *normalization window*. Normalization window has to be long enough to capture the nature of the signal. For example 1 s window seems a bad idea, since in case of window containing breathe pause we would get distribution close to uniform and information about low energy of that window would be lost. We define probability of short time frame energy as

$$probE(n) = \frac{E(n)}{\sum_{k=1}^{N} E(k)} \ . \tag{3}$$

Next step is to find minimum $probE(n)$ in the *classification window*. Length of the classification window can be shorter than the length of normalization window and it defines classification resolution. Taking into account the 4 Hz energy modulation characteristic for speech, the length of the classification window should be at least 250 ms. We define Minimum Energy Density (MED) for k-th classification window as

$$MED(k) = \min\{probE(n) : (k-1) \cdot M + 1 \le n \le k \cdot M\}, \tag{4}$$

where M is the number of frames in the classification window.

During training phase a threshold value for MED is found so that the windows with MED below that threshold are classified as speech and the rest as music. In fact, for classifying unseen data, there is no need to find minimum value of the classification window as in (4), because finding any frame with energy below the threshold is sufficient to classify the window as speech. Additionally we can reduce needed computations by, instead of normalizing each frame in normalization window, scaling the threshold. Final decision about class for a classification window is given by

$$class(k) = \begin{cases} speech & \text{if} \quad \exists n : E(n) < \lambda, \ \text{ where } \ (k-1) \cdot M + 1 \le n \le k \cdot M \\ music & \text{otherwise} \end{cases} , \tag{5}$$

where

$$\lambda = threshold \cdot \sum_{n=1}^{N} E(n) \ . \tag{6}$$

Figure 3 shows histogram outlines of MED feature for speech and music signals.

4 Test Corpora

To evaluate our algorithm we use two separate audio data sets. First set, which will be referred to as A, is the same that was used in [9] and consists of eighty

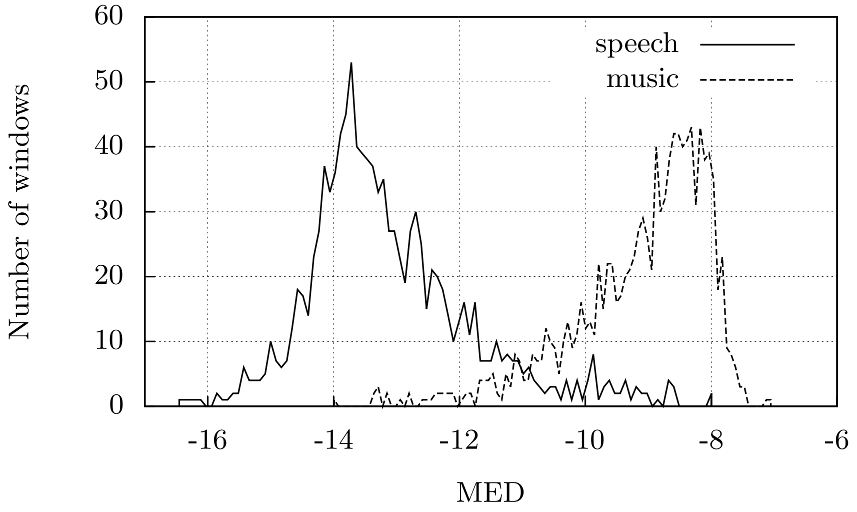


Fig. 3. Histogram outlines of MED calculated on audio samples used in [9]. Values of MED have been log-transformed.

15-second long audio samples of speech and the same amount of music samples. As authors stated, the data was collected by digitally sampling FM tuner (16-bit at a 22.05 kHz sampling rate). Speech data contains male and female speakers, in quiet and in noisy conditions. Music data set contains variety of music styles, with and without vocals. The second data set, which will be referred to as B, was collected by us. We also prepared eighty 15-second long audio samples of speech from mp3 of Polish audio-books and same amount of music derived from private mp3 library (16-bit at a 44.1 kHz, stereo files were transformed to mono). The speech samples feature both male and female, mostly professional speakers and actors while in music data set we try to capture variety of music genres like rock, pop, jazz, dance and reggae.

5 Experiment Evaluation

We examine our algorithm using 10 ms frames, 15 s (whole audio sample) normalization window and 1 s and 2.5 s classification windows. We compare results of our new feature with LEF and MLER. For MLER we analyse the effect of δ value first. The results, which are shown in Fig. 4, imply that in our case $\delta = 0.1$, as suggested in [10], is not the best possible option. Instead we choose $\delta = 0.02$, which is the cross point of lines representing average accuracy. To evaluate our algorithms for every experiment we use over 10 cross-validation runs. In each run we calculate MED for all samples. 70% of calculated parameters selected at random is used as training set and the remaining 30% is used for testing. During the training session the best threshold value that maximizes overall classification accuracy over the training set is found and that threshold is used to classify data under test set. The mean results of cross-validation runs of speech/music discrimination for 1 s and 2.5 s classification windows are shown in Tab. 1 and Tab. 2, respectively.

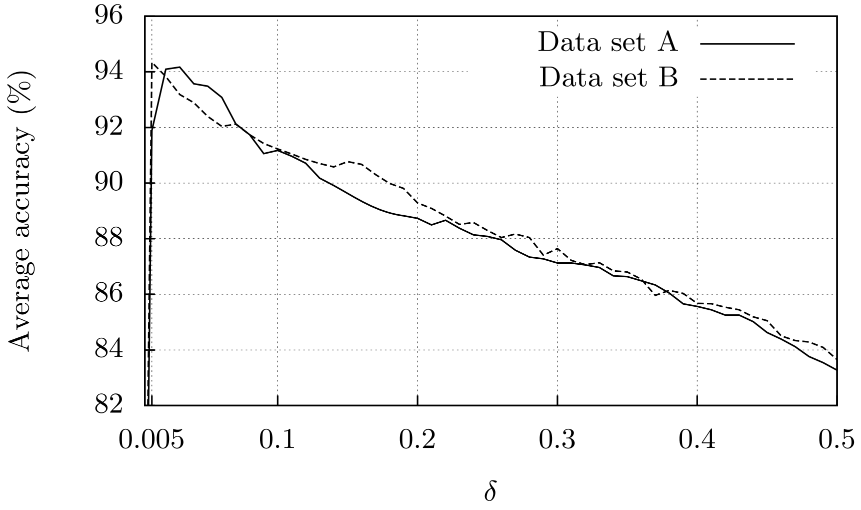


Fig. 4. Average accuracy of correct recognition based on MLER in function of parameter δ

Table 1. Correct classification results (mean and standard deviation) for the 1 s classification window

	Data set A			Data set B		
	LEF	MLER	MED	LEF	MLER	MED
speech	87.5 ± 3.9%	91.3 ± 1.0%	91.6 ± 1.5%	88.1 ± 2.3%	95.1 ± 1.2%	94.9 ± 1.3%
music	90.1 ± 3.2%	96.7 ± 0.6%	95.3 ± 1.4%	90.4 ± 1.3%	92.6 ± 1.4%	95.3 ± 1.0%
total	**88.8 ± 0.9%**	**94.0 ± 0.3%**	**93.5 ± 0.4%**	**89.3 ± 1.3%**	**93.8 ± 0.6%**	**95.1 ± 0.6%**

Table 2. Correct classification results (mean and standard deviation) for the 2.5 s classification window

	Data set A			Data set B		
	LEF	MLER	MED	LEF	MLER	MED
speech	92.4 ± 2.5%	95.4 ± 2.1%	94.5 ± 2.2%	96.3 ± 1.7%	96.8 ± 2.3%	98.0 ± 1.1%
music	91.0 ± 3.5%	95.7 ± 1.6%	97.0 ± 1.4%	94.3 ± 1.7%	95.9 ± 2.0%	96.0 ± 1.5%
total	**91.7 ± 1.6%**	**95.5 ± 1.2%**	**95.8 ± 1.5%**	**95.3 ± 1.1%**	**96.3 ± 1.2%**	**97.0 ± 0.7%**

6 Conclusions

The results in Tab. 1 and Tab. 2 demonstrate that MED method performs better than LEF and slightly better or similarly as MLER. However, our method is not dependent on any parameter, like δ in case of MLER, that has a strong effect on accuracy and its optimal value depends on tested data. In case of the 2.5 s classification window our method achieves 95.8% accuracy for data set A and 97% accuracy for data set B, what are very high results for single feature. In contrast, in [9] authors reported 98.6% accuracy on the 2.4 s window using GMM classifier based on 3 features.

Furthermore, in case of our algorithm, after finding the frame with energy below the threshold, the calculation stops for a given window, resulting in the reduction of the expected number of calculations. This fact and the manner in which the threshold energy value based on MED is found, distinguishes our algorithm from one presented in [7] and shows that MED is sufficient for good discrimination in case of speech and typical modern music. Considering its good performance and low computation load, the algorithm which is based on MED feature allows more effective speech/music discrimination.

It needs to be pointed out that our tests include only recordings of speech or music. There were no examples of speech over music, which would imply three class discrimination, because classifying such signal as speech or music is subjective. Nevertheless, our method alone has potential to be used for tasks like automatic corpus [6] creation from sources for which we have prior knowledge that are compound of alternating speech and music like audio-books, language courses or radio drama.

Acknowledgements. The project was funded by the National Science Centre allocated on the basis of a decision DEC-2011/03/B/ST7/00442.

References

1. Cabañas Molero, P., Ruiz Reyes, N., Vera Candeas, P., Maldonado Bascon, S.: Low-complexity f0-based speech/nonspeech discrimination approach for digital hearing aids. Multimedia Tools and Applications 54, 291–319 (2011)
2. Carey, M., Parris, E., Lloyd-Thomas, H.: A comparison of features for speech, music discrimination. In: Proceedings of 1999 IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 1, pp. 149–152 (March 1999)
3. Didiot, E., Illina, I., Fohr, D., Mella, O.: A wavelet-based parameterization for speech/music discrimination. Comput. Speech Lang. 24(2), 341–357 (2010)
4. Jones, R.C.: Electronic device for automatically discriminating between speech and music forms. US Patent 2761897 (1956)
5. Lu, L., Jiang, H., Zhang, H.: A robust audio classification and segmentation method. In: Proceedings of the Ninth ACM International Conference on Multimedia, MULTIMEDIA 2001, pp. 203–211. ACM, New York (2001)
6. Masior, M., Ziółko, M., Kacprzak, S.: Multi-lingual speech samples base, `http://speechsamples.agh.edu.pl/`

7. Okamura, S., Aoyama, K.: An experimental study of energy dips for speech and music. Pattern Recognition 16(2), 163–166 (1983)
8. Saunders, J.: Real-time discrimination of broadcast speech/music. In: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 1996, vol. 2, pp. 993–996 (May 1996)
9. Scheirer, E., Slaney, M.: Construction and evaluation of a robust multifeature speech/music discriminator. In: 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 1997, vol. 2, pp. 1331–1334 (April 1997)
10. Wang, W., Gao, W., Ying, D.: A fast and robust speech/music discrimination approach. In: Proceedings of the 2003 Joint Conference of the Fourth International Conference on Information, Communications and Signal Processing, and Fourth Pacific Rim Conference on Multimedia, vol. 3, pp. 1325–1329 (December 2003)
11. Wei, Z., Ranran, D., Minhui, P., Qiuhong, W.: Automatic speech corpus construction from broadcasting speech databases. In: 2010 International Conference on Computational Intelligence and Security (CIS), pp. 639–643 (December 2010)

Histogram Equalization Using Centroids of Fuzzy C-Means of Background Speakers' Utterances for Speaker Identification

Myung-Jae Kim, Il-Ho Yang, and Ha-Jin Yu*

School of Computer Science, University of Seoul, Seoul, Korea
mj@uos.ac.kr, heisco@hanmail.net, hjyu@uos.ac.kr

Abstract. In this paper, we propose a novel approach of histogram equalization for speaker recognition with short utterances which are not enough for building histograms. The proposed method clusters the features of randomly selected background speakers' utterances, and estimates the cumulative distribution using the centroids of the clusters sorted in ascending order and the samples of a short test utterance. The ranks are obtained from the test utterance and the sorted centroid set and the sum of the two ranks are used to estimate the cumulative distribution function. For the evaluation, we use ETRI PC database and simulate VoIP codecs for the test set. The system is compared with other feature normalization methods such as CMN, MVN and the conventional HEQ. Our proposed method reduces the error rates by 27.9%, 35.9%, and 30.1% relatively in the test environments: G.729, SILK and Speex, respectively.

Keywords: speaker recognition, speaker identification, histogram equalization.

1 Introduction

The current speaker recognition systems show good performance when the training and test environments are matched. However, it is difficult to match both environments in real situations. We use the feature normalization methods to overcome this problem. CMN (cepstral mean normalization) [1] and MVN (mean and variance normalization) [2] are widely used to normalize features. The feature normalization methods are suitable for removing linear channel effect but are weak in non-linear effects. In the speech and speaker recognition systems, the additive or channel noises affect the distribution of the utterances. To compensate for these effects, histogram equalization (HEQ) which is a non-linear transformation method using reference distribution such as standard normal distribution has been proposed [5]. Originally, the HEQ is used in order to control brightness and contrast of digital images in image processing [5]. Then it has been applied as a feature normalization method in speech recognition [9][10] and speaker recognition [7]. In speaker recognition, methods of applied HEQ have been proposed such as feature warping [6] and modified segmental HEQ [8]. The approaches divide an utterance into small windows and apply HEQ to

* Corresponding author.

each window, and the size of the windows should be at least three seconds long. According to Blanco's research [3], *"around 500 ordered samples are enough to estimate very robustly and easily any CDF"*. Therefore, it is difficult to expect good performance of HEQ with short utterances which have fewer than 500 feature vectors. To alleviate the problem with short utterances, we supplement the utterance with features from the training set for the background model. The features used to supplement are a set of fuzzy C-means centroids of each speaker's speech which is used for training the UBM.

In this paper, we describe a HEQ method for short utterances using complement features. In section 2, we review HEQ. In section 3, we describe our proposed method. In section 4, we show the experimental setup and results. Finally, we discuss the conclusion and the future works.

2 Histogram Equalization

HEQ transforms a given probability distribution into a reference probability distribution. The variable to be normalized is transformed by HEQ using the CDF (cumulative distribution function) mapping between original distribution and reference distribution. The transform function is given as

$$y = C_{ref}^{-1}(C_x(x)) \tag{1}$$

where x is the original value, y is the transformed value, C is the CDF of original distribution and C_{ref}^{-1} is the CDF of the inverse of the reference distribution. Figure 1 shows an example of the transformation. The distribution of samples before the transformation (left) has a multimodal distribution and the distribution of samples after the transformation (right) has a uni-modal distribution.

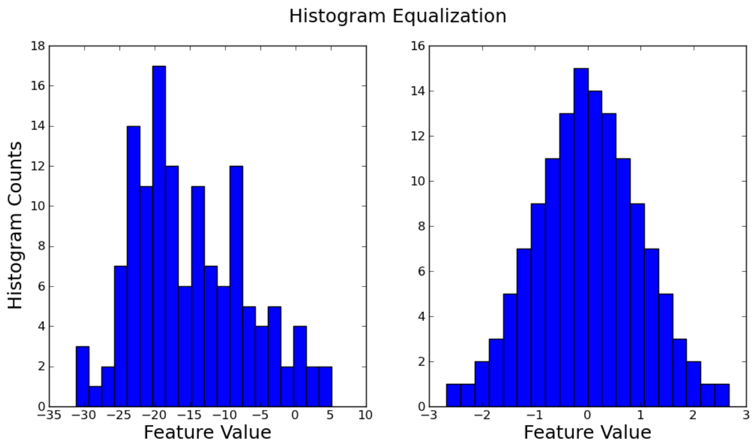


Fig. 1. An example of the transformation of samples using HEQ

2.1 Cumulative Histogram-Based HEQ

CHEQ (cumulative histogram-based HEQ) uses bins which are equally-spaced and non-overlapped to estimate CDF. The number of samples in each bin is used to build a histogram of the samples. The created histogram is used to estimate CDF of each bin. The details of CHEQ is described in [8]

2.2 Order-Statistics-Based HEQ

OHEQ (order-statistics-based HEQ) uses ranks of samples in ascending order. To build a histogram of the samples, the ranks are used. It is known that the order-statistic-based CDF estimation is more accurate than the cumulative histogram-based estimation when the amount of estimation samples is small [4]. The details of OHEQ is described in [9].

3 The Proposed Method

Our method is based on order-statistics-based HEQ. To reinforce short utterances, the proposed method uses the centroids of the clusters of the background speakers' utterances. Figure 2 shows the process of the proposed HEQ. In the offline step, we generate the supplement samples by clustering each speaker's UBM training set using a fuzzy C-means [12] algorithm (*m*=2). To get the rank of the samples, all of the centroids are collected and are sorted in ascending order. This supplement set is used to estimate CDF in offline and online steps.

The details of the proposed HEQ are described as follows. Let U be the M speakers' feature vectors which are used to train the UBM. Then U is given as

$$U = \{U^1, U^2, \cdots, U^m, \cdots U^M\} \tag{2}$$

where U^m is the observed *D*-dimensional features of *m*-th speaker in the UBM training set. Fuzzy C-means is applied to each U^m. As a result of the fuzzy C-means, each speaker has K centroids. Newly obtained centroids are as follows:

$$G = \{G^1, G^2, \cdots, G^m, \cdots, G^M\} \tag{3}$$

The K centroids of particular (*d*-th) component of *m*-th speaker can be represented as

$$G_d^m = \{g_d^m(1), g_d^m(2), \cdots, g_d^m(k), \cdots, g_d^m(K)\} \tag{4}$$

We can define a new sequence S consisting of T frames of a particular feature component as

$$S_d = \{s_d(1), s_d(2), \cdots, s_d(t), \cdots s_d(T)\} \tag{5}$$

where S_d is a sequence formed by concatenating a particular component of the centroids of all speakers. The length of T is M × K. Let X denote test feature vectors consisting of N frames. A test sequence of a particular component of X can be given as

$$X_d = \{x_d(1), x_d(2), \cdots, x_d(n), \cdots, x_d(N)\} \tag{6}$$

where $x_d(n)$ is the d-th feature component at n-th frame. To estimate CDF, we sort the sequence S_d and X_d in ascending order. We use two kinds of ranks. The one is the rank of $x_d(n)$ in the sequence S_d, which is denoted as $r_d^s(n)$. The other is the rank of $x_d(n)$ in the sequence X_d, which is represented as $r_d^x(n)$. The new rank is defined as follows

$$r_d(n) = r_d^s(n) + r_d^x(n) \tag{7}$$

Then the CDF is estimated as

$$\Phi_d(n) = \frac{r_d(n) - 0.5}{T + N} \tag{8}$$

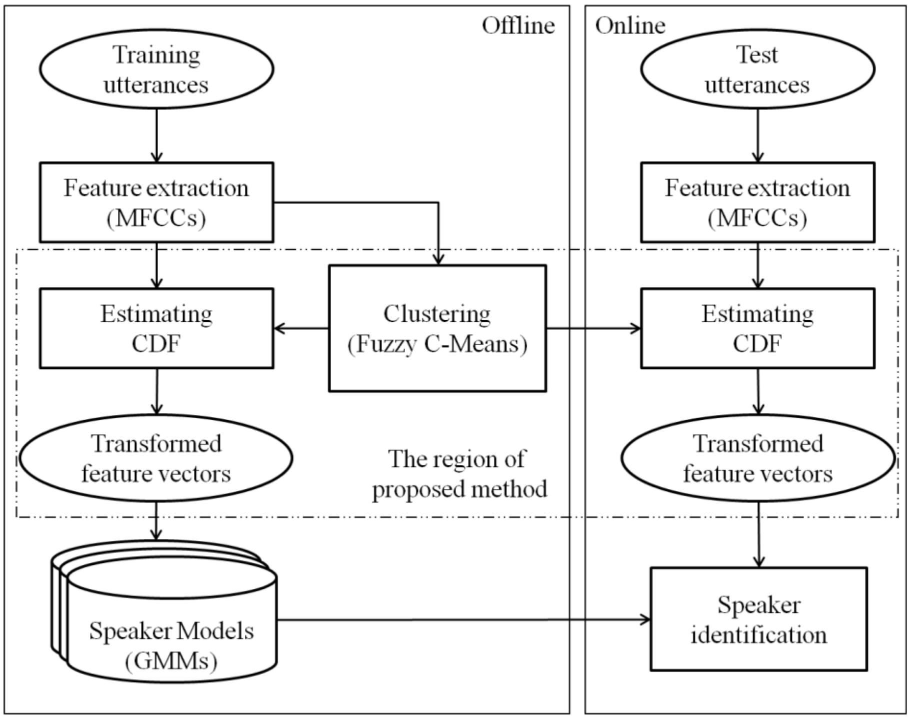

Offline
Online
Training utterances
Test utterances
Feature extraction (MFCCs)
Feature extraction (MFCCs)
Estimating CDF
Clustering (Fuzzy C-Means)
Estimating CDF
Transformed feature vectors
The region of proposed method
Transformed feature vectors
Speaker Models (GMMs)
Speaker identification

Fig. 2. The flow diagram of the proposed method

The transformed sequence is calculated as follows

$$Y_d = C_{ref}^{-1}(\Phi_d) \tag{9}$$

A CDF table is used to get the inverse CDF.

In this paper, we use standard normal distribution as the reference distribution.

$$P(x) = \frac{1}{\sqrt{2\pi}} \exp\left(\frac{-x^2}{2}\right) \tag{10}$$

4 Experimental Setup and Results

4.1 Database

To evaluate the proposed method, we use "Korean Speaker Recognition Database Recording using Middle Price Microphone" distributed by ETRI. The database contains week-set, month-set and season-set. We use sentence numbered 10 ~ 19 of the first trial of first session of week-set to adapt speaker models and use same sentence of first trial of 2, 3, 4 sessions to test the system. We use sentence numbered 10 ~ 19 of the first trial of the first session of month-set and season-set to train the UBM. Week-set is composed of 100 speakers, month-set is composed of 100 speakers and season-set is composed of 50 speakers. The database is recorded at 16 kHz. In order to match the sampling rate, the waveforms are downsampled from 16 kHz to 8 kHz. The total number of utterances of the UBM set is 1500 (150×10), the total number of training utterances is 1000 (100×10), and the total number of test utterances is 3000 ($3 \times 100 \times 10$). Each utterance consists of 200~300 samples and is reduced to 150~250 samples due to silence elimination.

4.2 Feature Extraction and Speaker Modeling

To evaluate speaker identification, test speech signals are filtered with a pre-emphasis filter with a factor 0.97. The filtered signal is partitioned into frames 25ms long with 10ms interval, and Hamming window is applied to each frame. The silent parts are removed by using energy threshold. Each frame is filtered by a bank of 26 mel-scaled triangular filters and then the output is cosine transformed into 18-dimensional MFCCs.

For speaker modeling, GMMs (Gaussian mixture models) are adopted using MAP ($\tau = 1$) from a UBM with 128 mixture components [11].

4.3 VoIP Codecs

To simulate channel mismatches, we apply three kinds of VoIP codecs G.729 [13], SILK [14] and Speex [15] to the test sets. Table 1 shows the specification of the codecs we use. We simulate the test set in 8 kHz mode.

Table 1. Specification of codecs used

Codec	Organization	Sampling rate	Version
G.729	ITU-T	8 kHz	3.3
SILK	Skype	8, 12, 16, 24 kHz	1.0.7
Speex	Xiph.org	8, 16, 32 kHz	1.2rc1

4.4 Results

We compare the performance of our proposed method with baseline systems which use MFCC without feature normalization method, and with conventional feature normalization methods. In the figures below, MFCC denotes the baseline of our speaker identification system using MFCC without feature normalization method. CMN and MVN mean that the MFCCs normalized using CMN and MVN respectively. CHEQ denotes the cumulative histogram-based HEQ. To build a histogram, we use 1000 bins for CHEQ. OHEQ denotes the order-statistics-based HEQ. The "proposed" denotes our proposed method. We select 10 speakers randomly from the UBM training set to get the centroids.

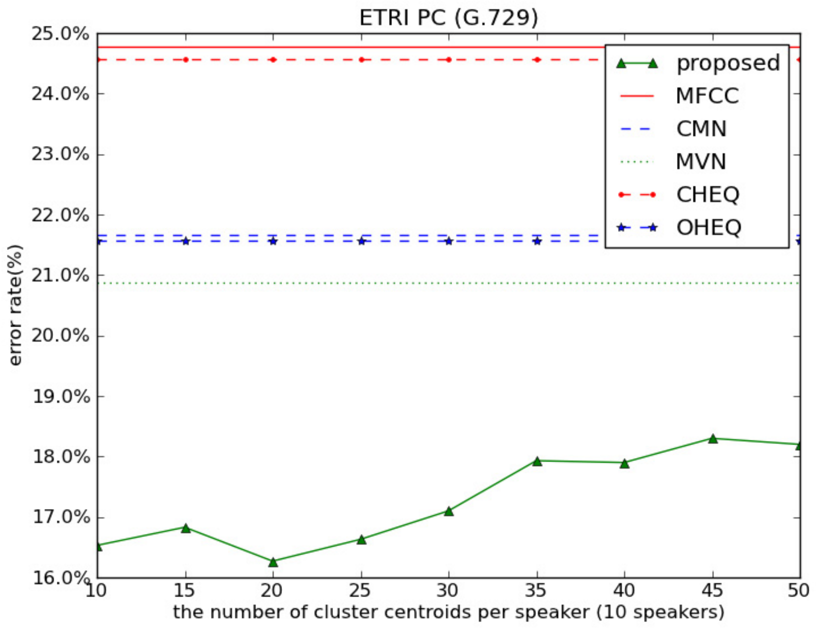


Fig. 3. Speaker identification error rates in G.729 environment

Figure 3 shows the results in G.729 environment, which shows the decreased performance by channel mismatch. Our method shows better performance than other methods when the number of centroids is from 10 to 50. Figure 4 shows the results in

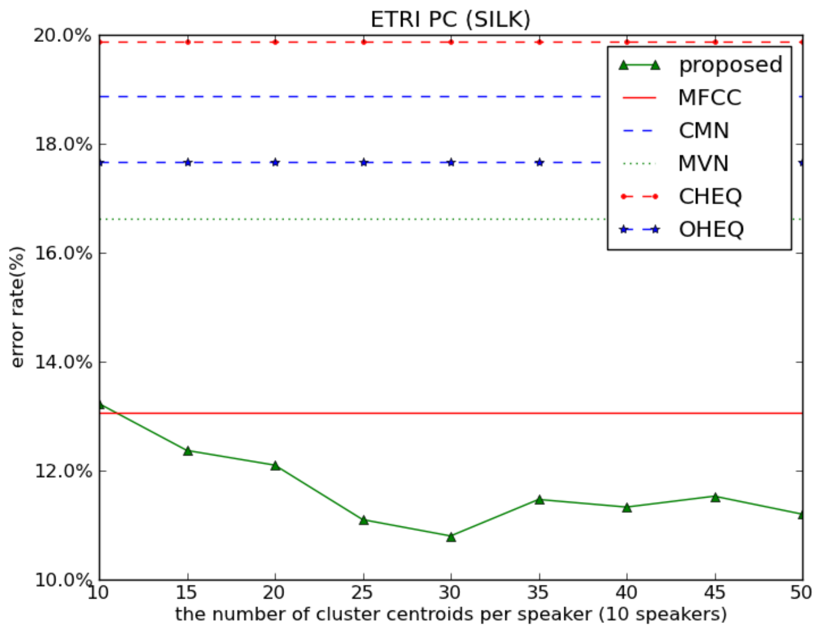


Fig. 4. Speaker identification error rates in SILK environment

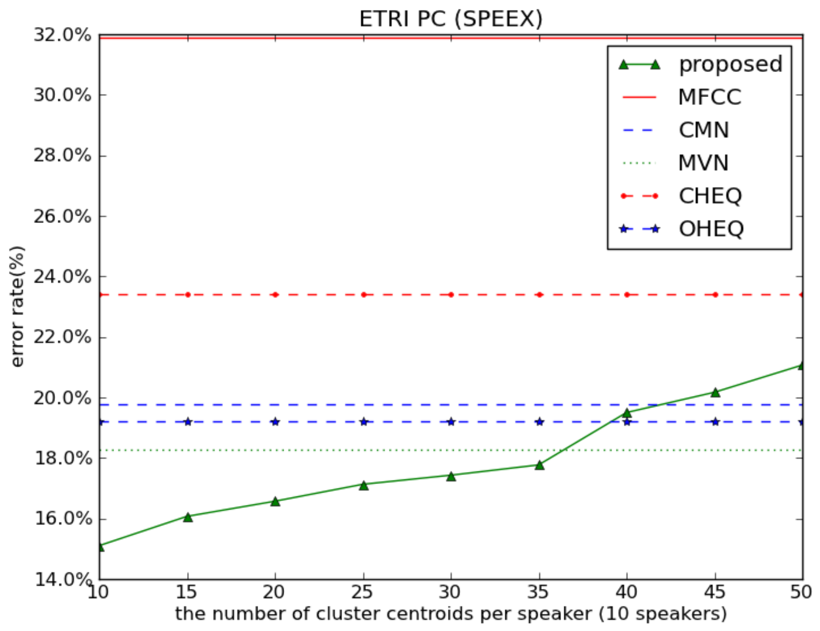


Fig. 5. Speaker identification error rates in Speex environment

SILK environment. Also, in SILK environment, the decreased performance is observed. The proposed method has the best performance when the number of centroids is between 15 and 50. Figure 5 shows the results in Speex environment. In Speex environment, CHEQ has poor performance than the MFCC-only system. Our proposed method shows better performance when the number of centroids is between 10 and 35.

Table 2. The relative error rate reduction when the proposed method has the best performance

Normalization method	Relative reduction of error rates (%)		
	G.729	SILK	SPEEX
MFCC	34.32	17.35	52.62
CMN	24.92	42.76	23.61
MVN	22.04	35.07	17.34
CHEQ	33.78	45.63	35.56
OHEQ	24.57	35.56	21.35

Table 3 shows the number of centroids used to calculate table 2.

Table 3. The number of cluster centroids when the proposed HEQ has the best performance

Channel	The number of cluster centroids
G.729	20×10
SILK	30×10
Speex	10×10

5 Conclusion and Future Works

In this paper, we propose a novel approach of HEQ using fuzzy C-means cluster centroids of the features for the universal background model. We use the centroids of fuzzy C-means clusters to estimate cumulative distribution function robustly. In section 4, we show the system performance with the change in the number of the centroids. The proposed method shows improved results in comparison with the conventional feature normalization methods in various codec environments. We can acquire the best performance when the number of the samples of the sum of the test samples (about 200 samples) and the supplement samples (about 300 samples) is about 500 according to Blanco's research [3].

We plan to use other clustering algorithms or feature selection algorithms to estimate CDF more robustly.

Acknowledgement. This research was supported by Basic Science Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2010-0024047).

References

1. Atal, B.S.: Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification. Journal of the Acoustical Society of America 55(6), 1304–1312 (1974)
2. Viikki, O., Laurila, K.: Cepstral domain segmental feature vector normalization for noise robust speech recognition. Speech Communication 25, 133–147 (1998)
3. Blanco, Y., Zazo, S., Principe, J.C.: Alternative Statistical Gaussianity Measure Using the Cumulative Density Function. In: Proceedings of the Second International Workshop on Independent Component Analysis and Blind Signal Separation, pp. 537–542 (2000)
4. Suh, Y., Ji, M., Kim, H.: Probabilistic Class Histogram Equalization for Robust Speech Recognition. Signal Processing Letters 14, 287–290 (2007)
5. Gonzalez, R.C., Wintz, P.: Digital Image Processing. Addision-Wesley, Reading (1987)
6. Pelecanos, J., Sridharan, S.: Feature warping for robust speaker verification. In: A Speaker Odyssey - The Speaker Recognition Workshop, pp. 213-218 (2001)
7. Skosan, M., Mashao, D.: Matching feature distributions for robust speaker verification. In: Proc. PRASA, pp. 42–47 (2004)
8. Skosan, M., Mashao, D.: Modified segmental histogram equalization for robust speaker verification. Pattern Recognition Letters 27(5), 479–486 (2006)
9. Segura, J.C., Benitez, C.A., Torre, A., Rubio, A.J., Ramirez, J.: Cpestral domain segmental nonlinear feature transformations for robust speech recognition. IEEE Signal Processing Letters 11, 517–520 (2006)
10. Torre, A., Peinado, A.M., Segura, J.C., Perez-Cordoba, J.L., Benitez, M.C., Rubio, A.J.: Histogram equalization of speech representation for robust speech recognition. IEEE Trans. Speech and Audio Processing 13, 355–366 (2005)
11. Reynolds, D.A., Quatieri, T.F., Dunn, R.B.: Speaker verification using adapted gaussian mixture models. Digital Signal Processing 10, 19–41 (2000)
12. Cannon, R.L., Dave, J.V., Bezdek, J.C.: Efficient Implementation of the Fuzzy c-Means Clustering Algorithms. IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-8, 248–255 (1986)
13. G.729: Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear prediction (CS-ACELP), `http://www.itu.int/rec/T-REC-G.729-200701-S/en`
14. SILK: Super Wideband Audio Codec, `https://developer.skype.com/silk`
15. Speex: A Free Codec for Free Speech, `http://www.speex.org/`

Unsupervised Segmentation for Different Types of Morphological Processes Using Multiple Sequence Alignment

Amit Kirschenbaum

Department for Natural Language Processing, University of Leipzig
Augustusplatz 10, D-04109 Leipzig, Germany
amit@informatik.uni-leipzig.de

Abstract. The aim of unsupervised and knowledge free morphological segmentation is the identification of boundaries between morphs in words of a given language without relying on any knowledge source about that language. This paper describes a segmentation method that draws on previous approaches based both on semantic and orthographical similarity to identify morphologically related words. Using a version of Multiple Sequence Alignment originally applied in bioinformatics, the method extracts both concatenative and non-concatenative (e.g. introflection and circumfixation) morphological patterns and can thus handle languages of different morphological types as well as non-dominant morphological processes within languages of a particular predominant morphological type.

Keywords: multiple sequence alignment, distributional similarity, unsupervised morphological segmentation.

1 Introduction

Morphology can be defined as "the study of the combinations of morphemes to yield words" [20]. Languages are traditionally classified according to their morphological structure where the basic types are analytical (or isolating) languages, and synthetic languages (the polysynthetic group is sometimes categorized as a different family and other times as a sub-group of the synthetic type). The synthetic group is further divided into agglutinative and fusional language types. However, phenomena that characterize one language type are typically present also in languages assigned to another type. For example, German, one of the languages we analyze with the current method, is usually classified as a fusional language i.e., a language in which several grammatical functions are expressed by one affix. However, in addition to the fusional elements, German also features elements from the other types: isolation, agglutination, polysynthesis.

Unsupervised methods address the problem of morphological analysis where the only source of knowledge is a raw text. The existence of languages of different morphological types makes this task especially challenging. Furthermore, the fact that languages constitute mixed morphological types, i.e., are not morphologically pure (see e.g., [30]), complicates the task even more. Consequently, only a method which can successfully handle all morphological types will be able to handle all morphological processes in a given language. However, many unsupervised methods use data structures which are suitable for morphological analysis of some languages types but less so for others. This can be demonstrated on concatenative vs. non-concatenative processes that appear in agglutinative and in introflective languages, respectively. In frequently explored languages (e.g., Dutch, English, Finnish, German, Turkish, in Morpho Challenge) affixes are mostly combined with their stems in a linear order i.e., in a concatenative manner. Phenomena that do not conform with this type of morphological process pose a problem to many algorithms especially if they rely on inherently contiguous data structures like tries (see Section 2). Such processes are, for instance, circumfixation (e.g., in German *spiel-en* 'to play', *ge-spiel-t* 'played' [past participle]) and introflexion (e.g. *werden - wird - wurde*). Non-concatenative languages like Hebrew and Arabic, in which introflexion presents a dominant morphological pattern, are challenging to the present unsupervised morphological segmentation. Due to the described complexity, the algorithms which do handle these languages, usually segment only the affixes which are added to the stem in a concatenative manner [34,31]. Similarly, when applied to languages with predominantly concatenative morphology, unsupervised methods often have difficulties handling their non-concatenative features [9].

The method presented in this paper enables the extraction of both linear and non-linear patterns and thus has the potential to deal successfully with phenomena like introflection and circumfixation. It is based on Multiple Sequence Alignment (MSA), an approach borrowed from bioinformatics, where it is used to discover biological relations among sequences of DNA, RNA, proteins, etc.

2 Related Work

A variety of schemes for unsupervised morphological analysis has been developed along the years. They differ in the definition of the the task (segmentation, finding morphologically related words, finding paradigms, etc.) and in the approaches for achieving these aims.

For the segmentation task, most of the approaches use a *list of words* to induce the morpheme boundaries. The earliest attempt of this kind, later improved by Hafer and Weiss [15], is due to Harris [18,19]. The algorithm, known as Letter Successor Variety, detects morpheme boundaries as a function of the number of distinct letters that follow, or precede, a letter sequence which is part of a word. If a peak is reached in that number, then it is assumed to be due to a morpheme boundary and a segmentation point is inserted.

Another type of algorithms which is used to derive morphological information utilizes *contextual information*. These methods, however, are focused on aspects of morphological analysis other than segmentation, e.g., finding morphologically related words, or deriving dependencies between morphological variants. Schone and Jurafsky [33] incorporate semantic cues for detecting morphologically related word pairs in English, German and Dutch. First, words are inserted into tries and potential affixes are determined by identifying the branching points in the trie. Pairs of candidate affixes are detected based on their common ancestor node in the trie and constitute affix rules. Pairs of potential morphological variants (PPMVs) are then defined as two words which share the same ancestor and the same affix rules. The method computes semantic vectors for each word using Latent Semantic Analysis [8] and compares the vectors of PPMVs to see how well these words correlate, in line with the notion that words which are morphologically related are also semantically related. Baroni et al. [2] also aim at finding pairs of morphologically related words in English and German, presupposing that orthographic and semantic similarities between pairs of words imply morphologically relatedness. The method constructs two lists: of orthographically related words, and of "semantically" related ones based on the mutual information of first-order co-occurrences. A combined score for morphological relatedness is computed for those pairs which exist in both lists. Freitag [12] creates clusters that correspond roughly to syntactic groups in English based on the mutual information between words and their immediate co-occurring words. The method then induces affix transformation rules which express relations between clusters and show possible affixation patterns. Bordag [3] enhances the LSV algorithm by employing contextual similarity to constrain the selection of candidates for affixes and stems and uses tries to learn prefixes and suffixes.

Contrary to the method presented in this paper most of the above mentioned methods using contextual information focus on inflectional morphology. Futhermore, a common practice to distinguish between stems and suffixes in the methods described above is using a trie. This data structure, however, is able to learn only concatenative processes. This poses a problem when non-concatenative processes like stem changes occur in the language, for example in German, where they are relatively few but occur frequently, or when non-concatenative processes constitute the dominant morphological pattern as in Semitic languages like Hebrew and Arabic. For this reason this approach is avoided in the present method. There are indeed approaches which do not rely on such structures and are able to discover also non-concatenative patterns. They use some data additional to the corpus and therefore cannot be classified as knowledge free, e.g., Hathout [21] used semantic features extracted from dictionary definitions, and Dreyer [10] incorporated sample paradigms as training data.

So far, the only method which employs *bioinformatics-inspired* approach for morphological segmentation is MetaMorph by Tchoukalov et al. [35]. This method starts with ordering words from a Hungarian corpus according to Levenshtein distance [27]: First the two most similar words from the 1000 most frequent words are added, and then, sequentially, a word from the rest of the corpus is added to the list if the word that is orthographically most similar to it is already on the list. The words are then aligned using Progressive Alignment [14] and the method creates a Profile HMM [11] such that each column of the alignment "acts as an HMM state whose character production probabilities correspond to the column's character distribution" [35, p.670], and the rest of the words are aligned to the Profile HMM. The algorithm first selects a set of columns in the alignment and then it segments all words of the corpus along this set of columns. To do so, the algorithm uses ParaMor-Morfessor Union system [26] and searches for a set of segmentation columns that maximizes the F-score against the ParaMor-Morfessor system. As known from bioinformatics [29], global MSA methods are effective for aligning sequences that are assumed to be biologically related, and providing inappropriate sequences to such method would not produce a meaningful result. Analogously, applying MSA to discover morphological patterns requires morphological relation among the aligned word forms. The MetaMorph algorithm, however, aligns words which are only orthographically similar, a criterion which is not sufficient to guarantee such relation. For a comprehensive survey of various other approaches and methods devoted to unsupervised morphological analysis, see [16].

3 Method

The method presented in this paper is based on a linguistic principle formulated, among others, by Bybee [5, p.118] that morphological relations exist among words which are related both semantically and phonologically. We assume that repeated patterns found among such related words correspond to morphological relations between them. The presented method detects the patterns, and segments the words accordingly. In the first step, the method finds a set of words distributionally similar to each input word. Then orthographically dissimilar words to input words are filtered out of these sets. The resulting list for each word form is then aligned by a MSA algorithm, and orthographical overlaps among these words are identified. Based on these overlaps, repeated patterns are extracted, and assumed morpheme boundaries are inserted.

3.1 Finding Semantically Related Words

In the first step, the method groups words which are distributionally similar to each input word. According to the distributional hypothesis [17], such words tend to be semantically similar. Words are therefore modeled in a high-dimension vector space, with each word form represented by a vector of weights. Each vector component reflects the significance of the co-occurrence of this word with another word in some context window. The number of random co-occurrences can be approximated to a Poisson distribution [24] under the independence assumption. The significance of a co-occurrence is computed by the Poisson collocation measure [32] which expresses the degree of surprise of a joint occurrence of two words. Following the method described in [4], the set of co-occurrents for each word is ordered in decreasing order of significance and only the M=200 with highest scores are retained. Dice's coefficient is then used as means of measuring the similarity between pairs of words by finding to what extent their respective resulting vectors agree.

3.2 Finding Phonologically Related Words

After a set of distributionally similar words had been retrieved for each input word, it is further narrowed down by filtering out phonologically dissimilar words. Phonology, in this case, is approximated by orthography. The orthographic difference between a given word and each of the distributionally similar words is computed with Needleman-Wunsch edit distance [28] with affine gaps penalties [13]. In this approach, different penalties are employed for opening a gap, as a result of inserting or deleting a character, and for extending one. We consider it more suitable for the given task than, e.g. Levenshtein edit distance [27] since it reduces the number of gap sequences (for example two words which share the same inflectional affixes should not be considered dissimilar only because their stem lengths are different). The Needleman-Wunsch [28] method is employed in bioinformatics to align a pair of biological sequences, finding a global alignment between the two sequences. In a biological context the cost of aligning one character with another is specified by similarity matrices, such as PAM [7] and BLOSUM [22] whose cell values reflect the likelihood of replacing one residue with another. Similarly, in the linguistic context it is likely that one character (or a string of characters) would replace another due to morphological processes, and that the replacing character (or string of characters) is thus related to the replaced one more than to the others.

In an unsupervised setting, however, the knowledge about relations between phonemes is not provided in advance, and the scoring scheme must be therefore uniform, without taking potential types of relations into account. Our setting assigns a positive score (+2) for character matches, and a negative score (−2) for mismatches. Gap opening is penalized with a cost of 4 and extending a gap costs 1. The maximum score is achieved when the two words are identical, and a minimum when all characters are mismatched and possible gaps are inserted when two words have different lengths. The resulting score is normalized, to a distance in the range $[0, 1]$:

$$d = \frac{max\_score - score}{max\_score - min\_score} \tag{1}$$

Words which are relatively dissimilar, with respect to orthography, from the input word ($d > 0.5$) are then removed from the above mentioned set.

3.3 Extracting Patterns through Multiple Sequence Alignment

MSA is a natural extension to pairwise alignment, and is used to find conserved regions within several sequences. In a bioinformatics context, MSA is applied to collections of sequences which are assumed to be related, and the conserved residues playing a functional or structural role in these relations [29]. A widely used heuristic of aligning multiple sequences is to align the sequences progressively; first the two most similar sequences are aligned and then less similar ones are aligned in a cumulative way, producing intermediate alignments, to construct the final alignment. In the context of morphological segmentation, selected sets of distributionally and orthographically similar words (as described above) are treated as sequences that are to be aligned. The first sequence of the alignment is the input word. The similarity criterion in this case is the similarity to the input word. For the alignment purpose we used BioJava [23], and modified the sequence alignment procedure to match our task.

Next, a pattern for the aligned sequences is to be found. The current method performs a series of pairwise comparisons of the aligned sequences, extracting identical fragments from each aligned pair, as a candidate pattern for the alignment. Each candidate pattern is stored with the number of corresponding sequences with which it matches. An aligned sequence can, of course, match more than one candidate pattern. Table 1 demonstrates how an alignment set for an input word looks, in this case for the word *Ladenöffnungszeiten* 'shop opening-hours'. Gaps (-) are inserted during the alignment process so that all sequences have the same length.

Table 1. An example for an alignment

`----ladenöffnungszeiten`
`---------öffnungszeiten`
`----ladenöffnung-------`
`---------öffnung-zeiten`
`----ladenschluß--zeiten`
`---------öffnungszeit--`
`---betreu----ungszeiten`
`ausbild------ungszeiten`

As can be observed, several patterns can be extracted from this alignment e.g., `-öffnungszeiten`, `ladenöffnung-`, `laden-zeiten`, etc.

Two scoring methods for candidate patterns were designed to choose the pattern that best matches an alignment:

Method A. This method attempts to achieve balance between candidate patterns that match many sequences and those that are long. The score for assigned for $pattern_i$ in an alignment is given by the following harmonic mean:

$$score(pattern_i) = \frac{2}{\frac{1}{count(pattern_i)} + \frac{1}{length(pattern_i)}} , \qquad (2)$$

where $count(pattern_i)$ is the number of times $pattern_i$ was found among pairs of aligned sequences and $length(pattern_i)$ is its length in characters.

After scoring each of the candidate patterns, the one with the highest score is selected as the one which describes best the alignment members, and those which match it are segmented accordingly. The segmented forms are then recorded together with the score (hereinafter "local" score) of the selected pattern.

Method B. This method uses the relative frequency for each candidate pattern compared to the other ones, weighted as a function of the alignment size, to score each of the patterns extracted from an alignment. Using the size of the alignment provides complementary information to the relative frequency, which has only local nature, so that a pattern covers a large set of sequences in a bigger alignment would have a higher score. The score is for $pattern_i$ given by the following formula:

$$score(pattern_i) = \frac{count(pattern_i)}{\sum_j count(pattern_j)} \log(size) \ , \tag{3}$$

where $count(pattern_i)$ is the number of times $pattern_i$ was found in the pair-wise comparison of aligned sequences and $size$ is the total number of sequences participated in the alignment. The selected pattern for an alignment is again the candidate pattern which achieved the highest score. The segmented form along with this local score are recorded as in method A.

Since a word can occur as an input word and as a member in an alignment set of some other input word, it may end up, in both methods, with several options for segmentation. A list of possible segmentations ranked according to their scores is created in order to select the most probable correct segmentation of each word. Two ranking options were explored. One is frequency-based, and the other is score based. In the former case the segmentation score is determined solely by the number of cases where this segmentation was selected for the given word form, and in the latter case, each segmentation option gets a “global” score which is the sum of the “local” scores, and the list of segmentation is ranked according to these scores.

4 Resources

The method was applied to German, English, and Hebrew. The German and the English corpora were part of data sets available at the Morpho Challenge 2009 contest[1] and were originally obtained from the Wortschatz[2] collection. Each of the corpora consists of three million plain text sentences. The corpora are tokenized and lower-cased. The evaluation of word segmentation was performed against CELEX lexical database [1], which consists of a list of word forms and their lemmas that are analyzed morphologically. The CELEX word forms were pre-processed to provide a segmented version for each of them, based on the lemma analysis in this database.

For Hebrew we used the tokenized version of the MILA Arutz7 corpus, a collection of news items and articles [25], comprising of 780,269 sentences. To the best of our knowledge there is no morphological segmentation corpus for modern Hebrew. We used a tagged sub-corpus of the MILA Arutz7 (containing 106,492 sentences) to extract the segmentations based on disambiguated morphological analyses for each word.

[1] `www.cis.hut.fi/morphochallenge2009/`

[2] `corpora.informatik.uni-leipzig.de/`

5 Evaluation and Results

5.1 Evaluation Measures

For English and German, the segments of the analyzed word forms were compared to those extracted from CELEX. Correctly found boundaries were classified as true positives (*tp*), boundaries which were found by the method but did not match the CELEX boundaries were classified as false positives (*fp*), and CELEX boundaries which were not detected by the method were classified as false negatives (*fn*). For Hebrew, the evaluation was performed analogically with the gold standard based on the MILA Arutz7 corpus as described above. Precision and recall were then calculated for each word w, based on the number of boundaries in each of the above categories:

$$Precision_w = \frac{\#tp}{\#tp + \#fp} \quad Recall_w = \frac{\#tp}{\#tp + \#fn} \ . \tag{4}$$

The average precision (P) and recall (R) are calculated based on the results for each word.

A ranked list of segmentation options is calculated both for both methods A and B, based on either the frequency of the those options or on the accumulated scores resulting from the pattern scores.

The evaluation results for German are presented in Table 2 on 49748 word forms which were found in the gold standard; for English in Table 3 on 23545 word forms which had entries in the gold standard; and for Hebrew in Table 4 which was evaluated on 87346 entries.

We report the results for top-1, top-2 and top-5 segmentation options. To have a comparison with a state of the art system we report the results of Morfessor [6] applied to the sets of words analyzed by our method. As can be seen, taking the scores into account to rank segmentations is in general better than relying on their raw frequency. This indicates that the score provides better insight into the quality of the patterns resulting from individual alignments. It reflects the fact that patterns derived from longer alignments and matching most words in the alignment provide more reliable results.

A detailed look at the data reveals that complex processes can be handled successfully by the presented method: Multimorphemeic words can be split into all their components which may include circumfixation in addition to continguos morphemes e.g., ein-ge-jag-t (*Angst einjagen* ‘to frighten’; ein:prefix, ge-t:circumfix, jag:stem). The method is able to detect stem vowel changes (introflection) as in the irregular verbs in German d-a-rf (*dürfen* ‘may’); k-ä-m-e (*kommen* ‘to come’); s-e-nd-e, ein-ge-s-a-nd-te-n, zu-ge-s-a-nd-te-n (*senden* ‘to send’). However, though these segmentations represent correct solutions with respect to the involved introflective processes, they are evaluated as incorrect since they are not represented in the CELEX gold standard.

Hebrew is a language with templatic morphology. A consonantal root carries the core semantics and when inserted into a derivational or inflectional pattern, a particular word form is created. Moreover, function words, as definite

Table 2. Results for German

	top-1		top-2		top-5	
	P	R	P	R	P	R
A						
freq	0.476	0.412	0.520	0.467	0.524	0.470
score	0.483	0.447	0.556	0.535	0.578	0.564
B						
freq	0.507	0.407	0.575	0.482	0.587	0.498
score	0.502	0.448	0.613	0.576	0.665	0.644
Morf.	0.637	0.478				

Table 3. Results for English

	top-1		top-2		top-5	
	P	R	P	R	P	R
A						
freq	0.411	0.489	0.464	0.540	0.471	0.545
score	0.378	0.490	0.463	0.581	0.499	0.608
B						
freq	0.500	0.512	0.578	0.600	0.591	0.617
score	0.473	0.517	0.588	0.648	0.644	0.706
Morf.	0.585	0.610				

Table 4. Results for Hebrew

	top-1		top-2		top-5	
	P	R	P	R	P	R
A						
freq	0.575	0.557	0.640	0.630	0.649	0.639
score	0.581	0.586	0.669	0.696	0.700	0.730
B						
freq	0.487	0.374	0.688	0.554	0.721	0.607
score	0.519	0.411	0.750	0.638	0.826	0.770
Morf.	0.629	0.687				

articles, conjunctions, or prepositions, which are usually distinct words in other languages are attached to the Hebrew word forms enhancing them as prefixes, while possessive markers enhance them as suffixes. Table 5 presents some examples which demonstrate correctly segmented word forms, resulting from the proposed method, that capture both concatenative and non-concatenative processes in Hebrew. The leftmost column presents the segmented word form, the middle column presents the root of the word form along with the translation of its core semantics, and the rightmost column presents a translation for the word form which includes morphemic information.

Table 5. Hebrew wordforms segmented correctly

seg. worform	root (core meaning)	gloss
h-ar-i-k-h	a-r-k (length)	prolonged (3SG.F)
h-b-w-gr-im	b-g-r (mature)	the graduates (M)
m-brk-im	b-r-k (bless)	bless (PRS.M.PL)
h-rby-wn	r-b-y (four)	the quartile

The comparison between the top-1 and top-2 (and top-5) results reveals that the method has further potentials for raising precision and recall for top-1, if we improve the mechanism of selecting the best pattern. In many cases, the top-1 option stayed under-segmented (often unanalyzed), and the second best choice corresponded to the correct analysis.

6 Conclusions

We have presented an unsupervised method for morphological segmentation which utilizes MSA for this task. The model is based on the principle that morphological relations exist among words which are semantically and phonologically related and constructs groups of such words following this principle. These groups are then aligned using Multiple Sequence Analysis, a method borrowed from bioinformatics where it used for finding biologically conserved regions.

To our knowledge, the method we present is the first relying solely on the alignment resulting from MSA to perform morphological segmentation. The advantage of this approach is that it can successfully deal with both concatenative and non-concatenative morphological features, enabling adequate segmentation of various language types as well as of non-contiguous patterns like introflection and circumfixation which are typically ignored by other algorithms, or dealt with in separate modules. There are several ways to improve the method. Method A and Method B are based on different principles. While Method A takes into account the length of the shared segment(s) and the number of words that match it, Method B considers the relative frequency of a given pattern and the overall size of the alignment. We assume that a method taking both of these aspects into account could yield better results. Furthermore, stricter approach to determining the orthographic similarity, or using a different method for finding orthographically similar words could improve the precision.

We believe that ability of the method to handle morphologically different processes is a very important property, which distinguishes it from the other so far developed systems for morphological segmentation.

References

1. Baayen, R.H., Piepenbrock, R., Gulikers, L.: The CELEX lexical database (release 2). CD-ROM (1995)
2. Baroni, M., Matiasek, J., Trost, H.: Unsupervised discovery of morphologically related words based on orthographic and semantic similarity. In: Proceedings of the ACL 2002 Workshop on Morphological and Phonological Learning, pp. 48–57 (2002)
3. Bordag, S.: Unsupervised and knowledge-free morpheme segmentation and analysis. In: Peters, C., Jijkoun, V., Mandl, T., Müller, H., Oard, D.W., Peñas, A., Petras, V., Santos, D. (eds.) CLEF 2007. LNCS, vol. 5152, pp. 881–891. Springer, Heidelberg (2008)
4. Bordag, S.: A Comparison of Co-occurrence and Similarity Measures as Simulations of Context. In: Gelbukh, A. (ed.) CICLing 2008. LNCS, vol. 4919, pp. 52–63. Springer, Heidelberg (2008)

5. Bybee, J.L.: Morphology: A Study of the Relation between Meaning and Form. Typological Studies in Language, vol. 9. John Benjamins Publishing Company (1985)
6. Creutz, M., Lagus, K.: Unsupervised Morpheme Segmentation and Morphology Induction from Text Corpora Using Morfessor 1.0. Tech. Rep. Report A81, Helsinki University of Technology (March 2005)
7. Dayhoff, M.O., Schwartz, R.M.: A model of evolutionary change in proteins. Atlas of protein sequence and structure 5(suppl. 3), 345–358 (1978)
8. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. Journal of the American Society for information science 41(6), 391–407 (1990)
9. Demberg, V.: A language-independent unsupervised model for morphological segmentation. In: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, pp. 920–927 (2007)
10. Dreyer, M.: A Non-Parametric Model for the Discovery of Inflectional Paradigms from Plain Text Using Graphical Models over Strings. Ph.D. thesis, Johns Hopkins University (2011)
11. Durbin, R., Eddie, S.R., Lrogh, A., Mitchinson, G.: Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids. Cambridge University Press (1998)
12. Freitag, D.: Morphology induction from term clusters. In: Proceedings of the Ninth Conference on Computational Natural Language Learning, CONLL 2005, pp. 128–135 (2005)
13. Gotoh, O.: An Improved Algorithm for Matching Biological Sequences. Journal of Molecular Biology 162(3), 705–708 (1982)
14. Gotoh, O.: Significant Improvement in Accuracy of Multiple Protein Sequence Alignments by Iterative Refinement as Assessed by Reference to Structural Alignments. Journal of Molecular Biology 264, 823–838 (1996)
15. Hafer, M.A., Weiss, S.F.: Word segmentation by letter success varieties. Information Storage and Retrieval 10, 371–385 (1974)
16. Hammarström, H., Borin, L.: Unsupervised Learning of Morphology. Computational Linguistics 37(2), 309–350 (2011)
17. Harris, Z.S.: Distributional Structure. Word 10(2/3), 146–162 (1954)
18. Harris, Z.S.: From phoneme to morpheme. Language 31, 190–222 (1955)
19. Harris, Z.S.: Morpheme boundaries within words: Report on a computer test. In: Transformations and Discourse Analysis Papers. Department of Linguistics, University of Pennsylvania (1967)
20. Haspelmath, M.: Understanding morphology. Arnold London (2002)
21. Hathout, N.: Acquistion of the morphological structure of the lexicon based on lexical similarity and formal analogy. In: Proceedings of the 3rd Textgraphs Workshop on Graph-Based Algorithms for Natural Language Processing, pp. 1–8. Association for Computational Linguistics (2008)
22. Henikoff, S., Henikoff, J.G.: Amino acid substitution matrices from protein blocks. Proceedings of the National Academy of Sciences 89(22), 10915–10919 (1992)
23. Holland, R.C.G., Down, T.A., Pocock, M.R., Prlic, A., Huen, D., James, K., Foisy, S., Dräger, A., Yates, A., Heuer, M., Schreiber, M.J.: BioJava: an open-source framework for bioinformatics. Bioinformatics 24(18), 2096–2097 (2008)
24. Holtsberg, A., Willners, C.: Statistics for sentential co-occurrence. Lund Working Papers in Linguistics 48, 135–147 (2001)
25. Itai, A., Wintner, S.: Language resources for Hebrew. Language Resources and Evaluation 42(1), 75–98 (2008)

26. Kurimo, M., Virpioja, S., Turunen, V.T., Blackwood, G.W., Byrne, W.: Overview and Results of Morpho Challenge 2009. In: Peters, C., Di Nunzio, G.M., Kurimo, M., Mandl, T., Mostefa, D., Peñas, A., Roda, G. (eds.) CLEF 2009. LNCS, vol. 6241, pp. 578–597. Springer, Heidelberg (2010)
27. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. Doklady Akademii Nauk SSSR 163(4), 845–848 (1965)
28. Needleman, S.B., Wunsch, C.D.: A general method applicable to the search for similarities in the amino acid sequence of two proteins. Journal of Molecular Biology 48(3), 443–453 (1970)
29. Notredame, C.: Recent progresses in multiple sequence alignment: a survey. Pharmacogenomics 3(1) (2002)
30. Pirkola, A.: Morphological typology of languages for ir. Jounral of Documentation 57(3) (2001)
31. Poon, H., Cherry, C., Toutanova, K.: Unsupervised morphological segmentation with log-linear models. In: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 209–217 (2009)
32. Quasthoff, U., Wolff, C.: The Poisson collocation measure and its applications. In: Proceedings of the Second International Workshop on Computational Approaches to Collocations (2002)
33. Schone, P., Jurafsky, D.: Knowledge-free induction of inflectional morphologies. In: Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics on Language Technologies, NAACL 2001, Stroudsburg, PA, USA (2001)
34. Snyder, B., Barzilay, R.: Unsupervised multilingual learning for morphological segmentation. In: Proceedings of ACL 2008: HLT, pp. 737–745 (2008)
35. Tchoukalov, T., Monson, C., Roark, B.: Morphological Analysis by Multiple Sequence Alignment. In: Peters, C., Di Nunzio, G.M., Kurimo, M., Mandl, T., Mostefa, D., Peñas, A., Roda, G. (eds.) CLEF 2009. LNCS, vol. 6241, pp. 666–673. Springer, Heidelberg (2010)

Cross-Lingual Random Indexing for Information Retrieval

Hans Moen and Erwin Marsi

Norwegian University of Science and Technology (NTNU),
Dept. of Computer and Information Science,
Trondheim, Norway
{hans.moen,emarsi}@idi.ntnu.no

Abstract. Cross-lingual information retrieval aims at retrieving relevant documents from a document collection in a language different from the query language. A novel method is proposed which avoids direct translation of queries by implicit encoding of translations in a bilingual *vector space model* (VSM). Both queries and documents are represented as vectors using an extension of *random indexing* (RI). As work on RI for information retrieval is limited, it is first evaluated for monolingual retrieval. Two variants are tested: (1) a *direct* RI model that approximates a standard VSM; (2) an *indirect* RI model intended to capture latent semantic relations among terms with a sliding window procedure. Next cross-lingual extensions of these models are presented and evaluated for cross-lingual document retrieval.

1 Introduction

In the classic *vector space model* (VSM) for *information retrieval* (IR) [26,17], both documents and queries are represented as vectors in a high-dimensional vector space. Each dimension represents term counts and terms are usually weighted using some variant of TF*IDF [10]. Relevant documents are retrieved by computing the cosine similarity between a query vector and the document vectors, retrieving the n most similar documents. A limitation of the standard VSM is that it cannot cope with semantically related terms, for example, synonyms. This was part of the motivation for *latent semantic indexing* (LSI), which uses dimensionality reduction as a means of accessing latent distributional similarities between terms [7]. Evidence for the claim that LSI improves IR seems open to interpretation. Initial evaluations suggested that LSI can improve results on certain benchmark data sets; see [3] for a summary of findings. However, more recent experimental results on a larger scale suggested otherwise [1].

Regardless of whether LSI improves retrieval or not, there is no dispute that it is computationally expensive. The core of LSI is truncated *singular value decomposition* (SVD), a mathematical operation for reducing a matrix that presumably captures higher order relations between terms. The computational cost of truncated SVD makes it hard to scale LSI to large document collections. *Random indexing* (RI), an iterative indexing method based on the principle of sparse distributed memory [11], was initially proposed as a simpler and cheaper alternative to LSI [12,23]. It is argued to deliver comparable results at a much lower computational cost. In addition, it is fully incremental, allowing addition of new documents without the need to recompute the existing model (as in LSI). It was initially evaluated for learning synonyms in a TOEFL test [12], measuring word similarity through distributional similarity, that is, through a statistical analysis of word co-occurrence frequencies in large text corpora. Since then it has been applied to a range of tasks with generally positive results [13,24]. In general, smoothing methods like LSI and RI are thought to promote a number of desirable properties in models of distributional similarity, including revealing latent meaning, reducing noise, capturing high-order co-occurrence relations, and reducing sparsity [29].

Given that LSI is claimed to improve upon the classic VSM, and that RI is claimed to be a comparable but cheap alternative to LSI, it is a logical step forward to evaluate RI in an IR context. There seem to be few studies on this. [22,5] explore RI in combination with *holographic reduced representations* (HRR). [30] use a extension of RI called *reflective random indexing* (RRI) for classifying MEDLINE articles. [2] use RI as word discrimination method in an IR task, and compare it to a word disambiguation method. [28] report results on combining RI and LSI for IR. Still, no good conclusion is given when it comes to the performance of using RI as a document index for IR. A recent review article about distributional semantics in the biomedical domain states: "To the best of our knowledge Random Indexing has not been extensively evaluated in an information retrieval context, presenting a research opportunity for its formal evaluation in the context of information retrieval from MEDLINE" [6]. The first contribution of this paper is therefore to add new empirical results on monolingual IR with RI.

Cross-lingual information retrieval (CLIR) aims at identifying relevant documents in a language other than that of the query [14]. Most approaches start with translating the query to the target language using bilingual dictionaries or machine translation systems. This raises the familiar problems in machine translation such as lack of lexical coverage and lexical translation ambiguity. Other approaches require bilingual data in the form of parallel text aligned at the word, sentence or document level. For instance, [8] propose a bilingual LSI model that requires pairs of documents and their translations for training. In contrast to existing approaches, we propose a new method called *cross-lingual random indexing* that avoids direct translation of the query. Instead translation is implicitly encoded in a RI model. There is no need for aligned bilingual text either, only a bilingual dictionary and monolingual corpora for both languages. The second contribution of this paper is therefore a new model for CLIR and its experimental evaluation.

The remainder of this paper has two major parts: random indexing for monolingual retrieval and (2) cross-lingual RI for bilingual retrieval. It concludes with a general summary of findings and an outlook on future work.

2 Monolingual Information Retrieval with Random Indexing

2.1 Direct Random Indexing

Conceptually, random indexing can be regarded as a method for compressing a standard term–document or term–term matrix, where rows (vectors) represent documents, columns represent unique terms and cells represent how many times a certain term occurs in a certain document. In practice, RI directly generates a matrix of reduced dimensionality through the following procedure:

1. Each term in the document collection gets a unique *index vector*. Index vectors are high-dimensional, but typically of substantially lower size than the total number of unique terms. These are very sparse randomly initiated vectors containing mostly zeros, apart from a few randomly chosen 1s and -1s. As a result these index vectors becomes "nearly orthogonal" to each other in the vector space.
2. Each document is then represented by a *document vector* obtained by summing the index vectors of all terms occurring in the document. This optionally includes term weighting and vector length normalization.

As a result of this procedure, documents containing the same terms have vectors composed of the same index vectors and are therefore more similar in the vector space. The vector for a query likewise is constructed by summing the (weighted) index vectors of all its terms.

2.2 Indirect Random Indexing

Indexing a text corpus with *sliding window* RI takes a somewhat different approach [13]. Instead of directly summing the index vectors of a document's terms, there is an intermediate step that first creates *term context vectors*. Indirect RI involves the following steps:

1. Each unique term in the document collection gets a unique *index vector*.
2. Next a *context vector* is generated for each term. The document collection is scanned by sliding a fixed-size window over the text, term by term. Each step, the context vector of the term in the center of the window – often referred to as the *target term* – is updated by adding the index vectors of the neighboring terms within the window. As a result, terms co-occurring with similar terms obtain similar context vectors in the vector space.
3. Context vectors are normalized by dividing them by the global frequency of the term in the document collection.
4. Each document is then represented by a *document vector* obtained by summing the context vectors of all its terms, optionally including term weighting and vector length normalization.

This method thus models higher-order co-occurrence relations among terms, captured through analyzing local co-occurrence relations among words. In addition, there are methods for encoding word order relations within the sliding window. These options and other experimental variables are detailed in the next section.

2.3 Experimental Setup

We adopted the well-established CLEF framework for evaluation of cross-lingual information retrieval, in particular, the ad hoc monolingual and bilingual tracks from CLEF 2005 [4]. Monolingual experiments address English, whereas bilingual experiments concern German and English as source and target language respectively. This choice was primarily prompted by our access to CLEF 2004-2008 data, as well as to a relatively large German-English translation dictionary. The CLEF data consists of three components: document collections, search topics, and relevance judgement; see [4] for details.

Document collections comprise news text from news wires, newspapers and opinion magazines. Stopword removal and lemmatization were applied as this was found to generally improve the IR scores. The full English corpus consist of 257,130 documents with approximately 130M words, and among these 325,617 unique ones after lemmatization. After stopword removal, the corpus is reduced to 70M words, with 325,392 unique words. All documents were used for training, and a subset of 169,477 was used as retrievable documents in the experiment. Documents were lemmatized with TreeTagger. Stopwords were removed using customized versions of the default stopword lists provided by the Lucene project [16]. Terms occurring only once were also removed. For all remaining terms, TF*IDF values were calculated [10], and used for weighting terms, i.e. their context vectors, when creation of document vectors.

Topics express the informational need of a user and consist of three fields: (1) a brief *title* stating the main keywords, (2) a single sentence *description* of the concept conveyed by the keywords, and (3) a more elaborate narrative. All experiments in this paper used the combination of *title* and *description* to create a query.

Relevance judgements specify which documents from the document collection are relevant to a particular topic. Documents are assessed as either relevant or irrelevant to the topic by a panel of human judges.

The RI algorithms used for the experiments in this paper are based on the JavaSDM package [9]. Scores are calculated using the `trec_eval` tool (version 7.3). Results are reported in terms of mean average precision (MAP) together with the total number and percentage of relevant documents retrieved over all 50 queries. For comparison we used Apache Lucene [16] (v4.1.0), a state-of-the-art search engine implementing a TF*IDF weigthed variant of the standard VSM. No additional weighting or "boosting" of specific sections or fields in the documents or queries were applied.

Experiments explored a number of different configurations. The first two parameters concern the RI model itself:

Dimensionality. The size of the vectors (index and context vectors) ranged from 1000 through 1800 to 4000.

Non-zeros. The total number of 1's and -1's randomly assigned to the index vectors.

In addition, there were two parameters that only apply to Indirect RI:

Window Size. The size of the sliding window ranged from 2+2 (i.e. two words on the left and two words on the right of the target term) up to 20+20.

Weighting Scheme. Index vectors of the neighboring terms in the sliding window are weighted and/or modified before they are added to a context vector. *distance weighting* uses the function $2^{1-distance}$, *distance* being the distance in words to the target term [13]. *Random permutations* (RP) [25] encode word order relations by shifting the elements in the index vectors according to both their position and their distance from the target term. In a similar fashion, *Direction vectors* only encode direction by shifting index vectors once, either left or right depending on which side of the target term they are located [25], plus weighting the vectors similarly as in Distance Weighting.

2.4 Results

Lucene retrieved 1817 relevant documents (88.08%), resulting in a a MAP score of 0.3713. Table 1 presents corresponding results for direct RI, indicating that about 64–72% of the relevant documents were found, with a MAP score in the range from 0.15–0.18. Increasing the number of non-zeros up till 8 was found to improve results while changing dimensionality had no effect.

Table 2 presents selected results for Indirect RI. Vector dimensionality does not affect the results beyond a certain size, approximately around 2000. The number of non-zeros also has little effect, less so than in the Direct RI experiments. Larger window sizes appear to yield better results than smaller sizes. Weighting schemes do not have any positive effect, suggesting that word order within the window is irrelevant. Smaller window sizes were tested for the other weighting schemes, but none of these performed better than without weighting. In sum, a medium vector dimensionality (1800) together with a large window size (16+16), unweighted, and few non-zeros (4) gave the best performance.

2.5 Discussion

The direct RI method is essentially an approximation of the standard TF*IDF-weighted VSM. However, where the VSM would have a dimensionality equal to the number of unique terms in the document collection (e.g. 325,617 for English), direct RI has just 1800, which amounts to approximately 2% of the size. This may explain why Direct RI scores lower than what may be expected from a standard VSM, here represented by Lucene.

Table 1. Results with direct random indexing for monolingual (English) ad hoc information retrieval track from CLEF 2005

Dimensions	Non-zeros	MAP	Found/2063	%Found
1800	2	0.1512	1340	64.95
1800	4	0.1769	1427	69.17
1800	**8**	**0.1839**	**1481**	**71.79**

Table 2. Results with indirect random indexing for monolingual (English) ad hoc information retrieval track from CLEF 2005

Dim.	Non-zeros	Window	Weighting	MAP	Found/2063	%Found
1800	4	2+2	No weighting	0.1411	1238	60.01
1800	4	4+4	No weighting	0.1722	1316	63.79
1800	4	8+8	No weighting	0.1920	1387	67.23
1800	4	12+12	No weighting	0.1965	1415	68.59
1800	4	**16+16**	No weighting	0.1987	**1426**	**69.12**
1800	4	20+20	No weighting	0.1984	1420	68.83
1800	2	16+16	No weighting	0.1954	1413	68.49
1800	**4**	16+16	No weighting	0.1987	**1426**	**69.12**
1800	8	16+16	No weighting	0.1965	1400	67.86
1000	4	16+16	No weighting	0.1961	1400	67.86
1800	4	16+16	No weighting	0.1987	**1426**	**69.12**
4000	4	16+16	No weighting	**0.1998**	1411	68.40
1800	4	16+16	Rand. Permutations	0.1422	1067	51.72
1800	4	16+16	Direction Vectors	0.1391	1221	59.19
1800	4	16+16	Dist. weighting	0.1477	1286	62.34
1800	4	16+16	**No weighting**	0.1987	**1426**	**69.12**

We also find that indirect RI achieves slightly better mean average precision than direct RI, suggesting a better ranking among the top 1000 retrieved documents, whereas direct RI yields better recall. This finding is in agreement with the conclusions in [22]. Differences are small though (2.67%) and this may therefore cast some doubt on the claim that the sliding window variant captures latent relations between terms. Alternatively, it may be interpreted as an indication that modeling latent semantic information does not consistently improve the IR results. In fact, some recent studies suggest that LSI also yields poor retrieval accuracy on a large number of TREC bench mark sets [1].

3 Cross-Lingual Information Retrieval with Random Indexing

3.1 Method

The core idea in the method for cross-lingual RI proposed here is that source and target language models share the same vector space. In this way, the vector representation of a query stated in the source language can be compared directly to the vector representation of documents in the target language. This removes the need for any explicit translation, as term translations and cross-lingual synonymy are implicitly encoded in the vector space. This is accomplished through a sharing of index vectors across languages during the random indexing procedure, so that terms that are translations of each other share a common index vector. Two variants of direct and indirect cross-lingual RI based on this idea are detailed below.

As a baseline for comparison, we use the dictionary to translate the queries, translating each source term into the corresponding $top-N$ most frequent target terms according to the TL corpus. In addition, terms not in the dictionary are simply copied over, assuming a lot of these are proper nouns. These translated queries are then used by Lucene for monolingual IR in the TL.

Direct Cross-Lingual Random Indexing. The method for cross-lingual direct RI is almost the same as for monolingual direct RI (cf. Section 2.1), except for one crucial modification in the first step, where index vectors are shared across languages. This assumes a translation dictionary mapping source language terms to target language terms, with one-to-many mappings in the case of translation ambiguity. First, a unique index vector is generated for each source term in the dictionary. Next, each target term gets the same index vector as its corresponding source term. If a target term serves as the translation of multiple source terms, their index vectors are merged with disjunction. The second step of creating query and document vectors is the same as for monolingual RI.

Indirect Cross-Lingual Random Indexing. As in the direct cross-lingual case, index vectors are again shared among source terms and their translations. Source language and target language document collections are then processed independently using the sliding window procedure to build term context vectors for source and target language terms respectively (step 2), followed by frequency correction (step 3). Notice that documents are not aligned in any way and are in fact completely unrelated. Finally, (multilingual) document vectors are obtained by summing the context vectors of all target terms contained in the document, whereas query vectors are constructed by summing vectors for their source terms.

A variant of this approach includes an extra step following the construction of the term context vectors. For each context vector of a source term, we add to it all the context vectors of its translations. Conversely, for each context vector of a target term, we add to it all the context vectors of the source terms it is a translation of. The resulting enriched context vectors will be referred to as *translation vectors*. The reasoning behind this operation is that translation vectors presumably encode second-order translation relations. That is, a pair of vectors representing source and target language texts is not only similar when the texts contain terms that are translations of each other, but also when the texts contain terms co-occurring with terms that are in turn translations of each other. This is akin to query expansion through related terms used to improve recall.

3.2 Experimental Setup

A proprietary German-English translation dictionary was used in the process of constructing index vectors. It is lemma-based, provides part-of-speech (POS) tags on both source and target side, and contains over 576k entries. In experiments we only used single-word expressions, leaving out the multi-word expressions, which did not seem to be benificial.

Cross-lingual experiments were based on the bilingual ad hoc retrieval track using 50 German topics to retrieve English documents. The German topics corresponded to the English topics used earlier in the monolingual experiments; the English document collection was the same as before (cf. Section 2.3). However, the use of a translation dictionary imposed some additional constraints. First, the dictionary entries are lemma-based, so for the purpose of look-up, the document collections were lemmatized with TreeTagger using pre-trained models for English and German [27]. Two variations where tested, one including out-of-dictionary terms during training, and one where terms were limited to those in the dictionary. For the latter, this reduced the number of unique English terms from 325,617 to 114,645 and the total number of indexed terms in the English document collection from approximately 130M down to 70M (after stopword removal). Likewise, the number of unique German terms was reduced from 1,057,526 to 144,766 and the total number of indexed terms from about 80M down to 36M.

Model parameters were adopted from the best scoring configurations in the monolingual IR experiment presented earlier: a vector dimensionality of 1800, 4 non-zeros for index vectors, and a unweighted window of 16+16 in indirect RI.

3.3 Results

Table 3 shows results for applying the cross-lingual random indexing method to the *bilingual* ad hoc IR track. These scores are clearly a lot lower than the monolingual scores, with direct RI again outperforming indirect RI in terms of MAP scores. However, the variant of indirect RI employing *translation vectors* performs best in terms of recall. The latter was also tested using out-of-dictionary terms, resulting in lower recall but higher MAP. Unfortunately none of the RI methods were able to beat the baseline relying on a two-step approach of query translation using the dictionary followed by monolingual IR with Lucene. As shown in Table 4, best MAP and recall scores were obtained by taking the two or three most frequent translations respectively.

Table 3. Results with cross-lingual random indexing for bilingual (German-English) ad hoc information retrieval track from CLEF 2005

Method	MAP	%Mono	Found/2063	%Found
Direct Cross-lingual RI	**0.0667**	**36.27**	592	28.70
Indirect Cross-lingual RI	0.0176	8.56	400	19.39
Translation vectors limited to dictionary	0.0501	25.21	**767**	**37.18**
Translation vectors not limited to dictionary	0.0656	33.02	659	31.94

Table 4. Results for combination of query translation and Lucene on bilingual (German-English) ad hoc information retrieval track from CLEF 2005

Method for query translation	MAP	%Mono	Found/2063	%Found
Dictionary ranked top1	0.1436	38.68	978	47.41
Dictionary ranked top2	**0.1541**	41.50	1068	51.77
Dictionary ranked top3	0.1437	38.70	**1091**	**52.88**
Dictionary ranked top4	0.1347	36.28	1073	52.01
Dictionary ranked top5	0.1275	34.34	1036	50.22

3.4 Discussion

Among the participants in CLEF 2005, none of them submitted any results for English-German in the ad hoc bilingual track. However, three teams targeted English from other source languages. University of Glasgow submitted results for Greek-English [15]. After query expansion, Greek lemmas were automatically translated into English with Yahoo's Babelfish, a full fledged MT system. The best results were obtained with the classic BM25 model [21] with empirically tuned parameter. They achieved a MAP score of 0.2935, 68.14% of their reported monolingual score. Johns Hopkins University worked on Greek-English, Hungarian-English and Indonesian-English, aiming for a language-independent solution based on character n-grams [18]. Queries were expanded prior to translation using the source language CLEF corpus. Next, queries were translated using online translation services: Yahoo's Babelfish for Greek, ToggleText's Kataku for Indonesian and TranslationExpert's InterTran for Hungarian. A statistical language model was employed for retrieval. They achieved MAP scores of 0.2418 (54.94%) for Greek, 0.3728 (84.71%) for Indonesian, and 0.1944 (44.17%) for Hungarian. University of Indonesia reported results for Indonesian-English. Queries were first translated using Transtool, a commercial MT system. Retrieval relied on VSM using the Lucene IR system, with a best MAP score of 0.1830 (52.16%).

Scores obtained with the cross-lingual RI methods are thus relatively low compared with other approaches using generic MT systems for translating the query prior to monolingual retrieval. We believe that the same issues that make the RI model score quite a bit lower than the full VSM in monolingual IR, are also present in the cross-lingual RI method tested here, together with other factors such as dictionary coverage.

There is some related work on the notion of bilingual vector spaces. Most related is the work by Dumais et al [8], who proposed a model for cross-lingual IR based on bilingual LSI. In contrast to the cross-lingual RI methods, their approach requires an aligned corpus of documents and their translations for training purposes. In a different area, Rapp proposed cross-lingual distributional similarity formalized as bilingual vector spaces to identify translation pairs in non-parallel text [20]. Peirsman & Padó used a bilingual vector space as an intermediary step in a model for learning selectional preferences [19]. Sahlgren & Karlgren describe an approach for automatic extraction of bilingual lexica using random indexing of parallel corpora [24].

4 Conclusion and Future Work

The first contribution of this paper is experimental results for random indexing in document retrieval by applying it to the monolingual (English) ad hoc IR track from CLEF 2005. It was found that indirect RI, which uses a sliding window approach during training, achieves slightly better mean average precision than direct RI, which is conceptually a compressed version of a standard VSM, suggesting a better ranking among the retrieved documents, whereas direct RI yields slightly better recall. A full VSM model as implemented in Lucene achieved better results than both of these. This is inconsistent with the claim that models such as LSI and RI improve retrieval because they model latent semantic relations among terms.

The second contribution is a new method for cross-lingual RI in which source and target language models share the same vector space, allowing direct comparison of the vector representations of source and target language texts without the need for any explicit translation. This is accomplished through a sharing of index vectors across languages during the random indexing procedure. It requires a translation dictionary and unrelated monolingual text corpora, but no aligned bilingual text. Of the three different variants proposed, indirect cross-lingual RI with translation vectors performed best when applied to the German-English bilingual ad hoc IR track from CLEF 2005. A straight-forward method of using a dictionary for translation of the queries and then Lucene for monolingual IR achieved better results than using our proposed methods.

Despite relatively low performance, the cross-lingual RI approach may still be attractive because of several advantages. First, it is very light-weight in terms of resources, as it only requires a translation dictionary. There is no need for bilingual data in the form of parallel documents or word-aligned text, which can be expensive to construct. Second, it inherits the computational simplicity from standard RI and is therefore scalable to huge document collections while retaining relateively small models. Third, additional target languages can be added without the need to retrain the existing models. Forth, queries and documents are in the same cross-lingual vector space, so no explicit translation step is required. In addition, the method may have potential uses in specialized domains utilizing specialized sublanguages where little or no aligned training data is available. One such example being the clinical domain, which contains specialized documents for which parallel or aligned text is difficult to produce and obtain. This may also include *cross-domain IR*, possibly incorporating domain knowledge into the cross-language/-domain dictionary to model domain-dependent relations among terms and documents.

There are still many unsolved questions related to application of RI in retrieval. For instance, no good explanation is yet given for why capturing latent semantic relations among terms seemingly does not improve document retrieval. One possible explanation is that the features which make two documents similar, or dissimilar, are not the same as those that determine similarity on a term level (e.g. synonymy). Another explanation could be that the way vectors are combined into documents, i.e. through TF*IDF weighted summation, is not optimal for capturing or preserving higher-order semantic relations among terms. More experimentation is needed to explore a wider range of model configurations on more benchmark data, both for monolingual IR using vectors of higher order semantic information and for CLIR with language pairs other than German–English. A direct comparison between LSI and RI for CLIR is desirable as well. There is also a need for a more thorough evaluation of using the presented *term translation vectors* in detecting semantically similar terms across languages.

Acknowledgements. We would like to thank Björn Gambäck for discussion and useful suggestions. This work was partly funded by the EviCare project (http://www.evicare.no) and by the European Community's Seventh Framework Programme (FP7/20072013) under grant agreement nr 248307 (PRESEMT).

References

1. Atreya, A., Elkan, C.: Latent semantic indexing (LSI) fails for TREC collections. SIGKDD Explorations 12(2), 5–10 (2010)
2. Basile, P., Caputo, A., Semeraro, G.: Semantic vectors: an information retrieval scenario. In: IIR, pp. 27–28 (2010)
3. Berry, M., Dumais, S., O'Brien, G.: Using linear algebra for intelligent information retrieval. SIAM Review 37(4), 573–595 (1995)
4. Braschler, M., Peters, C.: CLEF Methodology and Metrics. In: Peters, C., Braschler, M., Gonzalo, J., Kluck, M. (eds.) CLEF 2001. LNCS, vol. 2406, pp. 394–404. Springer, Heidelberg (2002)
5. Carrillo, M., Villatoro-Tello, E., López-López, A., Eliasmith, C., Montes-y-Gómez, M., Villaseñor-Pineda, L.: Representing context information for document retrieval. In: Andreasen, T., Yager, R.R., Bulskov, H., Christiansen, H., Larsen, H.L. (eds.) FQAS 2009. LNCS, vol. 5822, pp. 239–250. Springer, Heidelberg (2009)
6. Cohen, T., Widdows, D.: Empirical distributional semantics: Methods and biomedical applications. Journal of Biomedical Informatics 42(2), 390 (2009)
7. Deerwester, S., Dumais, S., Furnas, G., Landauer, T., Harshman, R.: Indexing by latent semantic analysis. Journal of the American Society for Information Science 41(6), 391–407 (1990)
8. Dumais, S., Letsche, T., Littman, M., Landauer, T.: Automatic cross-language retrieval using latent semantic indexing. In: AAAI Spring Symposium on Cross-Language Text and Speech Retrieval, pp. 15–21 (1997)
9. Hassel, M.: JavaSDM package (2004), `http://www.nada.kth.se/~xmartin/java/`
10. Jones, K.S.: A Statistical Interpretation of Term Specificity and its Application in Retrieval. Journal of Documentation 28(1), 11–21 (1972)
11. Kanerva, P.: Sparse distributed memory: A study of psychologically driven storage. MIT press (1988)
12. Kanerva, P., Kristoferson, J., Holst, A.: Random indexing of text samples for latent semantic analysis. In: Gleitman, L., Josh, A. (eds.) Proceedings of the 22nd Annual Conference of the Cognitive Science Society, p. 1036. Erlbaum, Mahwah (2000)
13. Karlgren, J., Sahlgren, M.: From Words to Understanding. In: Uesaka, Y., Kanerva, P., Asoh, H. (eds.) Foundations of Real-World Intelligence, pp. 294–308. CSLI Publications, Stanford (2001)

14. Kishida, K.: Technical issues of cross-language information retrieval: a review. Information Processing & Management 41(3), 433–455 (2005)
15. Lioma, C., Macdonald, C., He, B., Plachouras, V., Ounis, I.: Applying Light Natural Language Processing to Ad-Hoc Cross Language Information Retrieval. In: Peters, C., Gey, F.C., Gonzalo, J., Müller, H., Jones, G.J.F., Kluck, M., Magnini, B., de Rijke, M., Giampiccolo, D. (eds.) CLEF 2005. LNCS, vol. 4022, pp. 170–178. Springer, Heidelberg (2006)
16. Apache Lucene open source package, `http://lucene.apache.org/`
17. Manning, C., Raghavan, P., Schütze, H.: Introduction to information retrieval, vol. 1. Cambridge University Press, Cambridge (2008)
18. McNamee, P.: Exploring New Languages with HAIRCUT at CLEF 2005. In: Peters, C., Gey, F.C., Gonzalo, J., Müller, H., Jones, G.J.F., Kluck, M., Magnini, B., de Rijke, M., Giampiccolo, D. (eds.) CLEF 2005. LNCS, vol. 4022, pp. 155–164. Springer, Heidelberg (2006)
19. Peirsman, Y., Padó, S.: Cross-lingual Induction of Selectional Preferences with Bilingual Vector Spaces. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 921–929. Association for Computational Linguistics, Los Angeles, Los Angeles (2010)
20. Rapp, R.: Identifying word translations in non-parallel texts. In: Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics, pp. 320–322. Association for Computational Linguistics (1995)
21. Robertson, S.E., Walker, S., Jones, S., Hancock-Beaulieu, M., Gatford., M.: Okapi at trec-3. In: Proceedings of the Third Text REtrieval Conference (TREC 1994), Gaithersburg, USA (1994)
22. Ruiz, M., Eliasmith, C., López, A.: Exploring the Use of Random Indexing for Retrieving Information. Tech. Rep. CCC-08-006, INAOE (2008)
23. Sahlgren, M.: An introduction to random indexing. In: Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering, TKE, vol. 5 (2005)
24. Sahlgren, M., Karlgren, J.: Automatic bilingual lexicon acquisition using random indexing of parallel corpora. Natural Language Engineering 11(03), 327–341 (2005)
25. Sahlgren, M., Holst, A., Kanerva, P.: Permutations as a Means to Encode Order in Word Space. In: Proceedings of the 30th Conference of the Cognitive Science Society (2008)
26. Salton, G., Wong, A., Yang, C.: A vector space model for automatic indexing. Communications of the ACM 18(11), 613–620 (1975)
27. Schmid, H.: Probabilistic Part-of-Speech Tagging Using Decision Trees. In: Proceedings of International Conference on New Methods in Language Processing, Manchester, UK, vol. 12, pp. 44–49 (1994)
28. Sellberg, L., Jönsson, A.: Using random indexing to improve singular value decomposition for latent semantic analysis. In: Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC 2008) (May 2008)
29. Turney, P., Pantel, P.: From Frequency to Meaning: Vector Space Models of Semantics. Journal of Artificial Intelligence Research 37, 141–188 (2010)
30. Vasuki, V., Cohen, T.: Reflective random indexing for semi-automatic indexing of the biomedical literature. Journal of Biomedical Informatics 43(5), 694–700 (2010)

Rational Kernels for Arabic Text Classification*

Attia Nehar[1], Djelloul Ziadi[2], and Hadda Cherroun[1]

[1] Laboratoire d'Informatique et Mathématiques
Université Amar Télidji Laghouat, Algérie
{a.nehar,h.cherroun}@mail.lagh-univ.dz

[2] Laboratoire LITIS - EA 4108, Normandie Université, Rouen, France
Djelloul.Ziadi@univ-rouen.fr

Abstract. Many stemming techniques are used in the context of Arabic Text Classification. In this paper, we show the effect of stemming on classification systems. We introduce a new stemming technique -approximate stemming- based on the use of Arabic patterns. These patterns are modeled using transducers and stemming is done without depending on any dictionary. Using transducers for stemming words, documents are transformed into finite state transducers. This allow us to use rational kernels as a framework for Arabic Text Classification. Experiments show that, when compared with other approaches, our approach is more effective specially in term of Accuracy, Recall and F1.

Keywords: Arabic, Automata, Kernels, N-gram, Text Classification.

1 Introduction

Text classification (TC) is the task of automatically sorting a set of documents into one or more categories from a predefined set [1]. Text classification techniques are used in many domains, including mail spam filtering, article indexing, Web searching, automated population of hierarchical catalogues of Web resources, even automated essay grading task.

Due to the complexity of the arabic language, Arabic Text Classification (ATC) starts receiving great attention. Many algorithms have been developed to improve performance of Arabic TC systems [2–7]. In general, we can divide an Arabic text classification system into three steps:

1. **Preprocessing Step:** where punctuation marks, diacritics, stop words and non letters are removed.
2. **Features Extraction:** a set of features is extracted from the text, which will represent the text in the next step. For instance, Khreisat [5] used the N-gram technique to extract features from documents. Syiam *et al.* [7], used stemming to extract features.
3. **Learning Step:** many supervised algorithms were used to learn systems how to classify arabic text documents: Support Vector Machines [4, 6], K-Nearest

* This work is supported by the MESRS - Algeria under Project 8/U03/7015.

Neighbors [7] and many others. Most algorithms rely on distance measures over extracted features to decide how much two documents are similar.

In the second step, a feature vector is constructed. Several stemming approaches are developed [8]. Khoja and Garside [9] developed a dictionary based stemmer. It gives good performances, but the dictionary needs to be maintained. The stemmer of Al-Serhan *et al.* [10] finds the three-letter roots for arabic words without depending on any roots dictionary or pattern files.

Many arabic words have the same stem but not the same meaning. Reducing two semantically different words to the same root can induce classification errors. To prevent this, light stemming is used in TC algorithms [11]. Its main idea is that a lot of words generated from the same root have different meanings. The basis of this light-stemming algorithm consists of several rounds over the text, that attempt to locate and remove the most frequent prefixes and suffixes from the words. This leads to a lot of features due to the light stemming strategy.

In the third step, many distance measures could be used to calculate distance (or dissimilarity) between documents using these feature vectors. The quality of the classification system is related to the used distance measure.

In this paper, we study the effect of stemming on ATC. Let's illustrate this by an example. Given two simple documents d_1 =" يتعلم و يتربي الطفل في المدرسة" (Child learns and brought up in the school), and d_2 = "تقدم المدارس لاطفالنا التعليم و التربية" (Schools provide education for our children). we compute Euclidean distance between them using 3-grams, with and without stemming:

Distance	**3-grams**
Without stemming	0.18
With stemming	0.25

It is clear that distance between d_1 and d_2 is affected by stemming, specially using 3-grams.

In this work, we introduce a new stemming technique which do not rely on any dictionary. It is based on the use of transducers. This stemming technique transforms documents into finite state transducers. Then, rational kernels [12] are used as a framework to do ATC.

This paper is organized as follows. Section 2 presents, in more details, the stemming techniques. In Section 3, we recall some notions on weighted transducers and rational kernels. We present our new stemming approach, called "approximate-stemming", then we explain how to use rational kernels as a framework for Arabic TC. Experiments and results are reported and interpreted in Section 4.

2 Stemming Techniques

In the context of ATC, stemming is applied to reduce dimentiality of the feature vectors. Brute stemming (commonly called stemming) transforms each arabic word in the document, into its root. However, light stemming, reduces word into its light stem by removing prefixes and suffixes.

2.1 Brute Stemming

There are many brute stemming techniques used in the context of ATC. They can be classified into two types: *(i) Stemming using a dictionary*, where dictionary of arabic word stems is needed. *(ii) Stemming without dictionary*, where stems are extracted without depending on any root or pattern files.

Khoja's stemmer [9] removes the longest suffix and the longest prefix. It then matches the remaining word with verb and noun patterns, to extract the root by means of a dictionary. The stemmer makes use of many linguistic data files such as a list of all diacritic characters, punctuation characters, definite articles and stop words. This stemmer gives good performance but relies on dictionary which needs to be updated. The second technique, due to Al-Serhan *et al.* [10], finds the three-letter roots for arabic words without depending on any root or pattern files. They extract word roots by assigning weights and ranks to the letters that constitute a word. This algorithm, like any other brute stemming algorithm, gives the same stem for two semantically different words. This could decrease performance of the classification system.

2.2 Light Stemming

In Arabic language, some word variants do not have similar meanings (like the two words: مكتبة which means *library* and كاتب which means writer). However, these word variants give the same root if a brute stemming is used. Thus, brute stemming affects the meanings of words. Light stemming [11] aims to enhance the Text classification performance while retaining the words meanings. The basis of this light-stemming algorithms consists of several rounds over the text, that attempt to locate and remove the most frequent prefixes and suffixes from the word. However, it leads to a lot of features.

3 Framework for Arabic Text Classification

Before describing our framework, let's give in the following subsection, some preliminaries on Weighted Transducers and Rational Kernels.

3.1 Weighted Transducers and Rational Kernels

Transducers are finite automata in which each transition is augmented with an output label in addition to the familiar input label. Output labels are concatenated along a path to form an output sequence as with input labels. *Weighted transducers* are finite-state transducers in which each transition carries some weight in addition to the input and output labels. The weight of a pair of input and output strings (x, y) is obtained by summing the weights of the paths labeled with (x, y). The following definition gives a formal definition of weighted transducers [13, 14].

Definition 1. *A weighted finite-state transducer T over a semiring $(\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$ is an 8-tuple $T = (\Sigma, \Delta, Q, I, F, E, \lambda, \rho)$ where Σ is a finite input alphabet, Δ is a finite output alphabet, Q is a finite set of states, $I \subseteq Q$ the set of initial states, $F \subseteq Q$ the set of final states, $E \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times (\Delta \cup \{\epsilon\}) \times \mathbb{K} \times Q$ a finite set of transitions, $\lambda : I \rightarrow \mathbb{K}$ the initial weight function, and $\rho : F \rightarrow \mathbb{K}$ the final weight function*

For a path π in a transducer, $p[\pi]$ denotes the origin state of that path, $n[\pi]$ its destination state and $w[\pi]$ gives the sum of the weights of its arcs. The set of paths from the initial states I to the final states F labeled with input string x and output string y is denoted by $P(I, x, y, F)$. A transducer T is *regulated* if the output weight associated by T to any pair of input-output strings (x, y) given by:

$$[\![T]\!](x, y) = \bigoplus_{\pi \in P(I,x,y,F)} \lambda(p[\pi]) \otimes w[\pi] \otimes \rho[n[\pi]] \tag{1}$$

is well-defined in $\mathbb{K}$. $[\![T]\!](x, y) = \bar{0}$ if $P(I, x, y, F) = \emptyset$. Figure 1 shows an example of a simple transducer, with an input string x : فاعل and an output string y : فعل . The only possible path in this transducer is the singular set: $P(\{0\}, x, y, \{4\})$.

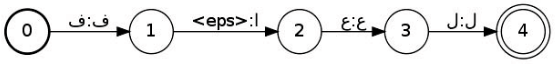


Fig. 1. Example of a transducer

Regulated weighted transducers are closed under the following operations called rational operations:

– the *sum* (or *union*) of two weighted transducers T_1 and T_2 is defined by:

$$\forall (x, y) \in \Sigma^* \times \Sigma^*, [\![T_1 \oplus T_2]\!](x, y) = [\![T_1]\!](x, y) \oplus [\![T_2]\!](x, y) \tag{2}$$

– the *product* (or *concatenation*) of two weighted transducers T_1 and T_2 is defined by:

$$\forall (x, y) \in \Sigma^* \times \Sigma^*, [\![T_1 \otimes T_2]\!](x, y) = \bigoplus_{\substack{x = x_1 x_2, \\ y = y_1 y_2}} [\![T_1]\!](x_1, y_1) \otimes [\![T_2]\!](x_2, y_2) \tag{3}$$

– The composition of two weighted transducers T_1 and T_2 with matching input and output alphabets Σ, is a weighted transducer denoted by $T_1 \circ T_2$ when the sum:

$$[\![T_1 \circ T_2]\!](x, y) = \bigoplus_{z \in \Sigma^*} [\![T_1]\!](x, z) \otimes [\![T_2]\!](z, y) \tag{4}$$

is well-defined in $\mathbb{K}$ for all $x, y \in \Sigma^*$

Rational Kernels are a general family of kernels, based on weighted transducers, that extend kernel methods to the analysis of variable-length sequences or more generally weighted automata. Let X and Y be non-empty sets. A function $K : X \times Y \to \mathbb{R}$ is said to be a kernel over $X \times Y$. Corinna *et al.* [12] give a formal definition for rational kernels:

Definition 2. *A kernel K over $\Sigma^* \times \Delta^*$ is said to be rational if there exist a weighted transducer $T = (\Sigma, \Delta, Q, I, F, E, \lambda, \rho)$ over the semiring $\mathbb{K}$ and a function $\varphi : \mathbb{K} \to \mathbb{R}$ such that for all $x \in \Sigma^*$ and $y \in \Delta^*$:*

$$K(x, y) = \varphi([\![T]\!](x, y)) \tag{5}$$

K is then said to be defined by the pair (φ, T).

3.2 Stemming by Transducers

Arabic language differs from other languages syntactically, morphologically and semantically. One of the main characteristic features is that most words are built up from roots by following certain fixed patterns[1] and adding prefixes and suffixes. For instance, the arabic word المدرسة (school) is built from the three-letters root or stem درس (learn) and using the measure مفعلة (see Table 1), then prefix ال and suffix ة (which is used to denote female gender) are added. Notice here that the letter ف denotes the first letter of the three-letters root, ع denotes the second letter and ل denotes the third one.

Table 1. Measures for the three-letters root د ر س and the built words

Measures	مفعلة	فاعل	الفاعلون	الفعالة	يفعل	يتفاعل
Words	مدرسة	دارس	الدارسون	الدراسة	يدرس	يتدارس

[1] Also called measures or binyan.

We will use measures to construct a transducer which do stemming. Figure 1 shows the example of the measure فاعل. This transducer ($T_{measure1}$) can be used to extract the three-letters root of any arabic word matching with this measure. This is achieved by composition operation (4).

We consider T_{word}, the transducer which map any string to it self, i.e, the only possible path is the singleton set $P(\{0\}, word, word, \{i\})$ (Figure 2 shows transducer associated to the arabic word مدرسة).

The composition of two transducers is also a transducer.

$$(T_{word} \circ T_{measure1})(word, y) = \sum_{z \in \Sigma^*} T_{word}(word, z) \cdot T_{measure1}(z, y)$$

Since the only possible string matching z is $z = word$, we conclude that:

$$(T_{word} \circ T_{measure1})(word, y) = T_{word}(word, word) \cdot T_{measure1}(word, y)$$

As we have $T_{word}(word, word) = 1$, so:

$$(T_{word} \circ T_{measure1})(word, y) = T_{measure1}(word, y)$$

If *word* matches with the measure the output projection will extract the root (or stem) y associated to *word.*

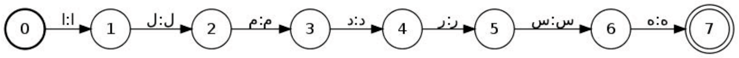


Fig. 2. Transducer corresponding to the word المدرسة (school)

In Arabic language, there are 4 verb prefixes (ن ا ي ت), 12 noun prefixes (ا، ال، ب، ت، س، ف، لل، ل، ي، و، ن، م) and more than 20 suffixes. As Modern Standard Arabic don't use diacritics, we don't consider them in our approach. his reduces patterns into 200. Indeed, the patterns (فَعَلَ، فَعُلَ، فُعَلٌ، فَعِلٌ) will result in only one pattern (فعل) after removing diacritics.

We adopt the following process, to construct the stemming transducer, which enable us to include all measures:

1. Building the transducer of all noun prefixes (resp. verb prefixes);
2. Building the transducer of all noun patterns (resp. verb patterns);
3. Building the transducer of all noun suffixes (resp. verb suffixes);
4. Concatenate noun transducers (resp. verb transducers) obtained in 1, 2 and 3.
5. Sum the two transducers obtained in step 4.

The first and third steps are very simple. We construct a transducer for each prefix (resp. suffix) then we do the union of these transducers. The resulting transducer represents the prefixes (resp. suffixes) transducer (see Figure 3). In the second step, we build all possible noun pattern transducers. Then, the sum of these transducers represents the transducer of all noun patterns. We do the same to build the transducer of all verb patterns. In the forth step, transducers obtained in steps 1, 2 and 3 are concatenated. The final transducer is obtained by doing the sum (union) of transducers built in step 4.

Tables 2 shows some examples of noun patterns. The resulting transducer could not be represented graphically because of large number of states (about 400 states). This transducer can stem any well-formed arabic word, i.e, a word which matches with some arabic measure. In addition, it can give us a semantic information about the stemmed word. This information can be used to improve the quality of classification system.

Table 2. Examples of noun patterns

3-letters	4-letters	5-letters	6-letters	7-letters
فعل	فاعل	مفاعل	متفاعل	استفعال

Transducers are created and manipulated using the OpenFst library [15], which is an open source library for constructing, combining, optimizing, and searching weighted finite-state transducers.

3.3 Rational Kernels for Arabic Text Classification

Like any text classification system, Our system is divided into three stages:

1. preprocessing step.
2. feature extraction: the previous transducer is applied on each word of the document resulting from step 1. Then, the transducer consisting of the concatenation of these words stems transducers will represent the document in the next step.
3. learning task: Rational kernels will be used to measure distance between documents [12, 14].

Considering a set of documents, each document is consisting of a sequence of words: $w_1 w_2 \dots w_n$. Applying our stemming transducer on each word of a document will transform this document into finite state transducer. These transducers will be packaged into an archive file (far) to be treated by the learning algorithm. OpenKernel, which is a library for creating, combining and using kernels for machine learning applications, will be used to accelerate experiments.

The next batch reports the main commands of OpenFst and OpenKernel libraries used to implement our classification system. To stem words in the document, we iterate on these words using the OpenFst command [15] *fstcompose* (line 1), where *word.fst* is a linear finite state transducer with identical input

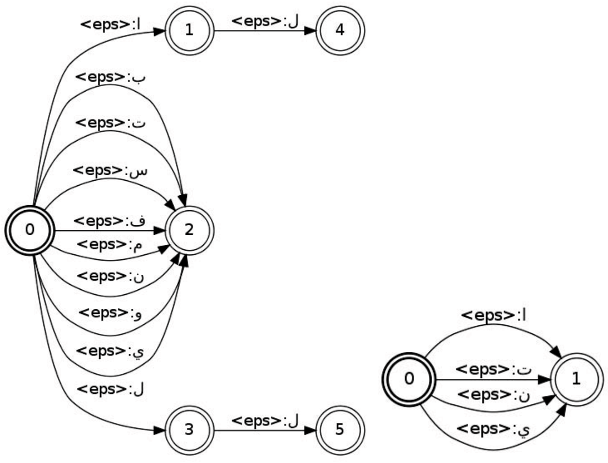


Fig. 3. Transducer of noun prefixes (left) and verb prefixes (right)

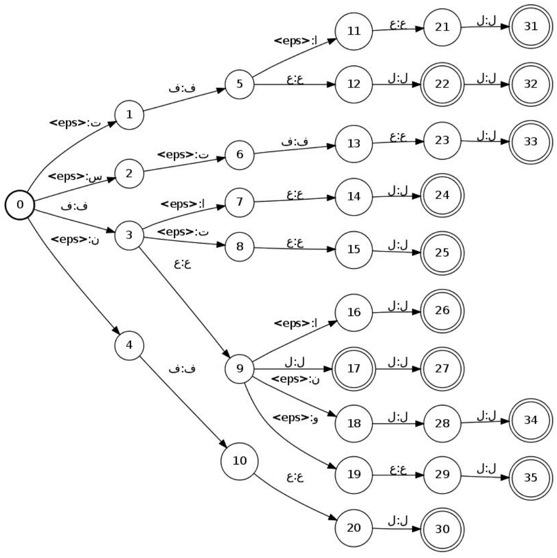


Fig. 4. Transducer of verb patterns

and output labels, which represents a word, and *model.fst* is our stemming transducer [16]. The resulting transducer *result.fst* represents the set of possible stems. Resulting transducers are right concatenated to a finite state transducer, representing the entire document, using the openfst command *fstconcate* (line 2). The set of finite state documents (FSTs) is then packaged in a FSTs archive (far) using the openkernel command *farcreate* (line 3), where *data.list* contains the list of all FSTs documents, one file per line, and *data.far* is the FST archive (Far).

Various types of kernels could be created using openkernel library. 3-gram kernels could be created using the command *klngram* (line 3), where the first argument *–order* specifies the size of the n-grams, and the second argument *–sigma* specifies the size of the alphabet (arabic alphabet size is 29). The first parameter is the FST archive (*data.far*) and the second parameter (*3-gram.kar*) is the resulting kernel archive.

```
1 fstcompose word.fst model.fst result.fst
2 fstconcate doc.fst result.fst doc.fst
3 farcreate data.list data.far
4 klngram −order=3 −sigma=29 data.far 3−gram.kar
5 svm−train −k openkernel −K 2−gram.kar cul.train cul.train.2−gram.model
6 svm−predict cul.test cul.train.2−gram.model cul.test.2−gram.pred
```

Openkernel library includes a plugin for the LibSVM implementation [17]. This enables us to do training, predicting and scoring on our dataset. Training command creates a model on the training set (line 5), where the first argument -k specifies the kernel format, the second one (-K) specifies the n-gram size. The first parameter specifies a correctly classified subset of the training set, the second parameter is the resulting model. In this command, *cul.train* contains a labeled sub set of training documents belonging to Cultural class. Having a model, we can use it to classify documents of the testing dataset with the command *svm-predict* (line 6), where the first parameter specifies a correctly classified subset of the testing set, the second parameter is the resulting model of the previous command. The last parameter contains the result of prediction using the model.

4 Experimental Results and Discussion

We perform experiments on the Saudi Press Agency (SPA) dataset [2] for training and testing the ATC system. This dataset contains 1526 text documents belonging to one of the six categories (culture, economic, social, general, politics and sport) as detailed on Table 3. As mentioned before, stop words, non arabic letters, symbols and digits were removed. We have used 80% of documents for training the classifier and 20% for testing. Learning is done using LibSVM implementation [17], included in Openkernel, with three different n-gram kernels ($n = 2, 3, 4$). Since we want to show the effect of stemming, we report results of the three classifier versions; without stemming, with our approximate-stemming and with Al-Serhan's stemmer, in term of accuracy, precision, recall and F1. In Tables 4, 6 and 8 we report results in term of accuracy and precision for the three classifiers with the tree kernels (bigrams, 3-grams and 4-grams). Tables 5, 7 and 9 give results in terms of recall and F1 for the same classifiers.

For the three classifiers, best results were reached with 3-grams kernel for most measures (accuracy, recall and F1). This can be explained by the fact that over 80% of arabic words can be mapped into 3-letter root patterns. However, for precision, stemming do not enhance performances. This can be explained by the fact that we consider all possible stems for each word. Thus, it is more beneficial to consider only the most probable stem by a statistical langage study [18].

Table 3. SPA corpus details

Categories	Training texts	Testings texts	Total
Culture	201	57	258
Economics	200	50	250
Social	203	55	258
Politics	200	50	250
General	205	50	255
Sports	205	50	255
	1214	312	1526

Table 4. Accuracy and Precision of SVM Classification using Bigram Kernel

Class	**Accuracy (%)**			**Precision (%)**		
	Without stemming	With stemming	With Serhan stemmer	Without stemming	With stemming	With Serhan stemmer
Culture	85,49	87,38	86,75	92,30	75,75	75,86
Economics	87,38	88,01	87,69	91,66	77,27	73,91
Social	82,65	84,23	82,64	-	100	50
Politics	85,17	85,49	87,06	63,63	60	73,68
General	85,17	88,01	84,54	90	84	68,75
Sports	95,90	95,90	94,32	97,36	95,12	88,09
Average	86,91	**88,17**	87,17	**86,99**	82,02	71,71

Table 5. Recall and F1 of SVM Classification results using Bigram Kernel

Class	**Recall**			**F1**		
	Without stemming	With stemming	With Serhan stemmer	Without stemming	With stemming	With Serhan stemmer
Culture	0,21	0,44	0,39	0,34	0,56	0,51
Economics	0,22	0,34	0,34	0,35	0,47	0,47
Social	-	0,09	0,02	-	0,17	0,03
Politics	0,14	0,24	0,28	0,23	0,34	0,41
General	0,16	0,38	0,20	0,28	0,52	0,31
Sports	0,74	0,78	0,74	0,84	0,86	0,80
Average	0,25	**0,38**	0,33	0,41	**0,49**	0,42

Table 6. Accuracy and Precision of SVM Classification using 3-gram Kernel

Class	**Accuracy (%)**			**Precision (%)**		
	Without stemming	With stemming	With Serhan stemmer	Without stemming	With stemming	With Serhan stemmer
Culture	89,59	90,22	87,06	87,50	88,23	80,76
Economics	91,48	92,43	92,11	84,84	88,23	87,87
Social	83,91	83,60	83,28	75	61,53	75
Politics	89,27	88,33	89,59	90	80,95	84
General	89,27	87,38	89,59	92	77,77	95,83
Sports	97,48	97,79	94,95	100	100	90,47
Average	90,17	89,96	89,43	**88,22**	82,79	85,65

Considering the effect of stemming on the performance of classifiers, results show that both our approximate stemmer and Al-serhan's one improve performances.

At last, compared with Al-Serhan's stemmer, our stemmer gives better results in most cases.

Table 7. Recall and F1 of SVM Classification using 3-gram Kernel

Class	Recall			F1		
	Without stemming	With stemming	With Serhan stemmer	Without stemming	With stemming	With Serhan stemmer
Culture	0,49	0,53	0,37	0,63	0,66	0,51
Economics	0,56	0,60	0,58	0,67	0,71	0,70
Social	0,11	0,15	0,05	0,19	0,23	0,10
Politics	0,36	0,34	0,42	0,51	0,48	0,56
General	0,42	0,38	0,41	0,57	0,51	0,58
Sports	0,84	0,86	0,76	0,91	0,92	0,83
Average	0,46	**0,48**	0,43	0,58	**0,59**	0,55

Table 8. Accuracy and Precision of SVM Classification using 4-gram Kernel

Class	Accuracy (%)			Precision (%)		
	Without stemming	With stemming	With Serhan stemmer	Without stemming	With stemming	With Serhan stemmer
Culture	89,91	89,59	84,54	87,87	92,85	90
Economics	92,74	91,80	87,38	84,61	92,85	85,71
Social	84,54	83,60	82,64	75	71,42	50
Politics	87,07	87,38	86,12	71,42	81,25	71,42
General	88,96	89,27	86,75	85,71	95,65	93,33
Sports	97,79	96,85	91,48	100	100	87,09
Average	90,17	89,75	86,49	84,10	**89**	79,59

Table 9. Recall and F1 of SVM Classification using 4-gram Kernel

Class	Recall			F1		
	Without stemming	With stemming	With Serhan stemmer	Without stemming	With stemming	With Serhan stemmer
Culture	0,51	0,46	0,16	0,64	0,61	0,27
Economics	0,66	0,52	0,24	0,74	0,67	0,37
Social	0,16	0,09	0,02	0,27	0,16	0,035
Politics	0,30	0,26	0,20	0,42	0,39	0,31
General	0,44	0,40	0,25	0,58	0,56	0,40
Sports	0,86	0,80	0,54	0,92	0,89	0,66
Average	**0,49**	0,23	0,42	**0,59**	0,55	0,34

5 Conclusion and Future Directions

This paper presents a new framework for Arabic Text classification. It is based on the use of transducers for stemming arabic words, and rational kernels for measuring distance between documents. First, our stemming transducer is built by means of arabic patterns. Second, rational kernels are also used to measure distances between documents. Experiments and analysis of this framework in the context of Arabic Text Classification show that stemming improves the quality of classifiers in term of accuracy, recall and F1. But it decreases the precision. 3-grams classifier reached the best results. Like that of Aljlayl *et al.* [11], our approach of stemming do not rely on dictionary, and gives better results than statistically based stemmer of Al-Serhan *et al.* [10].

In future work, approximate-stemmer will be enhanced by a statistical study of Arabic language. Other kernels, like word-grams and gappy grams, will be investigated.

References

1. Sebastiani, F., Ricerche, C.N.D.: Machine Learning in Automated Text Categorization. ACM Computing Surveys 34, 1–47 (2002)
2. Althubaity, A., Almuhareb, A., Alharbi, S., Al-Rajeh, A., Khorsheed, M.: KACST Arabic Text Classification Project: Overview and Preliminary Results. In: Proceedings of The 9th IBIMA Conference on Information Management in Modern Organizations (January 2008)
3. Duwairi, R.M.: Arabic Text Categorization. Int. Arab J. Inf. Technol. 4(2), 125–132 (2007)
4. Gharib, T., Habib, M., Fayed, Z.: Arabic Text Classification Using Support Vector Machines. International Journal of Computers and Their Applications 16(4), 192–199 (2009)
5. Khreisat, L.: A machine learning approach for Arabic text classification using N-gram frequency statistics. Journal of Informatrics 3(1), 72–77 (2009)
6. Mesleh, A.: Support Vector Machines based Arabic Language Text Classification System: Feature Selection Comparative Study. In: Sobh, T. (ed.) Advances in Computer and Information Sciences and Engineering, pp. 11–16. Springer, Netherlands (2008)
7. Syiam, M., Fayed, Z., Habib, M.: An Intelligent System For Arabic Text Categorization. International Journal of Intelligent Computing and Information Sciences 6(1), 1–19 (2006)
8. Al-Nashashibi, M., Neagu, D., Yaghi, A.: Stemming techniques for Arabic words: A comparative study. In: Computer Technology and Development (ICCTD), pp. 270–276 (November 2010)
9. Khoja, S., Garside, R.: Stemming arabic text (1999)
10. Al-Serhan, H., Shalabi, R.A., Kannan, G.: New Approach For Extracting Arabic Roots. In: Proceedings of The 2003 Arab Conf. on Infor. Technology, Alexandria, Egypt, pp. 42–59 (December 2003)
11. Aljlayl, M., Frieder, O.: On Arabic Search: Improving the Retrieval Effectiveness Via Light Stemming Approach. In: ACM Eleventh Conference on Infor. and Knowledge Management, pp. 340–347 (2002)
12. Cortes, C., Haffner, P., Mohri, M.: Rational Kernels: Theory and Algorithms. J. Mach. Learn. Res. 5, 1035–1062 (2004)
13. Berstel, J.: Transductions and Context-Free Languages. Teubner Studienbücher, Stuttgart (1979)
14. Cortes, C., Kontorovich, L., Mohri, M.: Learning languages with rational kernels. In: Bshouty, N.H., Gentile, C. (eds.) COLT. LNCS (LNAI), vol. 4539, pp. 349–364. Springer, Heidelberg (2007)
15. Allauzen, C., Riley, M.D., Schalkwyk, J., Skut, W., Mohri, M.: OpenFst: A General and Efficient Weighted Finite-State Transducer Library. In: Holub, J., Žďárek, J. (eds.) CIAA 2007. LNCS, vol. 4783, pp. 11–23. Springer, Heidelberg (2007)
16. Nehar, A., Ziadi, D., Cherroun, H., Guellouma, Y.: An Efficient Stemming for Arabic Text Classification. In: Innovations in Information Technology (IIT), pp. 328–332. IEEE (March 2012)
17. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2, 27:1–27:27 (2011)
18. Lakhdari, A., Cherroun, H.: Effective Unsupervised Morphological Analysis and Modeling: Statistical Study for Arabic Language. In: Book of Abstracts of the 23rd Meeting of Computational Linguistics in the Netherlands: CLIN, p. 85 (January 2013)

Experiments with Semantic Similarity Measures Based on LDA and LSA

Nobal Niraula, Rajendra Banjade, Dan Ştefănescu, and Vasile Rus

Department of Computer Science
The University of Memphis, USA
{rbanjade,nbnraula,dstfnscu,vrus}@memphis.edu

Abstract. We present in this paper experiments with several semantic similarity measures based on the unsupervised method Latent Dirichlet Allocation. For comparison purposes, we also report experimental results using an algebraic method, Latent Semantic Analysis. The proposed semantic similarity methods were evaluated using one dataset that includes student answers from conversational intelligent tutoring systems and a standard paraphrase dataset, the Microsoft Research Paraphrase corpus. Results indicate that the method based on word representations as topic vectors outperforms methods based on distributions over topics and words. The proposed evaluation methods can also be regarded as an extrinsic method for evaluating topic coherence or selecting the number of topics in LDA models, i.e. a task-based evaluation of topic coherence and selection of number of topics in LDA.

Keywords: semantic similarity, statistical methods, Latent Dirichlet Allocation.

1 Introduction

We address in this paper the important task of finding how semantically similar two texts are. We employ a novel set of semantic similarity methods that rely on the probabilistic method Latent Dirichlet Allocation (LDA; Blei, Ng, & Jordnan, 2003).

Semantic similarity is a widely used approach to the core problem of language understanding. It is an useful alternative to the true understanding approach which is intractable as it requires world knowledge. For instance, in dialogue-based Intelligent Tutoring Systems (ITS) it is important to understand students' natural language responses. One frequently used approach to address this issue is to compute how similar student responses are to benchmark, expert-articulated responses (Graesser, Olney, Haynes, Chipman, 2005; Rus & Graesser, 2006). That is, the student response assessment task is being modeled as a text-to-text similarity problem.

Below, we show an example of a real student response from an ITS and the corresponding benchmark answer authored by an expert.

Student Response: *An object that has a zero force acting on it will have zero acceleration.*

Expert Answer: *If an object moves with a constant velocity, the net force on the object is zero.*

The student response above is deemed correct as it is semantically similar to the expert answer. A student response is deemed incorrect if it is not similar to the expert response. More nuanced categorizations are possible, e.g. a student response can be partially correct.

In this paper, we model the problem of semantic similarity as a binary decision problem in which a student response is deemed either correct or incorrect. We limit ourselves to such binary judgments because the primary scope of this work is to assess the novel semantic similarity methods based on the unsupervised method Latent Dirichlet Allocation (LDA; Blei, Ng, & Jordnan, 2003). We plan to address more nuanced judgments of semantic similarity in the future. Also, the datasets that we used to evaluate the proposed methods only provide binary judgments.

It should be noted that this type of binary modeling has been extensively used in previously proposed semantic similarity tasks such as the Recognizing Textual Entailment task (Dagan, Glickman, and Magnini, 2004), the paraphrase identification task (Dolan, Quirk, & Brockett, 2004), or the student input assessment task (Rus & Graesser, 2006; McCarthy & McNamara, 2008).

The task of semantic similarity can be formulated at different levels of granularity ranging from word-to-word similarity to sentence-to-sentence similarity to document-to-document similarity or a combination of these such as word-to-sentence or sentence-to-document similarity. We propose in this paper novel solutions to the task of semantic similarity both at word and sentence level with an emphasis on sentence-level similarity. In particular, we rely on one probabilistic method, LDA (Blei, Ng, & Jordan, 2003), that automatically discovers a set of underlying topics, represented as distributions over words, in texts. That is, texts are regarded as distribution over topics. Words can be represented as a vector of contributions to topics in an LDA model.

The semantic similarity measures of larger texts, e.g. sentences, can be defined based on either individual word representations, e.g. by extending word-to-word similarity measures to sentence-to-sentence similarity (as in Lintean et al., 2010), or based on the representations of texts as distributions over topics (topics are distributions over words in the vocabulary). We propose here solutions based on both of these approaches. The LDA-based word-to-word semantic similarity measures are used in conjunction with greedy and optimal matching methods in order to measure similarity between larger texts such as sentences. The solutions based on the second approach, called text-to-text measures, are used directly to compute the similarity of two sentences.

For comparison purposes, we also report experimental results using an algebraic method, Latent Semantic Analysis (LSA, Landauer et al., 2007), that automatically derives meaning representations in the form of latent concepts. Like LDA, LSA is fully automated. Words are represented as vectors in an LSA-derived semantic space. The dimensions of this space are latent concepts. Similarity of individual words and texts are computed based on vector algebra. LDA has one conceptual advantage over LSA: LDA represents multiple meanings of a word explicitly while LSA does not.

We have experimented with a dataset compiled from dialogue-based intelligent tutoring systems as well as with the Microsoft Research Paraphrase corpus (Dolan, Quirk, & Brockett, 2004).

The rest of the paper is organized as in the followings. The next section provides an overview of related work. Then, we describe LDA and the semantic similarity measures based on LDA. The Experiments and Results section describes our experimental setup and the results obtained. We conclude the paper with Discussion and Conclusions.

2 Previous Work

The task of semantic similarity between two short texts, namely two sentences, has been addressed using various solutions that range from simple word overlap to greedy methods that rely on word-to-word similarity measures (Fernando & Stevenson, 2008;) to algebraic methods (Lintean, Moldovan, Rus, & McNamara, 2010) to machine learning based solutions (Kozareva & Montoyo, 2006).

The most relevant work to ours is by Lintean et al. (2010) who looked at the role of LSA (Landauer et al., 2007) in solving the paraphrase identification task. As already mentioned, LSA is a vectorial representation in which a word is represented as a vector in a low dimensionality space (300-500 dimensions or latent concepts; we use 300 dimensions in our experiments reported here). Computing the similarity between two words is equivalent to computing the cosine, i.e. the normalized dot product, between the corresponding LSA vectors.

Lintean et al. (2010) used LSA as a way to compute semantic similarity in two different ways. First, they used LSA to compute a word-to-word similarity measure which they combined with a greedy-matching method to obtain a sentence level similarity score. For instance, each word in one sentence was greedily paired with one word in the other sentence. An average of these word-to-word similarities was then assigned as the semantic similarity score of the two sentences. Second, LSA was used to directly compute the similarity of two sentences by applying the cosine (normalized dot product) of the LSA vectors of the sentences. The LSA vector of a sentence was computed by adding all the individual word vectors. We present results with these methods and, additionally, with a method based on optimal matching that only uses word-to-word LSA similarity.

LDA itself was occasionally used for computing the semantic similarity of texts. The closest use of LDA for a semantic similarity task was by Celikyilmaz, Hakkani-Tur, & Tur (2010) for ranking candidate answers to a question in Question Answering (QA). Given a question, they ranked candidate answers based on how similar these answers were to the target question. That is, for each question-answer pair they generated an LDA model which then they used to compute a degree of similarity (DES) that consists of the product of two measures: sim1 and sim2. Sim1 captures the word-level similarities of the topics present in an answer and the question. Sim2 measures the similarities between the topic distributions in an answer and the question. The LDA model was generated based solely on each question and candidate answers. As opposed to our task, in which we compute the similarity between sentences, the candidate answers in Celikyilmaz, Hakkani-Tur, & Tur (2010) are longer, consisting of more than one sentence. This particular difference is important when it comes to computing the semantic similarity based on LDA as the shorter the texts the sparser the distributions, in particular the distribution over topics, based on which the similarity is computed.

Similar to Celikyilmaz, Hakkani-Tur, & Tur (2010), we define several semantic similarity measures based on the topic and word distributions in LDA. We do use Information Radius as Celikyilmaz, Hakkani-Tur, & Tur (2010) and, in addition, propose similarity measures based on Hellinger and Manhattan distances.

Another use of LDA for computing similarity between texts, namely blogs, relied on a very simple measure of computing the dot product of topic vectors as opposed to a similarity based on distributions (Chen et al., 2012). Because using such topic vectors for short texts leads to very sparse topic vectors, we did not experiment and do not report results with similarity methods based on just topic vectors.

The work presented here extends our previous work on LDA-based semantic similarity (Rus, Niraula, & Banjade, 2013). To the best of our knowledge, LDA has not been used so far for addressing the task of paraphrase identification in the context of student responses in dialogue-based ITSs, which is the focus of our work.

3 LDA-Based Similarity Measures

Latent Dirichlet Allocation (LDA; Blei, Ng,& Jordan, 2003) belongs to the broader category of methods called topic models. Topic models are based on the assumption that a relatively small set of latent topics underly natural language texts. The topics are groups of semantically related words. A word can belong to multiple topics. If one interprets each topic as a concept then LDA directly models polysemy which LSA does not. In LSA, each word has a unique vector representation. That is, multiple senses of the same word are mapped to the same representation in the reduced LSA space. Some argue that the LSA vector for a given word represents an average of all the senses of the word, while others argue that it represents the dominant, most frequent sense. Given this theoretical advantage of LDA over LSA when it comes to modeling word meanings, one wonders which one is better at tasks in which word meanings play a role such as sentence-level text-to-text similarity. This paper is a step towards understanding the strengths of LDA versus LSA.

It is important to add that LDA has been proposed to address several limitations of the earlier Probabilistic Latent Semantic Indexing model (pLSI; Hoffman, 1999). For instance, the pLSI model cannot handle unseen documents. Also, the number of parameters to be estimated in the pLSI models increases linearly with the number of documents leading to overfitting.

3.1 Latent Dirichlet Allocation

LDA is a generative probabilistic model for collections of discrete items, i.e. words in our case. The only observed things are the words (denoted by *w*) in documents. All else are latent variables. LDA derives the parameters of the latent variables using only the observed words in the corpus. Thus, LDA captures significant intra-document statistical structure via mixing distributions.

We will use the notation in Blei, Ng, and Jordan (2003) to explain the basic LDA model. A word, denoted *w*, is a discrete unit entry in a vocabulary V whose elements are indexed {1,…,V}. A document is a sequence of N words denoted **w = <w_1, w_2, …, w_N,>**, where w_i is the *i*-th word in the document. A corpus D is a collection of documents D = {$\mathbf{w}_1$, $\mathbf{w}_2$, …, $\mathbf{w}_M$}.

Documents are regarded as random mixtures of topics and a topic is a distribution over words in the vocabulary. LDA follows the following generative process for a document **w**.

(a) Choose a topic distribution $\theta \sim \text{Dir}(\alpha)$; the dimensionality *k* (number of topics) of the Dirichlet distribution is given;
(b) For each of the N words w_i in **w:**
  (i) Select a topic z_i based on θ;
  (ii) Choose a word w_i using $p(w_i | z_i, \beta)$

LDA has two Dirichlet priors: α for document-topic distributions and β for topic-word distributions. These two priors, α and β, are also known as hyper-parameters for the document-topic and topic-word Dirichlet distributions. Although they can be vector valued, many LDA implementations use α and β as scalars to simplify and get symmetric Dirichlet priors. Currently most LDA users choose symmetric Dirichlet priors using some heuristics. One such heuristics is mentioned by Steyvers and Griffiths (2006): although the values of these priors depend on vocabulary size and the number of topics, setting $\alpha = 50/k$ and $\beta = 0.01$ worked well for many different text collections. We followed this latter approach in our work presented here.

LDA estimation includes learning the various distributions, e.g., the set of topics, the word probabilities for each topic, the topic mixture proportion of each document, and the topic of each word in each document. Estimation of the LDA parameters directly and exactly maximizing the likelihood of the whole data collection is intractable. Approximate estimation methods are used to solve the problem. The three popular methods reported in the literature are: variational methods (Blei et al, 2003), expectation propagation (Griffiths and Steyvers, 2004), and Gibbs sampling (Griffiths and Steyvers, 2004). We used in our work an implementation based on Gibbs sampling (i.e., JGibbLDA).

3.2 Number of Topics

The standard LDA model requires the specification of the number of latent topics in advance. That is, the number of topics is set by the user. Choosing the right number of topics is important as they determine the quality of the LDA model. Many believe that choosing the right value for the number of topics is more art than science.

One solution is to try a range of values and choosing the best number of topics according to some intrinsic criterion, such as the coherence of the topics, or according to some extrinsic criterion such as accuracy on a task, .e.g. paraphrase identification. We use in this paper as a starting point the topic coherence for selecting the number of topics (see next subsection). Furthermore, our experiments with using LDA for the task of paraphrase identification can be viewed as an extrinsic, task-based selection or validation of the number of topics.

Other methods to select the number of topics exist. Some rely on heuristics for selecting the number of topics. Nonparametric Bayesian models such as Hierarchical Dirichlet process were also proposed to automatically estimate the number of topics (Teh et al., 2004). The nonparametric models are not computationally efficient (Wallach et al., 2009).

3.3 Assessment of Topics

As mentioned, we used topic coherence as an intrinsic criterion to select the number of topics upfront. Newman et al. (2010) have explored techniques for measuring topic coherence and presented a comparative study of topic coherence evaluation using Wikipedia, Google n-gram dataset, and WordNet. The pointwise mutual information (PMI) method was best when compared to human judgments of topic coherence. They counted the frequency of the co-occurring words in a window of 10-word in Wikipedia corpus and 5 in case of Google 5-grams.

Similarly, we used the average PMI of the top 10 and also top 20 words to assess the quality of topic coherence. That is, we formed all possible pairs with the top 10 or 20 words in each topic (words in each topic are decreasingly ordered based on their contribution to the topic) and computed the PMI for each pair based on word frequencies derived from a Wikipedia-based corpus.

The PMI was calculated using 4,134,837 English-language Wikipedia articles dumped on January 3, 2013. It contained 1,284,156,826 tokens and 5,693,208 word types (i.e. unique words) counted after removing digits and punctuation and changing to lower case. After removing the stop words, the number of tokens was 672,542,579. We found that a 100-topic LDA model leads to highest average topic coherence (we varied the number of topics from 10 to 300, the typical dimensionality used in LSA spaces). Experimental results on the paraphrase identification task, which can be viewed as an extrinsic, task-based evaluation of topic coherence, confirmed that using k=100 topics is best. Given that measuring topic coherence based on the average PMI of top 10 words recommends the same best number of topics as our task-based evaluation further supports the use of top 10 words PMI for measuring topic coherence (as suggested by Newman et al., 2010). The best coherence when using top 20 words is for a 20-topic LDA model. However, the average PMI for the 20 topics model is not significantly different from the 100 topics model.

In a way, our extrinsic, task-based validation of the number of topics is stronger than the validation based on human judgments provided by Newman and colleagues (2010) as they asked human judges to assess only a subset of the topics. Furthermore, it is not clear whether they asked the human judges to consider only top 10 words from each topic or not. They used the top 10 words only when computing the PMI.

3.4 LDA-Based Semantic Similarity Measures

As we already mentioned, LDA is a probabilistic generative model in which documents are viewed as distributions over a set of topics and each word in a document is generated based on a distribution over words that is specific to each topic.

A first semantic similarity measure among words would then be defined as a dot-product between the corresponding vectors representing the contributions of each word to a topic. It should be noted that the contributions of each word to the topics does not constitute a distribution, i.e. the sum of contributions does not add up to 1. Assuming the number of topics T, then a simple word-to-word measure is defined by the formula below where we denote by φ distributions over words for a topic *t*.

$$LDA - w2w(w,v) = \sum_{t=1}^{T} \varphi_t(w)\varphi_t(v)$$

More global text-to-text similarity measures could be defined in several ways. Because a document is a distribution over topics, the similarity of two texts needs to be computed in terms of similarity of distributions. The Kullback-Leibler (KL) divergence defines a distance, or how dissimilar, two distributions p and q are as in the formula below.

$$KL(p,q) = \sum_{i=1}^{T} p_i \log \frac{p_i}{q_i}$$

If we replace p with θ_d (text/document d's distribution over topics) and q with θ_c (text/document c's distribution over topics) we obtain the KL distance between two documents (documents d and c in our example).

Furthermore, KL can be used to compute the distance between two topics using their distributions over words (φ_{t1} and φ_{t2}). The KL distance has two major problems. In case q_i is zero KL is not defined. Furthermore, KL is not symmetric which does not fit well with semantic similarity measures which in general are symmetric. That is, if text A is a paraphrase of text B that text B is a paraphrase of text A. The Information Radius (IR) measure solves these problems by considering the average of pi and qi as below.

The IR can be transformed into a similarity measure as in the following (Dagan, Lee, & Pereira, 1997):

$$SIM(p,q) = 10^{-\delta IR(c,d)}$$

All our results reported here for LDA similarity measures between two documents c and d are computed by multiplying the similarities between the distribution over topics (θ_d and θ_c) and distribution over words (φ_{t1} and φ_{t2}). For space reasons, we do not provide all the details.

The Hellinger distance between two distributions is another option that allows avoiding the shortcomings of the KL distance.

$$HD(p,q) = \frac{1}{\sqrt{2}} \sqrt{\sum_{1}^{T} (\sqrt{p_i} - \sqrt{q_i})^2}$$

The Hellinger distance varies from 0 to 1 and is defined for all values of p_i and q_i. A value of 1 means the distance is maximum and thus the distributions are very different. A value of 0 means the distributions are very similar. We can transform the Hellinger distance into a similarity measure by subtracting it from 1 such that a zero distance means a large similarity score and vice versa.

Lastly, we used the Manhattan distance between distributions p and q as defined below.

$$MD(p,q) = 2 \times (1 - \sum_{1}^{T} \min(p_i, q_i))$$

MD is symmetric, defined for any values of p and q, and ranges between 0 and 2. We can divide MD by 2 and subtract from 1 to transform it into a similarity measure.

4 From Word Representations to Text-to-Text Similarity

As mentioned, we focus in this paper on two categories of methods: those that rely on word-to-word similarity measures and those that compute similarity globally, i.e. avoiding word-to-word similarities. In LSA, text-to-text similarity can be computed directly using the global vectors of each sentence which are obtained by summing up the individual word vectors. In LDA, global text-to-text similarity measures can be computed using the distributions over topics and over words without the need for word-to-word similarity measures.

Word-to-word similarity measures can be expanded to work at text-to-text level using greedy (see Lintean et al., 2010) or optimal matching algorithms (Rus & Lintean, 2012). We experimented with a method that guarantees optimal overall best match using the job assignment algorithm, a well-known combinatorial optimization problem. The assignment problem can be formulated as finding a permutation π for which $S_{OPT} = \sum_{i=1}^{n} w(s_i, t_{\pi(i)})$ is maximum where w(si,tπ(i)) is the fitness of worker si to job ti. Such an assignment is called optimum assignment. An algorithm, the Kuhn-Munkres method (Kuhn, 1955; Munkres, 1957), has been proposed that can find a solution to it in polynomial time.

In our case, we optimally match words in text T1 to words in text T2 based on how well the words in T1 fit the words in T2. The fitness between the words is nothing else but their word-to-word similarity according to some metric of word similarity, in our case LDA or LSA-based word-to-word measures.

5 Experimental Setup and Results

We present results with the previously described methods on the User Language Paraphrase Corpus (ULPC; McCarthy and McNamara, 2008) and additionally on the Microsoft Research Paraphrase corpus (MSRP; Dolan, Quirk, & Brockett, 2004). The ULPC corpus contains pairs of target-sentence and student response texts. These pairs have been evaluated by expert human raters along 10 dimensions of paraphrase characteristics. We used the ”Semantic Completeness” dimension that measures the semantic equivalence between the target-sentence and the student response on a binary scale, similar to the scale used in MSRP corpus. From a total of 1,998 pairs, 1,436 (71%) were classified by experts as being paraphrases. The data set is divided into three subsets: training (1,012 instances, 708-304 split of TRUE-FALSE paraphrases), validation (649 instances, 454-195 split), and testing (337 instances, 228-109 split). The average number of words per sentence is 15.

The MSRP corpus consists of 5,801 sentence pairs collected from newswire articles, 3,900 of which were labeled as paraphrases by human annotators. The whole set is divided into a training subset (4,076 sentences of which 2,753, or 67.5%, are true paraphrases), and a test subset (1,725 pairs of which 1,147, or 66.5%, are true paraphrases). A simple baseline for the MSRP corpus, the majority baseline when all instances are classified as positive, gives an accuracy and precision of 66.5% and perfect recall. The average number of words per sentence is 17 in this corpus.

We followed a training-testing methodology according to which we first trained to learn some parameters of the proposed model after which we used the learned values for the parameters on testing data. In our case, we learned a threshold for the text-to-text similarity score above which a pair of sentences is deemed a paraphrase and any score below the threshold means the sentences are not paraphrases. We report performance of the various methods using accuracy (percentage of correct predictions), F-measure (harmonic mean of precision and recall), and kappa statistics (a measure of agreement between our method's output and experts' labels while accounting for chance agreement).

We experimented with both word-to-word similarity measures and text-to-text similarity measures. The word-to-word similarity measures were expanded to work at sentence level using optimal matching. For LDA, we used the word-to-word measure and text-to-text measures described earlier. For LSA, we use the cosine between two words' LSA vectors as a measure of word-to-word similarity. For LSA-based text-to-text similarity we first add up the word vectors for all the words in a text thus obtaining two text vectors, one for each text, and then compute the cosine between these two text vectors.

An important step in the process of obtaining the LSA vectorial representation is the derivation of the semantic space, i.e. discovering the latent dimensions or concepts, from a large enough corpus. In our work, we experimented with an LSA space computed from the TASA corpus (compiled by Touchstone Applied Science Associates), a balanced collection of representative texts from various genres (science, language arts, health, economics, social studies, business, and others). The TASA corpus contains 10,937,986 words with a vocabulary size of 91,897 after removing stop words.

We varied the number of topics for the LDA model and observed changes in performance. Fewer topics usually means semantically less coherent topics as more words with different meaning will be grouped under the same topic. Our experiments revealed that using just top 10 or 20 words for measuring topic coherence indicates the opposite: the fewer the topics the higher their semantic coherence, e.g. topics sets of size 100, 40, or 20, all have higher average topic coherence scores compared to 200- or 300-topic models. We concluded that the 100, 40, and 20 models yield results similar to higher 200 and 300 topics models. That is, using 100 topics models could be a good choice that balances a sufficiently large number of topics and good topic coherence when addressing sentence-level semantic similarity tasks such as paraphrase identification.

We also present results obtained using 300 dimensions for the LSA space, a standard value, and a similar number of topics for LDA (see column T=300 in Table 1). This number of dimensions has been empirically established by LSA researchers to deliver best results. We also present results for 100 dimensions to compare with the best LDA model which corresponds to 100 topics.

Table 1. Results on ULPC (column 2 and 3) and MSRP (column 3 and 4) test data with LDA-based methods for various number of topics (T=100 represents the most coherent set of topics)

Method	***Accuracy/ Kappa/F-measure (T=300)***	***Accuracy/Kappa/ F-measure (T=100)***	***Accuracy/Kappa/ F-measure (T=300)***	***Accuracy/Kappa/F-measure (T=100)***
LDA-IR	71.17/16.17/81.94	68.24/3.09/80.92	67.47/4.52/79.87	67.01/3.15/79.98
LDA-Hellinger	71.32/18.85/81.75	68.24/2.46/80.99	67.36/4.39/79.73	67.18/3.50/80.04
LDA-Manhattan	71.07/10.10/82.50	71.21/23.41/81.16	66.78/3.56/79.91	67.18/4.04/80.04
LDA-Greedy	**77.32/34.40/85.75**	**76.85/37.89/84.94**	73.04/35.01/81.31	73.10/34.27/81.32
LDA-Optimal	76.97/36.96/85.06	75.96/36.75/84.14	**73.27/36.74/80.71**	**73.15/36.86/80.71**
LSA-Greedy	77.22/33.82/85.73	*Same*	72.86/33.89/81.11	*same*
LSA-Optimal	77.12/36.80/85.24	*Same*	73.04/35.95/80.80	*same*
LSA	**77.47/37.54/85.50**	*Same*	**73.56/34.61/81.83**	*same*

The results in Table 1 indicate that the best LDA-based methods rival the LSA based method. A combination of greedy matching and LDA word-to-word similarity yields best accuracy and F-measure results on the ULPC corpus while text-to-text similarity based on LSA yields best accuracy. The 100-topic LDA model produces similar accuracy results on ULPC and a higher kappa (kappa=37.89 for 100-topic model and kappa=34.40 for the 300-topic model).

Similarly, for the MSRP corpus the LDA models produce results very close to LSA. The 100-topic LDA model has a slightly better kappa score compared to the 300-topic model. The 100-topic models yields very similar accuracy score to the 300-topic model, and an identical F-measure score.

All the distance-between-distributions based LDA measures (top 3 rows in Table 1) yield modest results. This is mainly due to the sparsity of topic distributions in short texts compared to the size of the model in terms of number of topics. If a 100-topic model is used and the sentence has on average 15 words, in the best case scenario in which each word in the sentence corresponds to a unique topic, 85 of the remaining topics in the 100-topic model would have a probability of zero. This leads to small distances/large similarities between the corresponding topic distributions.

6 Discussion and Conclusions

We presented in this paper our work on defining semantic similarity measures at word and sentence level based on LDA. A measure based on word representations as vectors of topic contributions yields competitive results with the unsupervised algebraic method of LSA. Furthermore, Table 1 indicates that semantic similarity measures based on distances among distributions over words and topics (see the rows for LDA-IR, LDA-Hellinger, LDA-Manhattan) are not useful for short texts due to topic sparseness in short texts. We plan to investigate this issue in future work.

Acknowledgments. This research was supported in part by Institute for Education Sciences under awards R305A100875. Any opinions, findings, and conclusions or recommendations expressed in this material are solely the authors'.

References

1. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. The Journal of Machine Learning Research 3, 993–1022 (2003)
2. Celikyilmaz, A., Hakkani-Tür, D., Tur, G.: 2010. LDA Based Similarity Modeling for Question Answering. In: NAACL-HLT. Workshop on Semantic Search, Los Angeles, CA (June 2010)
3. Chen, X., Li, L., Xiao, H., Xu, G., Yang, Z., Kitsuregawa, M.: Recommending Related Microblogs: A Comparison between Topic and WordNet based Approaches. In: Proceedings of the 26th International Conference on Artificial Intelligence (2012)
4. Dagan, I., Glickman, O., Magnini, B.: Recognizing textual entailment (2004), `http://www.pascalnetwork.org/Challenges/RTE`
5. Dagan, I., Lee, L., Pereira, F.C.N.: Similarity Based Methods For Word Sense Disambiguation. In: ACL, pp. 56–63 (1997)
6. Dolan, B., Quirk, C., Brockett, C.: Unsupervised Construction of Large Paraphrase Corpora: Exploiting Massively Parallel News Sources. In: COLING 2004, Geneva, Switzerland (2004)
7. Fernando, S., Stevenson, M.: A semantic similarity approach to paraphrase detection. In: Proceedings of the Computational Linguistics UK, CLUK 2008 (2008)
8. Graesser, A.C., Olney, A., Haynes, B.C., Chipman, P.: Autotutor: A cognitive system that simulates a tutor that facilitates learning through mixed-initiative dialogue. In: Cognitive Systems: Human Cognitive Models in Systems Design. Erlbaum, Mahwah (2005)
9. Griffiths, T.L., Steyvers, M.: Finding scientific topics. Proceedings of the National Academy of Sciences of the United States of America 101(suppl. 1), 5228–5235 (2004)
10. Hofmann, T.: Probabilistic latent semantic indexing. In: Proceedings of SIGIR 1999, pp. 50–57 (1999)
11. Kozareva, Z., Montoyo, A.: Paraphrase identification on the basis of supervised machine learning techniques. In: Salakoski, T., Ginter, F., Pyysalo, S., Pahikkala, T. (eds.) FinTAL 2006. LNCS (LNAI), vol. 4139, pp. 524–533. Springer, Heidelberg (2006)
12. Kuhn, H.W.: The Hungarian Method for the assignment problem. Naval Research Logistics Quarterly 2, 83–97 (1955)
13. Landauer, T., McNamara, D.S., Dennis, S., Kintsch, W.: Handbook of Latent Semantic Analysis. Erlbaum, Mahwah (2007)
14. Lintean, M., Moldovan, C., Rus, V., McNamara, D.: The Role of Local and Global Weighting in Assessing the Semantic Similarity of Texts Using Latent Semantic Analysis. In: Proceedings of the 23rd International Florida Artificial Intelligence Research Society Conference, Daytona Beach, FL (2010)
15. Mimno, D., Wallach, H.M., Talley, E., Leenders, M., McCallum, A.: Optimizing semantic coherence in topic models. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 262–272. ACL (2011)
16. Munkres, J.: Algorithms for the assignment and transportation problems. Journal of the Society for Industrial and Applied Mathematics 5(1), 32–38 (1957)
17. Newman, D., Lau, J.H., Grieser, K., Baldwin, T.: Automatic evaluation of topic coherence. In: HLT-NACL, pp. 100–108. ACL (2010)
18. McCarthy, P.M., McNamara, D.S.: User-Language Paraphrase Corpus Challenge (2008)

19. Rus, V., Graesser, A.C.: Deeper natural language processing for evaluating student answers in intelligent tutoring systems. Paper Presented at the Annual Meeting of the American Association of Artificial Intelligence (AAAI 2006), Boston, MA, July 16-20 (2006)
20. Rus, V., Lintean, M.: A Comparison of Greedy and Optimal Assessment of Natural Language Student Input Using Word-to-Word Similarity Metrics. In: Proceedings of the Seventh Workshop on Innovative Use of Natural Language Processing for Building Educational Applications, NAACL-HLT 2012, Montreal, Canada, June 7-8 (2012)
21. Rus, V., Niraula, N., Banjade, R.: Similarity Measures Based on Latent Dirichlet Allocation. In: Gelbukh, A. (ed.) CICLing 2013, Part I. LNCS, vol. 7816, pp. 459–470. Springer, Heidelberg (2013)
22. Steyvers, M., Griffiths, T.: Probabilistic topic models. Handbook of Latent Semantic Analysis 427(7), 424–440 (2006)
23. Wallach, H., Mimno, D., McCallum, A.: Rethinking LDA: Why priors matter? Advances in Neural Information Processing Systems, 22, 1973–1981 (2009)
24. Teh, Y.W., Jordan, M.I., Beal, M.J., Blei, D.M.: Hierarchical dirichlet processes. Journal of the American Statistical Association 101(476), 1566–1581 (2006)

MDL-Based Models for Transliteration Generation

Javad Nouri[1], Lidia Pivovarova[1,2], and Roman Yangarber[1]

[1] University of Helsinki, Department of Computer Science, Finland
[2] St.Petersburg State University, Russia

Abstract. This paper presents models for automatic transliteration of proper names between languages that use different alphabets. The models are an extension of our work on automatic discovery of patterns of etymological sound change, based on the Minimum Description Length Principle. The models for pairwise alignment are extended with algorithms for prediction that produce transliterated names. We present results on 13 parallel corpora for 7 languages, including English, Russian, and Farsi, extracted from Wikipedia headlines. The transliteration corpora are released for public use. The models achieve up to 88% on word-level accuracy and up to 99% on symbol-level F-score. We discuss the results from several perspectives, and analyze how corpus size, the language pair, the type of names (persons, locations), and noise in the data affect the performance.

1 Introduction

The task of machine transliteration involves mapping the representation of a word to another language, typically using a different alphabet, based on its sound or spelling, rather than its meaning. Transliteration is commonly applied to proper names, as well as to terms in rapidly growing areas, such as medicine or technology, [12]. Two principal applications for machine transliteration are machine translation and information search—multilingual information retrieval (IR), information extraction (IE), and named-entity recognition (NER). While in machine translation the goal may be to produce only one correct transliteration for each word, in other tasks we may wish to produce several possible transliterations, and merge different variants of the same name.

There are two main approaches to machine transliteration: transliteration generation and transliteration mining (discovery). In transliteration generation one builds a transliteration model, which takes a source named entity as input and produces its representations in the target language as output.

Transliteration generation can be considered in a broad sense as a special case of alignment and transduction of words. The similarity of machine transliteration and alignment of etymologically-related words have been observed by [14], who applied the same model, a finite state transducer, for both tasks. In [1], transliteration is mentioned in the broader context of linguistic string-transduction tasks, such as paraphrasing, morphological transformation and co-reference resolution.

The definition that [3] proposes for *cognate*—"words with a common form and meaning across the languages"—is applicable to both transliterated and etymologically related words. Machine transliteration is, of course, a different task from cognate alignment. The main principle for word changes in etymology is the regularity of sound change, whereas transliteration of names may not always follow regular rules. Transliteration does not always follow pronunciation in practice—it can be based on script, or tradition, or on translation. For example, the French name *Jean* [ʒɑ̃] used to be transliterated into Russian as *Жеан* [ʐɛan], i.e., based on its French *spelling* rather than its pronunciation. The same name may be transliterated into another language differently depending on tradition: e.g., the name of the famous Russian author *Лев Толстой* is commonly *translated* into English as *Leo Tolstoy*; the name of his son, which is identical in Russian, is commonly *transliterated* as *Lev Tolstoy*. Furthermore, transliteration rules may be different in different domains [6].

Thus, there may be more noise in transliteration data than in etymological data. On the other hand, words in remotely related languages, (e.g., Finnish–Hungarian), will have substantially more complex correspondences. Therefore, while the tasks of transliteration and etymological alignment may be similar from the algorithmic point of view, it is not *a priori* obvious that the same algorithms will work for both tasks.

The models that we use for transliteration were originally developed for automatic discovery of patterns in etymological sound change, [16]. We apply these models to 13 parallel corpora in 7 languages: Farsi, English, Russian, Greek, Hebrew, French, and Japanese (Katakana script for foreign names), which are extracted from Wikipedia headlines using *language links*. We report results on automatic transliteration of names of American actors from English into four languages that use different kinds of writing systems: alphabetic (Russian and Greek) and consonantal (Farsi and Hebrew). We also examine the performance on data of different semantic type—person names vs. location names. For example, for English-Russian transliteration we use 3 different datasets: person names of English origin (American actors), person names of Russian origin (Russian writers), and location names of Russian origin (Russian cities).

As far as we are aware, the comparison of transliteration results among different semantic types has not been addressed in the literature to date; [10] studied the influence of the origin of names and of noise in the data on the results of transliteration, but they did not apply their method to words of different types.

2 Related Work

A comprehensive survey in [9] classifies machine transliteration generation methods into several broad categories: rule-based, phonetics-based, spelling-based, hybrid—a mix of spelling and phonetics, and combined—applying several methods and then selecting the best transliteration via re-ranking. According to this classification scheme, the models we use would be classified as spelling-based, as they use only the spelling of the words with no information about their pronunciation. Some of our etymological models use phonetic features [18]; they will be applied to the transliteration task in future work.

The most common approach for spelling-based transliteration uses source-channel models, [11,4]. Many methods used for machine transliteration are adopted from phrase-based statistical machine translation [15,5]. The importance of the representation of word-pairs is emphasized in [6], i.e., the features used in the transliteration task. The representation in [6] combines uni-grams and bi-grams, which are then used as features in an optimization task. It was shown in [12] that sequential n-grams outperform the bag-of-word n-gram models.

A Minimum Description Length (MDL) approach to transliteration is described in [21]. This work, as well as others [13,8], uses the Expectation-Maximization (EM) algorithm: in the Expectation stage the probabilities of substring correspondence are counted; in the Maximization stage word pairs are re-aligned using these probabilities.

Automatic transliteration and evaluation has been explored in a series of workshops on Named Entities, which organized shared tasks in transliteration generation and mining [23].

3 Data

We use datasets that we extracted from cross-language links in the Wikipedia. We did not try to extract as much data as possible; rather, we focused on language and on topical homogeneity of each dataset. In this paper, we present work on Wikipedia dumps dated up to 12 December 2012.[1] We used only the page titles, language links, and category links. The full text of the articles is not used in the transliteration task, with one exception, below. To extract the datasets we used Wikipedia Categories; we tried to focus on categories that can guarantee higher homogeneity in the data. For example, the majority of names in category *American actors* are of English origin, and most of the names in category *Russian Writers* are of Russian origin. However, there are many exceptions among person names; this is especially true for languages such as English or Russian, which are in use across wide geographic areas.

We also used location names, since toponyms may be more stable and more consistent than person names. For the dataset of Iranian Locations, we parsed the content of the corresponding Wikipedia pages. Since the number of titles for Iranian cities in the Russian Wikipedia was small, we collected names of towns and locations from Russian-Wikipedia pages about Iranian *Shahrestans* (counties). The data used our experiments is summarized in Table 1.

Each dataset was semi-automatically cleaned. Some amount of noise was intentionally left in data; e.g., the name of the Russian city Санкт-Петербург (*[Sankt-Peterburg]*) is usually "transcribed" into English as *Saint-Petersburg*, because this is a commonly accepted translation, although it is not phonetically accurate. We removed patronymics, which are very common in the Russian data,

[1] Wikipedia dumps are available under the GNU Free Documentation License and Creative Commons License at the Wikimedia Web-site.

Table 1. Transliteration corpora/datasets, extracted from Wikipedia headlines and language links. The language codes are: En–English, Fa–Farsi, Fr–French, Gr–Greek, He–Hebrew, Jp–Japanese (Katakana), Ru–Russian.

Dataset	**Language pair**	**Size: # of pairs**	**Dataset**	**Language pair**	**Size: # of pairs**
American Actors	En–Ru	1471	*Russian Cities*	Ru–En	1136
	En–He	1245		Ru–Fa	870
	En–Fa	840		Ru–Fr	828
	En–Gr	407		Ru–Jp	317
Russian Writes	Ru–En	1462	*French Cities*	Fr–Ru	828
Iranian Cities	Fa–En	439	*Iranian Locations*	Fa–Ru	1893
	Fa–Ru	469			

since in most cases they are omitted in other languages. We also removed all accent marks from the Greek dataset. Since these marks are obligatory in Greek script, transliteration models based on these data cannot be used in real-world application, though this data still interesting for a transliteration task.[2]

It was stated in [23] that "a reasonably large" dataset for the transliteration task should consist of ∼10 000 name pairs, which is orders of magnitude larger than our datasets. Some authors report satisfactory results using considerably smaller datasets. For example, a word-level accuracy of 33% for Arabic-to-English transliteration is reported using a training set of only 935 name pairs, and an accuracy of 46% for Russian-to-English, using a training set of 545 name pairs [22]. In this paper we present models that achieve good results (Section 6) on relatively small datasets; DirecTL+ [7], which we use as a baseline for comparison, also demonstrates reasonable results on our datasets.

4 Method

We use *Etymon*, a set of MDL-based models that we developed for analyzing etymological data, as the basis for our transliteration models.[3] The collection of models is described in [16,18,17]. The models take as input a phonetic representation of genetically related words, or *cognates*, and aim to discover regular phonetic changes between languages within a language family. This is done by searching for the best *pairwise alignment* of words, by optimizing the description length of the alignments. Transliteration is an analogous task, since in order to learn how to transliterate from one language to another, it seems that a natural prerequisite is to align the words in the training set. Thus, it seemed reasonable to suppose that we could use these models for aligning the data. After alignment, we introduce a set of *prediction* procedures for performing the actual transliteration—based on the alignment. We briefly describe the models; detailed explanations can be found in [16,18]—and the prediction procedure.

[2] We consider recovery of accents a separate task, beyond the scope of this work.
[3] The tools are publicly available from `http://etymon.cs.helsinki.fi/`

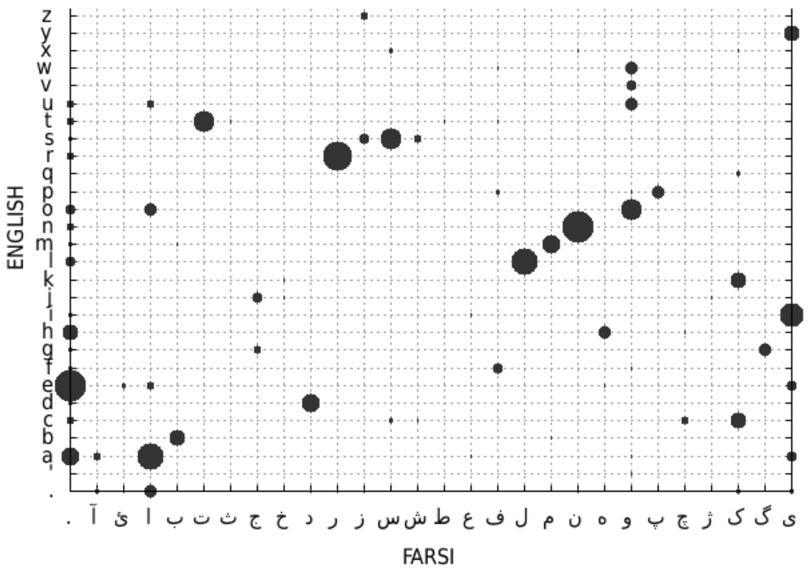


Fig. 1. English–Farsi alignment matrix; *American Actors* dataset. The size of each ball indicates the probability of the corresponding symbol-pair alignment.

4.1 1×1 Alignment

Our baseline 1×1 model finds an optimal alignment, where each symbol of the source word may align to at most one symbol of the target word, with possible insertions or deletions. Information about the context of the symbols is not used.

For example, the alignment matrix for English to Farsi transliteration on the *American Actors* is shown in Figure 1. The matrix shows that, e.g., English **e** is most frequently aligned to "." due to omission of short vowels in Farsi script. Mapping English **a** to Farsi آ is rare, as seen from the size of the corresponding bubble; this happens when **a** designates [a] that is situated at the beginning of the word. The 1×1 model is unable to capture this rule; the only information it uses is that **a** is transliterated to ا more frequently than to any other symbol.

4.2 2×2 Alignment

The 2×2 model extends the 1×1 model, by allowing for alignment of up to two consecutive symbols at a time on each level; this model also takes into account the start and end word boundaries. Unlike the 1×1 model, this model captures some information about the symbols' context when learning the correspondences. For example, it should discover cases where one symbol in language *A* corresponds to two symbols in language *B*; e.g., Russian 'ч' is often transliterated as "ch" in English—the 1×1 model is by definition unable to discover such correspondences. The 2×2 model can also discover that certain symbols are transliterated differently when they appear at the beginning or at the end of a word, as in the example of **a** and آ mentioned above, in Section 4.1. For example, the name *Alda* from the *American Actors* dataset is correctly transliterated into Farsi as آلدا by the the 2×2 model—with آ at the beginning. The 1×1 model incorrectly predicts the transliteration الدا—with ا as the most probable correspondence for **a** in both initial and final position, unable to exploit the context information.

4.3 Prediction

Once all word pairs in the training set are aligned, the discovered symbol correspondences can be used to predict transliterations for new, unseen words. The result of the convergence of the alignment algorithm is a count matrix, such as one shown in Figure 1, which indicates how often each symbol (in the 1×1 model)—or pair of symbols (in the 2×2 model)—of the source language is aligned to symbols in the target language in an optimal way. We have implemented an algorithm that predicts the target representation of a given source word based on the alignment. Theoretically, this is done by searching among *all* possible strings in the target language to select the string that yields the lowest cost under the model, when aligned with the source word. In practice, this can be achieved efficiently, by using simple table lookup for the baseline 1×1 model, and by a Dynamic Programming algorithm for the more complex models. Prediction based on the 1×1 model is straightforward, since symbols are aligned independently of their context; we assign to each source symbol the single target symbol to which it is cheapest to align:

$$t_i = \arg\min_{t \in T} L(s_i, t) \tag{1}$$

where s_i is the ith symbol in source word, T is the alphabet of the target language augmented with the special symbol '.' to allow for deletions, and $L(x, y)$ is the cost (code-length) of aligning the source-language symbol x to the target-language symbol y under the learned model.

The 2×2 prediction is more complicated, since it is possible to align zero, one, or two source symbols to symbols of the target language, and we need to choose the lowest cost alignment for the entire source word. We solve this optimization problem using Dynamic Programming (DP). To predict a target word, the algorithm starts from the beginning of the source word, and for each symbol s_i, finds the best prediction up to s_i based on previously computed partial alignments. The algorithm computes the cost $L(i)$ of the best prediction up to s_i, for all i. Thus, for predicting the best target sequence corresponding to the source word *up to* the i-th symbol, s_i, the possible *final* candidate alignments are in the set C, where:

$$C = \big\{(s_i : .), (s_i : t), (s_i : tt'), (s_{i-1}s_i : .), (s_{i-1}s_i : t), (s_{i-1}s_i : tt')\big\},$$

and t and t' are symbols from the target alphabet. Each of these 6 alignments has a fixed cost under the learned 2×2 model. The optimal prediction up to the i-th source symbol is given by minimizing the sum of one of these alignments, plus the cost $L(i-1)$ of the optimal alignment up to symbol s_{i-1}, for the first three candidates in C, or $L(i-2)$, the optimal cost up to s_{i-2} for the last three candidates in C,—where $L(i-1)$ and $L(i-2)$ have been pre-computed by DP previously. Using this approach the best target word—under the model—can be predicted in linear time.

5 Evaluation

The quality of machine transliteration depends on the ultimate task for which transliteration is being developed. In multilingual IE and IR, the system may make use of multiple possible variants. For example, in many cases more than one transliteration may be acceptable for a particular name, but if the system populates a database with events or relationships, identifying and merging different references to the same real-world entity is needed across multiple sources, [20,19]. In a multi-lingual setting, this capability is indispensable, [2]

Here, we evaluate performance of the transliteration models at the word level and at the symbol level. The most common word-level measure is accuracy [9]:

$$A = \frac{\text{number of correct transliterations}}{\text{total number of test words}} \tag{2}$$

Symbol-based evaluation measures are more diverse than word-based ones; in general, they are based on an edit distance between the system response and the expected transliteration. In this paper we use Normalized Edit Distance and Mean F-score. The normalized edit distance is computed as:

$$NED = \frac{\sum_i ED(c_i, r_i)}{\sum_i |c_i|} \tag{3}$$

where c_i is the expected transliteration for word i, r_i is the system response, and $ED(c_i, r_i)$ is an edit distance (here, the Levenshtein edit distance).

The symbol-level Mean F-score [23] is based on the Longest Common Subsequence between an expected transliteration c and the system response r[4]:

$$LCS(c,r) = \frac{1}{2}(|c| + |r| - ED'(c,r)) \tag{4}$$

Recall, Precision and F-score for a particular word are calculated on the basis of LCS (distance ED' allows insertions and deletions and no substitutions):

$$P = \frac{LCS(c,r)}{|r|} \qquad R = \frac{LCS(c,r)}{|c|} \qquad F = 2\frac{R \times P}{R + P} \tag{5}$$

We average the F-score over all words to get the mean over the entire data set.

[4] We slightly simplify all formulae here, assuming only one expected transliteration and one system response for each word.

Alongside our models, we use two other models for comparison. One is a naive baseline, where each symbol of source alphabet is transliterated as fixed symbol (or a string of symbols) from the target alphabet. In many cases this is a one-to-one mapping, but there are many exceptions; e.g., the Russian щ corresponds to English *shch* while Russian ь is most frequently omitted in transliteration. We did not apply this baseline to Katakana, since it is difficult to make reasonable correspondence between Katakana and Russian symbols.

The second model we used for comparison is the open-source system DirecTL+ [7]. It uses aligned data as input for the training; for alignment, we use the M2M-aligner, an open source program by the same authors, [8].[5]

We evaluate the models' performance via leave-one-out cross-validation.

6 Results

The results are shown in Tables 2 and 3, followed by the overall scores, average over all datasets that we tested. Although on some of the datasets, DirecTL+ beats the Etymon models, Etymon's performance appears higher overall.

One shortcoming of this evaluation scheme may be that only one correct answer is permitted for each word pair. For example, in the *American Actors* dataset, the English surname *Murray* is transliterated into Russian in two different ways: twice as Мюррей and twice as Мюррэй—therefore, for this name (*Murray*) any model can get at most 50% accuracy at the word level. We did not measure how this ambiguity ultimately affects the evaluation results, though it is common for person names.

By comparison, location names are more consistent; in most cases the toponyms are older and represent a more homogeneous transliteration scheme. Loan words and repetitions are more rare among location names. Thus, the results on location data are in general higher. For example, if we consider the results on three English-Russian datasets, namely *American Actors*, *Russian Writers* and *Russian Cities*, we can see from the tables that for both forward and backward transliteration the highest performance is achieved on the *Russian Cities* dataset. Comparing the datasets *Russian Writers* and *Russian Cities* is informative: both datasets use the same language pair, have the same language of names origin, and approximately the same size. However, we observe a difference of 20% in word-level accuracy on Ru-En transliteration and 14% on En-Ru, due to differences in the nature of the names.

It is also interesting to compare the *Iranian Cities* and *Iranian Locations* datasets for Russian-Farsi transliteration. As was described in Section 3, the latter contains a list of the Shahrestan's locations with population over 800. The dataset is four times larger, but it is also more noisy: Wikipedia editors seem to pay less attention to transliteration of smaller place names. In fact, we have

[5] We use default parameters for both programs. It may be possible to achieve better results through elaborate tuning of the parameters, though we did not explore parameter tuning. By comparison, our Etymon models have no parameters to tune.

Table 2. Transliteration results

Size: # of pairs	**Model**	**Word level Accuracy**	**NED**	**Mean F-Score**	**Word level Accuracy**	**NED**	**Mean F-Score**
American Actors							
		En → Fa			Fa → En		
840	1x1	0.223	0.256	0.816	0.081	0.371	0.703
	2x2	**0.393**	**0.180**	**0.867**	0.080	0.346	0.730
	Baseline	0.233	0.273	0.817	0.032	0.433	0.641
	DirecTL+	0.157	0.363	0.797	**0.118**	**0.324**	**0.756**
		En → Gr			Gr → En		
407	1x1	0.157	0.312	0.776	0.079	0.385	0.692
	2x2	**0.437**	**0.171**	**0.878**	**0.268**	**0.238**	**0.812**
	Baseline	0.179	0.343	0.750	0.101	0.456	0.650
	DirecTL+	0.342	0.232	0.849	0.140	0.417	0.693
		En → He			He → En		
1245	1x1	0.160	0.301	0.764	0.070	0.382	0.696
	2x2	**0.415**	**0.186**	**0.868**	0.104	0.337	0.738
	Baseline	0.074	0.426	0.725	0.043	0.430	0.640
	DirecTL+	0.160	0.331	0.817	**0.131**	**0.327**	**0.755**
		En → Ru			Ru → En		
1471	1x1	0.338	0.222	0.815	0.309	0.223	0.814
	2x2	**0.430**	**0.176**	**0.851**	**0.388**	**0.177**	0.853
	Baseline	0.298	0.250	0.799	0.282	0.250	0.795
	DirecTL+	0.387	0.214	0.834	0.373	0.189	**0.854**
Russian Cities							
		En → Ru			Ru → En		
1136	1x1	0.448	0.113	0.904	0.509	0.082	0.957
	2x2	**0.762**	**0.040**	**0.972**	**0.881**	**0.018**	**0.989**
	Baseline	0.379	0.176	0.868	0.823	0.028	0.983
	DirecTL+	0.501	0.163	0.886	0.813	0.028	0.985
		Fa → Ru			Ru → Fa		
870	1x1	0.180	0.230	0.815	0.441	0.110	0.924
	2x2	0.302	**0.170**	**0.866**	**0.684**	**0.060**	**0.964**
	Baseline	0.125	0.264	0.781	0.507	0.098	0.928
	DirecTL+	**0.325**	0.190	0.852	0.514	0.100	0.947
		Ru → Jp			Jp → Ru		
317	1x1	0.013	0.552	0.470	0.016	0.377	0.733
	2x2	**0.565**	**0.126**	**0.904**	**0.300**	**0.145**	**0.876**
	DirecTL+	0.022	0.742	0.541	0.287	0.159	0.870
		Ru → Fr			Fr → Ru		
828	1x1	0.389	0.124	0.930	0.355	0.154	0.890
	2x2	0.697	0.051	0.968	**0.668**	**0.065**	**0.953**
	Baseline	0.383	0.122	0.914	0.307	0.210	0.864
	DirecTL+	**0.736**	**0.042**	**0.973**	0.396	0.189	0.873

Table 3. Transliteration results, continued, including overall averaged scores.

Size: # of pairs	Model	Word level Accuracy	NED	Mean F-Score	Word level Accuracy	NED	Mean F-Score
Russian Writers							
		En → Ru			Ru → En		
1462	1x1	0.400	0.153	0.878	0.415	0.126	0.920
	2x2	**0.634**	**0.091**	**0.934**	**0.689**	**0.073**	0.943
	Baseline	0.347	0.201	0.856	0.651	0.075	**0.944**
	DirecTL+	0.462	0.176	0.875	0.588	0.090	0.933
French Cities							
		Ru → Fr			Fr → Ru		
828	1x1	0.088	0.338	0.745	0.113	0.357	0.715
	2x2	0.148	0.297	0.776	**0.381**	**0.182**	**0.863**
	Baseline	0.075	0.376	0.704	0.081	0.471	0.699
	DirecTL+	**0.199**	**0.259**	**0.808**	0.176	0.381	0.767
Iranian Cities							
		En → Fa			Fa → En		
439	1x1	0.196	0.334	0.787	0.109	0.280	0.817
	2x2	**0.435**	**0.155**	**0.896**	0.228	0.205	0.857
	Baseline	0.175	0.353	0.789	0.057	0.282	0.790
	DirecTL+	0.132	0.391	0.786	**0.289**	**0.185**	**0.863**
		Ru → Fa			Fa → Ru		
469	1x1	0.382	0.197	0.856	0.134	0.282	0.803
	2x2	**0.525**	**0.139**	**0.890**	**0.252**	0.237	0.827
	Baseline	0.267	0.277	0.803	0.092	0.296	0.775
	DirecTL+	0.151	0.332	0.800	0.222	**0.210**	**0.846**
Iranian locations							
		Ru → Fa			Fa → Ru		
1893	1x1	0.380	0.201	0.863	0.135	0.274	0.812
	2x2	**0.553**	**0.134**	**0.902**	0.278	0.217	0.841
	Baseline	0.285	0.270	0.816	0.078	0.318	0.752
	DirecTL+	0.155	0.345	0.813	**0.317**	**0.189**	**0.854**
	Results averaged over all datasets						
	1x1	0.235	0.259	0.804			
	2x2	**0.442**	**0.162**	**0.878**			
	Baseline	0.245	0.278	0.795			
	DirecTL+	0.311	0.253	0.832			

found many inaccuracies among the *Iranian Locations*. For example, the Iranian place-name بوئین /buin/ appears in Russian as Бу /bu/. Due to such noise in the data, for these datasets we achieved approximately the same results according to all measures, although the number of word pairs in the *Iranian Cities* dataset (439) is four times smaller than in the *Iranian Locations* dataset (1893). This may mean that it is possible to use quite small training sets for transliteration, if the data are highly homogeneous and clean.

7 Discussion and Current Work

To summarize, the main contributions of the presented work are: we provide a new, simple, and manually verified *data set* for evaluation of transliteration models; we apply models built for etymological alignment to the task of cross-lingual transliteration; we introduce simple extensions for *prediction* to the alignment models, which yield procedures for transliteration based on the alignment. We attempt to ground this work clearly in the context of other related approaches.

The MDL-based Etymon models, applied to the transliteration task without significant modifications, have achieved results that are comparable with state-of-the-art methods reported in the literature. We have discussed how the nature of data, as well as its homogeneity, impacts performance quality.

Current work includes adapting Etymon's *context-sensitive* models for transliteration. These models were shown, [18], to achieve substantially lower compression cost and normalized edit distance than the 1×1 and 2×2 models. We are implementing the prediction algorithm for these models, which is more complex and requires a target language model. Another complication is that the context models require each symbol to be represented as a vector of phonetic features. Thus, the next step will be an implementation of phonetic representations of the data. We also plan to expand our datasets by including more language pairs, and more complex types of data, including company names.

References

1. Andrews, N., Eisner, J., Dredze, M.: Name phylogeny: A generative model of string variation. In: Proceeding of the 2012 Joint Conference of Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL (2012)
2. Atkinson, M., Piskorski, J., van der Goot, E., Yangarber, R.: Multilingual real-time event extraction for border security intelligence gathering. In: Wiil, U.K. (ed.) Counterterrorism and Open Source Intelligence. Springer Lecture Notes in Social Networks, vol. 2 (2011)
3. Bergsma, S., Kondrak, G.: Alignment-based discriminative string similarity. In: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (2007)
4. Ekbal, A., Naskar, S.K., Bandyopadhyay, S.: A modified joint source-channel model for transliteration. In: Proceedings of the COLING/ACL, Stroudsburg, PA (2006)
5. Finch, A., Sumita, E.: Phrase-based machine transliteration. In: Proceedings of the Workshop on Technologies and Corpora for Asia-Pacific Speech Translation, TCAST (2008)
6. Goldwasser, D., Roth, D.: Transliteration as constrained optimization. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (2008)
7. Jiampojamarn, S., Cherry, C., Kondrak, G.: Joint processing and discriminative training for letter-to-phoneme conversion. In: Proceedings of ACL 2008: HLT, Columbus, Ohio (2008)

8. Jiampojamarn, S., Kondrak, G., Sherif, T.: Applying many-to-many alignments and hidden markov models to letter-to-phoneme conversion. In: Human Language Technologies 2007: North American Chapter of the Association for Computational Linguistics, Rochester, New York (2007)
9. Karimi, S., Scholer, F., Turpin, A.: Machine transliteration survey. ACM Computing Surveys 43(3) (2011)
10. Karimi, S., Turpin, A., Scholer, F.: Corpus effects on the evaluation of automated transliteration systems. In: Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (2007)
11. Li, H., Zhang, M., Su, J.: A joint source-channel model for machine transliteration. In: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics (2004)
12. Lindén, K.: Multilingual modeling of cross-lingual spelling variants. Information Retrieval 9(3) (2006)
13. Pervouchine, V., Li, H., Lin, B.: Transliteration alignment. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (2009)
14. Schafer, C.: Novel probabilistic finite-state transducers for cognate and transliteration modeling. In: 7th Biennial Conference of the Association for Machine Translation in the Americas (AMTA) (2006)
15. Sherif, T., Kondrak, G.: Substring-based transliteration. In: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (2007)
16. Wettig, H., Hiltunen, S., Yangarber, R.: MDL-based Models for Alignment of Etymological Data. In: Proceedings of RANLP: The 8th Conference on Recent Advances in Natural Language Processing, Hissar, Bulgaria (2011)
17. Wettig, H., Nouri, J., Reshetnikov, K., Yangarber, R.: Information-theoretic modeling of etymological sound change. In: Approaches to Measuring Linguistic Differences, Mouton de Gruyter (2013)
18. Wettig, H., Reshetnikov, K., Yangarber, R.: Using context and phonetic features in models of etymological sound change. In: Proceedings of EACL Workshop on Visualization of Linguistic Patterns and Uncovering Language History from Multilingual Resources, Avignon, France (2012)
19. Yangarber, R.: Verification of facts across document boundaries. In: Proc. IIIA 2006, Helsinki, Finland (2006)
20. Yangarber, R., Best, C., von Etter, P., Fuart, F., Horby, D., Steinberger, R.: Combining information about epidemic threats from multiple sources. In: Proc. RANLP 2007 MMIES Workshop, Borovets, Bulgaria (2007)
21. Zelenko, D.: Combining MDL transliteration training with discriminative modeling. In: Proceedings of the Named Entities Workshop: Shared Task on Transliteration (2009)
22. Zelenko, D., Aone, C.: Discriminative methods for transliteration. In: Proceedings of EMNLP: Conference on Empirical Methods in Natural Language Processing (2006)
23. Zhang, M., Li, H., Kumaran, A., Liu, M.: Report of news 2012 shared task on machine transliteration. In: Proceedings of NEWS 2012 Named Entities Workshop, vol. 12 (2012)

Using Regression for Spectral Estimation of HMMs

Jordan Rodu[1], Dean P. Foster[1], Weichen Wu[2], and Lyle H. Ungar[2]

[1] University of Pennsylvania, The Wharton School, Department of Statistics
3730 Walnut Street, Philadelphia, PA USA
{jrodu,foster}@wharton.upenn.edu
[2] University of Pennsylvania, Computer and Information Science
200 S. 33rd St, Philadelphia, PA USA
{wewu,ungar}@cis.upenn.edu

Abstract. Hidden Markov Models (HMMs) are widely used to model discrete time series data, but the EM and Gibbs sampling methods used to estimate them are often slow or prone to get stuck in local minima. A more recent class of reduced-dimension spectral methods for estimating HMMs has attractive theoretical properties, but their finite sample size behavior has not been well characterized. We introduce a new spectral model for HMM estimation, a corresponding spectral bilinear regression model, and systematically compare them with a variety of competing simplified models, explaining when and why each method gives superior performance. Using regression to estimate HMMs has a number of advantages, allowing more powerful and flexible modeling.

Keywords: Hidden Markov Models, Spectral Methods.

1 Introduction

Hidden Markov Models (HMMs) [1] are widely used in modelling time series data from text, speech, video and genomic sequences. In applications where the dimension of the observations is much larger than the dimension of the hidden state space, spectral methods can be used project the high dimensional observations down to a much lower dimensional representation that captures the information of the hidden state in the HMM. We call this class of model “spectral HMMs” (*sHMMs*) and show in this paper that sHMMs can be estimated in a variety of ways.

Standard algorithms for HMMs estimate the unobservable transition matrix T and emission matrix O, but are prone to getting stuck in local optima (for instance the EM algorithm) or are computationally intensive (Gibbs sampling). In contrast, sHMM methods estimate a fully observable representation of T and O and are fast, do not have local minima, have nice theoretical error bound proofs, and are optimal in linear estimation sense.

[8] showed that a set of statistics using unigrams, bigrams and trigrams of observations are sufficient to estimate such models. We present a simpler estimation technique and show that it generalizes to a rich collection of regression-based methods for estimating HMMs. In regression, one can easily include more information such as a longer history, or more features about the observed data. These cannot as easily be added into a pure HMM model. Our methods are particularly useful for language modeling, where the emissions of the Hidden Markov Models are words drawn from a large vocabulary (tens or hundreds of thousands of words), and the hidden state is a much lower dimensional representation (30-100 dimensions).

HMMs of this size are widely used in modeling NLP. Many variants of and applications of HMMs have been proposed including (to present a random list of recent work) multiple span-HMM to predict predicates in different domains [9], factorial-HMMs to resolve the pronoun anaphora [11], multi-chain HMMs to compute the meaning of terms in text [18], tree-modified HMMs to do machine translation [19], fertility-HMM to reduce word alignment errors [20] and continuous HMMs to summarize speech documents without text [13].

Our main HMM estimation method, which we call a *spectral HMM* is inspired by the observation in [8] that the 'Observable Operator' model [10] which estimates the probability of a sequence $x_1, x_2, \ldots, x_t$ as

$$Pr(x_1, x_2, \ldots, x_t) = 1^\top A(x_t)A(x_{t-1}) \cdots A(x_1)\pi \tag{1}$$

in terms of the still unobservable $A(x) = T\text{diag}(O^\top x)$ (where $x = e_i$ denotes word i in a vocabulary, and e_i denotes as usual the vector of all zeros and a one in the i^{th} position) and the unigram probabilities π, can be rewritten to be a fully observable, partially reduced model through clever projections and combinations of the moment statistics. [6] extend this to a fully reduced, fully observable model. This extension directly motivates simplified bilinear and regression estimation procedures.

We find that a wide range of spectral methods work well for estimating HMMs. HMMs have an intrinsically bi-linear model, but using a linear approximation works well in practice, especially when one still keeps the use of recursive prediction. Our regression methods are competitive with the "traditional" method of moments methods, and make it relatively easy to add in much richer sets of features than either EM or standard spectral HMM estimations.

The rest of the paper is organized as follows. In section 2 we formally describe the reduced dimension spectral HMM ($sHMM$) model and the bilinear and simplified regression models that it motivates. We also compare our $sHMM$ model against the partially reduced dimension model of [8]. Section 3 gives our experimental results, and discusses prediction accuracy of the different methods in different limits. Section 4 concludes.

2 Approximations to HMMs

Consider a discrete HMM consisting of a sequence of observations $(x_1, x_2, ..., x_t)$ at discrete times $1...t$. Each observation, x_i corresponds to one of n labels (e.g. words). There is a corresponding sequence of hidden states, $(h_1, h_2, ..., h_t)$, where h_i corresponds to one of m labels.

Assume that $m << n$, as is the case, for example, when the vocabulary size n of words is much bigger than the hidden state size. Let T of size $m \times m$ denote the transition matrix; $T_{ij} = Pr(h_t = i | h_{t-1} = j)$. Let O of size $n \times m$ denote the emission matrix; $O_{ij} = Pr(x_t = e_i | h_t = j)$.

We estimate an sHMM using a matrix U which projects each observation x_t onto a low dimensional representation y_t using $y_t = U^\top x_t$, where x_t is defined as before. We work primarily in the y space, which is dimension m instead of the n-dimensional observation space. Note that unlike h, which is a discrete space, y lies in a continuous space.

U is the mapping between the original high dimension observation space and the reduced dimensional representation space. This matrix received a full treatment in [8] and therefore is not the focus of this paper. It is worth noting, however, that U is not unique, and need only satisfy a handful of properties. We call U the *eigenword* matrix, as $y = U^\top x$ forms a low dimensional representation of each word x in the vocabulary. For completeness, we note that a version of U can be easily estimated by taking the largest left singular vectors of the bigram matrix P_{21}, where

$$[P_{21}]_{i,j} = P(x_t = e_i, x_{t+1} = e_j).$$

We use this version in the empirical results presented below. This works well in theory (see details below) and adequately in practice, but better Us can be found, either by estimating U from another much bigger data set, or by using more complex estimation methods [5].

In all of our methods, we will estimate a model to predict the probability of the next item in the sequence given what has been observed so far:

$$\Pr(x_{t+1} | x_t, x_{t-1}, \ldots, x_1) = \Pr(x_{t+1}, x_t, x_{t-1}, \ldots, x_1) / \Pr(x_t, x_{t-1}, \ldots, x_1).$$

We do this in the reduced dimension space of y_i.

2.1 sHMM Model and Estimation

Our core sHMM algorithm estimates $\Pr(x_t, x_{t-1}, \ldots, x_1)$ via the method of moments, writing it in terms of $c_\infty^\top$, c_1 and $\mathcal{C}(y_t)$, and in turn writing each of these three items in terms of moments of the Ys. From [8] and [6] we have

$$Pr(x_1, x_2, \ldots, x_t) = c_\infty^\top \mathcal{C}(y_t) \mathcal{C}(y_{t-1}) \cdots \mathcal{C}(y_1) c_1 \tag{2}$$

with

$$c_1 = \mu, \quad c_\infty^\top = \mu^\top \Sigma^{-1}, \quad \mathcal{C}(y) = \mathcal{K}(y)\Sigma^{-1}$$

and parameters

$$\begin{aligned} \mu &= \mathbb{E}(y_1) = U^\top O\,\pi \\ \Sigma &= \mathbb{E}(y_2 y_1^\top) = U^\top O\,T\,\mathrm{diag}(\pi)\,O^\top U \\ \mathcal{K}(a) &= \mathbb{E}(y_3 y_1^\top y_2^\top)a = U^\top O\,T\,\mathrm{diag}(O^\top U a)\,T\,\mathrm{diag}(\pi)\,(O^\top U) \end{aligned}$$

This yields the following estimate of Pr():

$$\widehat{\Pr}(x_t, x_{t-1}, \ldots, x_1) = \widehat{c}_\infty^\top \widehat{\mathcal{C}}(y_t)\widehat{\mathcal{C}}(y_{t-1}) \cdots \widehat{\mathcal{C}}(y_1)\widehat{c}_1 \tag{3}$$

where

$$\widehat{c}_1 = \widehat{\mu}, \quad \widehat{c}_\infty^\top = \widehat{\mu}^\top \widehat{\Sigma}^{-1}, \quad \widehat{C}_y = \widehat{\mathcal{C}}(y) = \widehat{\mathcal{K}}(y)\widehat{\Sigma}^{-1}$$

and $\widehat{\mu}$, $\widehat{\Sigma}$ and $\widehat{\mathcal{K}}()$ are the empirical estimates of the first, second and third moments of the Y's, namely

$$\widehat{\mu} = \frac{1}{N}\sum_{i=1}^{N} Y_{i,1}, \quad \widehat{\Sigma} = \frac{1}{N}\sum_{i=1}^{N} Y_{i,2}Y_{i,1}^\top, \quad \widehat{\mathcal{K}}(y) = \frac{1}{N}\sum_{i=1}^{N} Y_{i,3}Y_{i,1}^\top Y_{i,2}^\top\, y$$

Here $Y_{i,t}$ indexes the N different independent observations (over i) of our data at time $t \in \{1, 2, 3\}$.

Our HMM model is shown in Figure 1.

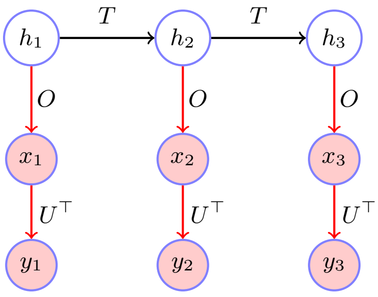


Fig. 1. The HMM with states h_1, h_2, and h_3 which emit observations x_1, x_2, and x_3. These observations are further projected onto the lower dimensional space with observations y_1, y_2, y_3 by U from which our core statistic $\mathcal{C}_y$ is computed based on $\mathcal{K} = E(y_3 y_1^\top y_2^\top)$ which is a $(m \times m \times m)$ tensor.

[6] proved that the *sHMM* model is PAC learnable if the true model is an HMM and the projection matrix U has the property that range$(O) \subset$ range(U) and $|U_{ij}| \leq 1$. Given any small error ϵ and small confidence parameter δ, when the sample triples of observations are bigger than a polynomial of m, n, ϵ and δ, the probability estimated by reduced dimensional tensor $\mathcal{C}(y)$ in Eqn. 3 is smaller than ϵ with high confidence $1 - \delta$.

For any $t \in [2..\infty)$, the estimated value of y_t, denoted by $\hat{y}_t$, can be recursively estimated using the information at the previous time:

$$\hat{y}_t = \frac{\mathcal{C}(y_{t-1})\hat{y}_{t-1}}{\hat{c}_\infty^\top \mathcal{C}(y_{t-1})\hat{y}_{t-1}} \tag{4}$$

with $\hat{y}_1 = \widehat{\mu}$. Since the denominator in Eqn. 4 is a scalar constant for a particular time, we will separate the rescaling step from the recursive computation. Let $\lambda_t = \hat{c}_\infty^\top \mathcal{C}(y_{t-1})\hat{y}_{t-1}$. First we estimate $\tilde{y}_t = \mathcal{C}(y_{t-1})\hat{y}_{t-1}$ using the information from time $t-1$, then we set $\hat{y}_t = \tilde{y}_t / \lambda_t$.[1]

Note that once we have computed $\tilde{y}_t$, λ_t is computed deterministically; hence the key component in estimating $\hat{y}_t$ is the computation of

$$\tilde{y}_t = \mathcal{C}(y_{t-1})\,\hat{y}_{t-1}. \tag{5}$$

The observable HMM representation with $\hat{y}_1$, $\hat{c}_\infty$ and $\mathcal{C}(y)$ is sufficient to predict the probabilities of sequences of observations generated by an HMM. For joint probability of an observation sequence $(x_1, x_2, ..., x_t)$ one can use Eqn. 2. The conditional probability of the same sequence can be computed directly using $\hat{y}_t$. The conditional probability of observing i at time t is

$$Pr[x_t = e_i | x_1, x_2, ..., x_{t-1}] = [U\hat{y}_t]_i \tag{6}$$

This concludes the full presentation of the sHMM model. As mentioned in the introduction, this motivates simpler approximations which will now be discussed.

2.2 Bilinear Regression Model

Our *sHMM* model (5) that outputs the current $\tilde{y}_t$ is bilinear in y_{t-1} and $\hat{y}_{t-1}$. In other words, let $y_{j,t}$ be the j^{th} element of y_t, and $[\mathcal{C}]_{ijk} = c_{ijk}$. Then we can write

$$\tilde{y}_{j,t} = \sum_{i,k} c_{ijk} y_{i,t-1} \hat{y}_{k,t-1} \tag{7}$$

This leads naturally to our first simplified estimation technique–using linear regression by regressing $\tilde{y}_t$ on the outer product of y_{t-1} and $\hat{y}_{t-1}$ as shown in eqn. 7. We call this estimation method *Bilin-RRegr*. "Bilin"since it is Bilinear, "R" for recursive, since it is recursively estimated and predicted using the previous value of $\hat{y}_{k,t-1}$, and "Regr", since it is estimated using regression.

[1] Note the use of $\tilde{y}_t$ for the non-rescaled version of $\hat{y}_t$.

A note on training this model: in order to learn the parameters c_{ijk} we first estimate $\tilde{y}$'s and $\hat{y}$'s using linear regression on our empirically collected trigram data using the actual $y = U^\top x$'s as the "responses" to be predicted. We then estimate the parameters in 7 using a second regression in which these intial estimates of y form the responses. One could iterate this to fixed point, but the above process is in practice sufficient.

Also, although *sHMM* uses the method of moments to estimate the parameters while *Bilin-RRegr* uses linear regression, when used to make predictions the two methods are used identically.

2.3 Other Regression Models

As mentioned in the introduction, many methods can be used to estimate the *sHMM* model. We focus on two main simplifications: one can linearize the bilinear model, and one can drop the recursive estimation. Recursion shows up in two places: when doing estimation, one can regress either on y_t and $\hat{y}_t$ or on y_t and y_{t-1}, and when using the model to predict, one can do a "rolling" prediction, in which y_{t+1} is predicted using the observed y_t and the predicted $\hat{y}_t$. These choices are made independently. For example the base spectral HMM method uses trigrams (no recursion) to estimate, but uses recursion to predict.

The bilinear equation in Eqn 7 can be linearized to give a simpler model to estimate $\tilde{y}_t$ using regression on y_{t-1} and $\hat{y}_{t-1}$. In the experimental results below, we call the resulting recursive linear model *Lin-RReg*:

$$\tilde{y}_t = \alpha\, y_{t-1} + \beta\, \hat{y}_{t-1} \tag{8}$$

We can also further simplify either the recursive bilinear model in Eqn 7 or the recursive linear model of Eqn 8 by noting that a simple linear estimate of $\hat{y}_{t-1}$ is $\hat{y}_{t-1} = Ay_{t-2}$. Since the matrix A is arbitrary, it can be folded into the model, giving a simple linear regression, *Lin-Regr*, model

$$\begin{aligned}\hat{y}_t &= \alpha\, y_{t-1} + \beta_1\, \hat{y}_{t-1}\\ &= \alpha\, y_{t-1} + \beta_2 A\, y_{t-2}\\ &= \alpha\, y_{t-1} + \beta\, y_{t-2}\end{aligned}$$

Note that here we estimate $\hat{y}$ directly instead of first estimating the unscaled $\tilde{y}$ and then rescaling to get our $\hat{y}$. Similarly, one can build a non-recursive bilinear model *Bilin-Regr.*

All of the above estimators work completely in the reduced dimension space Y. They are summarized in Table 1, along the single-lag version of *Lin-Regr*, *Lin-Regr-1*, and a couple of partially reduced dimension models which are described in the following section.

2.4 Partially Reduced Dimension Models

Instead of our fully dimension-reduced model *sHMM*, one can, following [8] estimate a tensor $\mathcal{B}(x)$, which is only projected into the reduced dimension space in two of its three components. $\mathcal{B}(x)$ thus takes an observation x, and produces an $m \times m$ matrix, unlike $\mathcal{C}(y)$ which takes a reduced dimension y and produces an $m \times m$ matrix.[2]

Given $\mathcal{B}(x)$, which is estimated from bigram an trigram occurrence counts, similarly to $\mathcal{C}(y)$, the probability of the next item in a sequence is predicted using the same recursive (rolling) method described above. The fundamental equation is similar in form:

$$Pr(x_1, x_2, \ldots, x_t) = b_\infty^\top \mathcal{B}(x_t)\mathcal{B}(x_{t-1}) \cdots \mathcal{B}(x_1) b_1 \tag{9}$$

See [8] for details. We call this method *HKZ* after its authors.

Our fully reduced dimension *sHMM* offers several advantages over the original *HKZ* method. Working entirely in the reduced dimension space reduces number of parameters to be estimated from $m^2 n$ to m^3. This comes at a cost in that the theorems for *sHMM* require U to contain full range of O instead of only just being full dimension.

The other big change in this paper over [8] is the use of linear regression to estimate the model. Computing a regression, unlike using the method of moments, requires computing the inverse of the covariance of the features (the outer product of y_t and $\hat{y}_t$). At the cost of doing the matrix inversion, we get more accurate estimates, particularly for the rarer emissions.

Using a regression model also gives a tremendous increase in flexibility; The regression can easily include more terms of history, giving more accurate estimates, particularly for more slowly changing or non-Markovian processes. This comes at a cost of estimating more parameters, but if the history is included in linear, instead of a bilinear model, this is relatively cheap.

3 Experiments

In this section, we present experimental results on synthetic and real data for a variety of algorithms for estimating spectral HMMs.

[2] Those familiar with the original paper will note that we have slightly re-interpreted B_x, which Hsu et al. call a matrix, and that what we call x here, they call δ_x. Also the resulting $m \times m$ matrices are identical, specifically

$$\begin{aligned} C(y) &= \mathcal{K}(y)\Sigma^{-1} \\ &= (U^\top O)T \operatorname{diag}(O^\top U y)(U^\top O)^{-1} \\ &= (U^\top O)T \operatorname{diag}(O^\top x)(U^\top O)^{-1} \\ &= \mathcal{B}(x) \end{aligned}$$

Table 1. Methods compared in our experiments. "Num Params" is the number of parameters, not including the $m \times n$ parameters for U. $\hat{y}$ denotes the estimate of y scaled by λ_t as in Eqn. 4, and $\tilde{y}$ denotes the unscaled estimate.

Method	Equation	Num. Params.
sHMM	$\tilde{y}_t = \mathcal{C}(y_{t-1})\hat{y}_{t-1}$	m^3
Bilin-RRegr	$\tilde{y}_t = \mathcal{C}(y_{t-1})\hat{y}_{t-1}$	m^3
Bilin-Regr	$\hat{y}_t = \Gamma(y_{t-1})y_{t-2}$	m^3
Lin-RRegr	$\tilde{y}_t = \alpha\, y_{t-1} + \beta\, \hat{y}_{t-1}$	$2m^2$
Lin-Regr	$\hat{y}_t = \alpha\, y_{t-1} + \beta\, y_{t-2}$	$2m^2$
Lin-Regr-1	$\hat{y}_t = \alpha\, y_{t-1}$	m^2
Lin-Regr-X	$\hat{x}_t = \alpha\, x_{t-1} + \beta\, x_{t-2}$	$2n^2$
HKZ	$\tilde{y}_t = \mathcal{B}(x_{t-1})\hat{y}_{t-1}$	$m^2 n$
EM(BaumWelch)	*MLE*	m^2

Table (1) lists the methods we used in our experiments. The number of parameters being estimated in each case (not including the U projection matrix) are listed on the right side. We expect models with more parameters to better on larger training sets and worse on smaller ones.

3.1 Synthetic Data Test

The synthetic data is generated by constructing HMMs as follows: A potential transition matrix T is generated with normally distributed elements. It is accepted if its second eigenvalue is in the range 0.9 ± 0.1. Similarly, emission matrices O are generated with normally distributed elements and accepted if the second eigenvalue is in 0.8 ± 0.1. This allows us to generate a selection of HMMs, but to control the length of memory of the HMM and the difficulty of estimating it.

We run the experiments as follows. For each of 10 runs, we generate a random HMM model (T, O) as described above and use it to generate a longer observation sequence as training data and 100 short (length 10) sequences as test data.

We then estimate the various models using the training data. First we build the unigram P_1, bigram P_{21} and trigram P_{3x1} of the observations and use them to estimate the projection matrix U and model parameters such as α, β, Γ and $\mathcal{C}$ and $\mathcal{B}$. U consists of the first m singular vectors corresponding to the m largest singular values of P_{21}. For the EM algorithm we use the R package [7]. Finally, we apply every method on each of test sequences and predict the last observation of each test sequence given the preceding observations.

Each method in table (1) was tested varying several properties: training sequence lengths (figure 2a), the dimension of observations (figure 2b), and the state transition probabilities (figure 3). In the last table, the second eigenvalue (2nd EV) of the transition matrix is varied. When this is close to 1, the process mixes slowly. In other words, it behaves close to a deterministic process. When this 2nd eigenvalue is close to zero, the process mixes rapidly. Basically it behaves like a sequence of IID hidden states. Hence more naive estimators will do well.

We report the prediction accuracy averaged over the 10 runs. We count a prediction as correct if the true observation has the highest estimated probability.

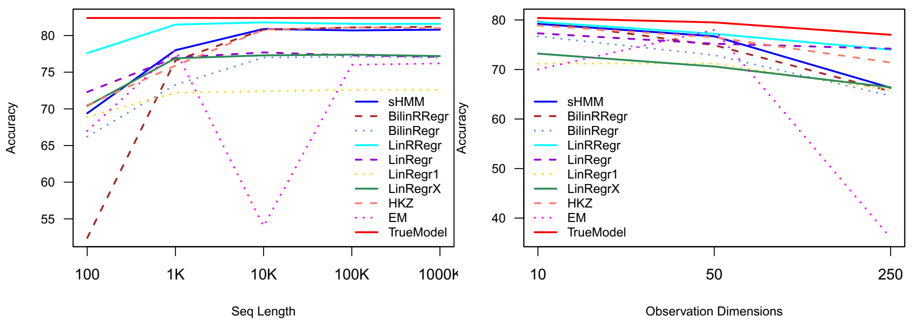


Fig. 2. Prediction accuracy on synthetic data. Number of correct predictions of the 10th observation given the preceding 9 observations on 100 HMM sequences generated with dimension of states $m = 4$, second eigenvalue of transition matrix $T = 0.9$, second eigenvalue of emission matrix $O = 0.8$. Results are the average of 10 runs. The standard errors of 10 runs ranged from .06 to 3.1. **Left: Accuracy as a result of training sequence length. Observation dimension n = 10. Right: Accuracy as a result of observation dimension. Training length 10K**.

3.2 NLP Data Test

We also evaluated our sHMM and rHMM on real NLP data sets. As with the synthetic data experiment, we predict the last word of a test sequence using the preceding words.

We use the New York Times Newswire Service (*nyt-eng*) portion of English Gigaword Fourth Edition corpus (LDC2009T13) in Penn Treebank [14]. We used a vocabulary of ten thousand words, including tokens for punctuation, sentence boundaries, and a single word token for all out-of-vocabulary words. The corpus consisted of approximately 1.3 billion words from 1.8 million document. Our training and test data set are drawn randomly without replacement from the *nyt-eng* corpus. The training data consists of long sequences of observations with lengths varying from 1K to 1000K. The test data consists of 10,000 sequences of observations of length 100.

Following the language modeling literature, we use perplexity to measure how well our estimated language models fit the test data [2,15]. Suppose a predicted distribution of a word x is p and the true distribution is q, the perplexity $PP(x)$ is defined as $PP(x) = 2^{H(p,q)}$, where $H(p,q)$ is the cross-entropy of p and q. i.e. $H(p) = -\sum_x q(x) \log_2 \frac{qx}{p(x)}$. Because our true distribution q is a unit vector with only one element 1 at the x-th dimension, the actual computing of perplexity of word x is simplified as $PP(x) = \frac{1}{p_x}$. A lower perplexity $PP(x)$ indicates a better prediction on x.

We use the same test procedure and methods as for the synthetic data set. The perplexities of language models on *nyt-eng* corpus are shown in figures (4a) and (4b) with vocabularies of 1,000 and 10,000 words.

The results show several main trends, which are illustrated by two-way comparisons

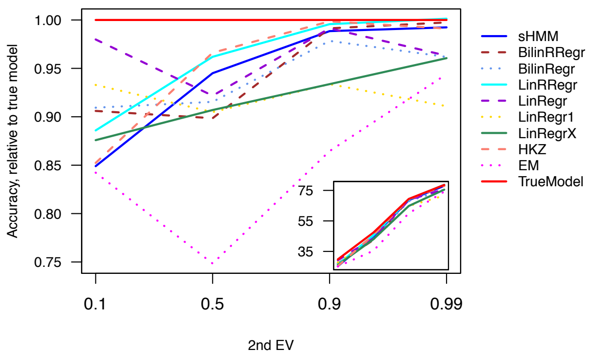


Fig. 3. Prediction accuracy relative to that of True Model on synthetic data in terms of the second eigenvalue of the transition matrix. Inset: actual prediction accuracy. Number of correct predictions of the 10th observation given the preceding 9 observations on 100 HMM sequences. Results are the average of 10 runs. The standard errors of 10 runs ranged from 1.3 to 3.7. The model parameters are the number of states $m = 4$, the number of observations $n = 10$, the training length = 10K, and the second eigenvalue of the emission matrix $O = 0.8$.

- Fully reduced $sHMM$ vs. Partially reduced method HKZ
  - For small training sequences, $sHMM$ is better than HKZ, as one would expect, since $sHMM$ has far fewer parameters to estimate; $\mathcal{C}(y)$ is m/n times smaller than $\mathcal{B}(x)$. As theory predicts, in the limit of infinite training data, the two models are equivalent.
- Fully reduced $sHMM$ vs. Bilinear recursive regression *Bilin-RReg.*
  - On synthetic data generated from an HMM, for smaller training sets $sHMM$ performs better.
- Bilinear regression *Bilin-RReg.* vs. Linear regression *Lin-RReg.*
  - As expected, the simpler model linear model works better with short training sequences (We are not regularizing our regression, and so overfitting is possible). *Lin-RReg* unlike *Bilin-RReg*, is not a correct model of an HMM, and so will not perform as well in the limit of infinite training data.
- Recursive Methods (*Bilin-RReg*, *Lin-RReg*) vs. non-recursive ones (*Bilin-Reg*, *Lin-Reg*)
  - Recursive prediction always helps for linear models. For the more complicated bilinear model, recursion helps if there is sufficient training data. Keeping more lags in the model helps (e.g. *Lin-Regr* vs. *Lin-Regr1*).
- EM method EM
  - The EM method is prone to get stuck in local minima and often gives poor results. One could use a more sophisticated EM method, such as random restarts or annealing methods, but a major advantage of all of the spectral methods presented here is that they are fast (see, for example [4]) and guaranteed to converge to a good solution.

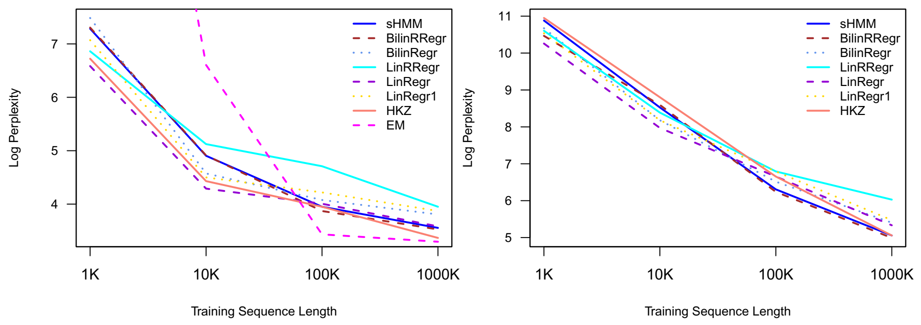


Fig. 4. Log of perplexities of language models on *nyteng* corpus. Left: corpus vocabulary size 1000 words. Right: corpus vocabulary size 10,000 words Note: EM has been excluded on the right in order to preserve a sensible yaxis scale the performance was poor across all sequence lengths.

4 Discussion

HMM's are intrinsically nonlinear, but it is often advantageous to use a linear estimator, even when the data are generated by a model that is not linear. Linear estimators are fast, are guaranteed to converge to a single global optimum, and one can prove strong theorems about them. None of these properties are true of iterative algorithms such as EM for estimating nonlinear models.

We compared two major classes of techniques for estimating HMMs, method of moments ($sHMM$ and HKZ) and regression methods. All the methods presented here inherit the advantage of [8]'s method in that they use the projection matrix U containing the singular vectors of the bigram co-occurrence matrix to reduce the observations from a high dimension observation space X to a low dimension space Y. The Y space captures the information inherent in the hidden states and has same dimension as the hidden states. In this sense, the Y space can be seen as a linear transformation of the hidden state space. One could, of course, do regression in the original observation space, but that leads to models with vastly more parameters, making bilinear models prohibitively expensive. Models in the reduced dimension Y space have far fewer parameters and hence lower computational and sample complexity.

The method of moments models are simple to estimate, requiring only unigram bigram and trigram counts, and *not* requiring any recursive estimation (only recursive prediction). However, using regression models to estimate HMMs allows us far more flexibility than the method of moments models. Simple linear models can be used when training data are limited. Bilinear models that are identical to the $sHMM$ model can be used when more data are available. Longer histories can be used to estimate slowly changing HMMs (e.g. when the second eigenvalue of the transition matrix is close to 1) or when one does not believe that the HMM model is correct. Richer feature sets such as part of speech tags can also be added to the regression models when they are available.

Much work has been done generalizing the (partially reduced) HKZ method [16,17] and extending it and our fully reduced $sHMM$ to probabilistic parsers [3,4,12]. We believe that extensions of the regression-based estimators presented in this paper should prove valuable in these settings as well.

References

1. Baum, L.E., Eagon, J.A.: An inequality with applications to statistical estimation for probabilistic functions of markov processes and to a model for ecology. Bull. Amer. Math. Soc (1967)
2. Brown, P.F., de Souza, P.V., Mercer, R.L., Pietra, V.J.D., Lai, J.C.: Class-based n-gram models of natural language. Computational Linguistics (1992)
3. Cohen, S., Stratos, K., Collins, M., Foster, D., Ungar, L.: Spectral learning of latent-variable pcfgs. In: Association of Computational Linguistics (ACL), vol. 50 (2012)
4. Cohen, S.B., Stratos, K., Collins, M., Foster, D.P., Ungar, L.: Experiments with spectral learning of latent-variable pcfgs. In: NAACL (2013)
5. Dhillon, P., Foster, D., Ungar, L.: Multi-view learning of word embeddings via cca. In: NIPS (2011)
6. Foster, D., Rodu, J., Ungar, L.: Spectral dimensionality reduction for HMMs. ArXiV (2012)
7. Himmelmann, S.S.D.L.: HMM: Hidden Markov Models (2010), `http://CRAN.R-project.org/package=HMM`
8. Hsu, D., Kakade, S.M., Zhang, T.: A spectral algorithm for learning hidden markov models. In: COLT (2009)
9. Huang, F., Yates, A.: Open-domain semantic role labeling by modeling word spans. In: Association of Computational Linguistics (ACL) (2010)
10. Jaeger, H.: Observable operator models for discrete stochastic time series. Neural Computation 12(6) (2000)
11. Li, D., Miller, T., Schuler, W.: A pronoun anaphora resolution system based on factorial hidden markov models. In: Association of Computational Linguistics (ACL) (2011)
12. Luque, F., Quattoni, A., Balle, B., Carreras, X.: Spectral learning for non-deterministic dependency parsing. In: EACL (2012)
13. Maskey, S., Hirschberg, J.: Summarizing speech without text using hidden markov models. In: Association of Computational Linguistics (ACL) (2006)
14. Parker, R., et al.: English gigaword, 4th edn. Linguistic Data Consortium, Philadelphia (2009)
15. Rosenfeld, R.: A maximum entropy approach to adaptive statistical language modeling. Computer Speech and Language 10, 187–228 (1996)
16. Siddiqi, S.M., Boots, B., Gordon, G.J.: Reduced-rank hidden markov models. In: Proc. 13th Intl. Conf. on Artificial Intelligence and Statistics (AISTATS) (2010)
17. Song, L., Boots, B., Siddiqi, S., Gordon, G., Smola, A.: Hilbert space embeddings of hidden markov models. In: Proc. 27th Intl. Conf. on Machine Learning (ICML) (2010)
18. Turdakov, D., Lizorkin, D.: Hmm expanded to multiple interleaved chains as a model for word sense disambiguation. In: PACLIC (2009)
19. Zabokrtsky, Z., Popel, M.: Hidden markov tree model in dependency-based machine translation. In: ACL-IJCNLP (2009)
20. Zhao, S., Gildea, D.: A fast fertility hidden markov model forword alignment using mcmc. In: EMNLP (2010)

Iterative Rule Segmentation under Minimum Description Length for Unsupervised Transduction Grammar Induction

Markus Saers, Karteek Addanki, and Dekai Wu

Human Language Technology Center
Dept. of Computer Science and Engineering
Hong Kong University of Science and Technology, Hong Kong
{masaers,vskaddanki,dekai}@cs.ust.hk

Abstract. We argue that for purely incremental unsupervised learning of phrasal inversion transduction grammars, a minimum description length driven, iterative top-down rule segmentation approach that is the polar opposite of Saers, Addanki, and Wu's previous 2012 bottom-up iterative rule chunking model yields significantly better translation accuracy and grammar parsimony. We still aim for unsupervised bilingual grammar induction such that training and testing are optimized upon the same exact underlying model—a basic principle of machine learning and statistical prediction that has become unduly ignored in statistical machine translation models of late, where most decoders are badly mismatched to the training assumptions. Our novel approach learns phrasal translations by recursively subsegmenting the training corpus, as opposed to our previous model—where we start with a token-based transduction grammar and iteratively build larger chunks. Moreover, the rule segmentation decisions in our approach are driven by a minimum description length objective, whereas the rule chunking decisions were driven by a maximum likelihood objective. We demonstrate empirically how this trades off maximum likelihood against model size, aiming for a more parsimonious grammar that escapes the perfect overfitting to the training data that we start out with, and gradually generalizes to previously unseen sentence translations so long as the model shrinks enough to warrant a looser fit to the training data. Experimental results show that our approach produces a significantly smaller and better model than the chunking-based approach.

1 Introduction

In this paper we argue that significantly better phrasal inversion transduction grammars, or ITGs [1], can be obtained through unsupervised learning with a minimum description length, or MDL, driven, iterative top-down rule segmentation approach, than with the polar opposite: a maximum likelihood driven, iterative bottom-up rule chunking model (as described in our previous work [2]). The size of the search space—all possible combinations of phrasal transduction rules—is so huge that any help we can get in navigating it is welcome; choosing a top-down rather than bottom-up search strategy gives a smaller initial set of rules, and generating new candidates by segmenting (linear in the length of the rule) rather than chunking (quadratic in the size of the rule set) is much more efficient. Given this relationship between the size and number of rules on the one hand, and the search complexity on the other, using minimum description length as learning objective makes a lot of sense: it is designed to explicitly factor in the size of the model, and to keep it parsimonious, as opposed to the maximum likelihood objective, which has no mechanism for rewarding parsimony. We show empirically that the proposed search strategy gives better translation accuracy with a smaller model than our previous model [2].

Our approach also represents a new attack on the problem suffered by most current SMT approaches of learning phrase translations: they require enormous amounts of run-time memory, contain a high degree of redundancy, and fails to provide an obvious basis for generalization to abstract translation schemas. In particular, phrasal SMT models such as [3] and [4] often search for candidate translation segments and transduction rules by committing to a word alignment based on very different modelling assumptions [5, 6], and heuristically derive lexical segment translations [7]. In fact, it is possible to improve the performance by tossing away most of the learned segmental translations [8]. In contrast, we adopt a more "pure" methodology for evaluating transduction grammar induction than typical system building papers. Instead of embedding our learned ITG in the midst of many other heuristic components for the sake of a short term boost in BLEU, we focus on scientifically understanding the behavior of pure MDL-based search for phrasal translations, divorced from the effect of other variables, even though BLEU is naturally much lower this way. The common practice of plugging some aspect of a learned ITG into either (a) a long pipeline of training heuristics and/or (b) an existing decoder that has been patched up to compensate for earlier modeling mistakes—as we and others have done before, see for example [9–20]—obscures the specific traits of the induced grammar. Instead, we directly use our learned ITG in translation mode (any transduction grammar also represents a decoder when parsing with the input sentence as a hard constraint) which allows us to see exactly which aspects of correct translation the transduction rules have captured.

When the structure of an ITG is induced without supervision, we and others have so far assumed that smaller rules should get clumped together into larger rules. This is a natural way to search, since maximum likelihood (ML) tends to improve with longer rules, which is typically balanced with Bayesian priors [10]. Bayesian priors are also used in Gibbs sampling [11, 15], as well as other nonparametric learning methods [19, 20]. All of the above evaluate their models by feeding them into mismatched decoders, making it hard to evaluate how accurate the learned models themselves were. In this work we take a radically different approach, and start with the longest rules possible and attempt to segment them into shorter rules iteratively. This makes ML useless, since our initial model maximizes it. Instead, we balance the ML objective with a minimum description length (MDL) objective, which let us escape the initial ML optimum by rewarding *model parsimony*. The MDL objective is very similar to a Bayesian prior over the structure.

Transduction grammars can also be induced from treebanks instead of unannotated corpora, which cuts down the vast search space by enforcing additional, external constraints. This approach was pioneered by [21], and there has been a lot of research since, usually referred to as **tree-to-tree**, **tree-to-string** and **string-to-tree**, depending on where the analyses are found in the training data. This complicates the learning process by adding external constraints that are bound to match the translation model poorly; grammarians of English should not be expected to care about its relationship to Chinese. It does, however, constitute a way to borrow nonterminal categories that help the translation model.

The MDL objective that we will be using to drive the learning has been used before, but to induce monolingual grammars. [22] uses a method similar to MDL called *Bayesian model merging* to learn the structure of hidden Markov models as well as stochastic context-free grammars (SCFGs). The SCFGs are induced by allowing sequences of nonterminals to be replaced with a single nonterminal (chunking) as well as allowing two nonterminals to merge into one. [23] uses it to learn nonterminal categories in a context-free grammar. It has also been used to interpret visual scenes by classifying the activity that goes on in a video sequences [24]. Our work in this paper is markedly different to even the previous NLP work in that (a) we induce an inversion transduction grammar rather than a monolingual grammar, and (b) we focus on learning the terminal segments rather than the nonterminal categories. We would, of course, like to learn nonterminal categories as well, but will defer that to future work.

We start by taking a closer look at the minimum description length principle (Section 2). Then we describe how the top-down ITG is initialized (Section 3) and generalized (Section 4). After that we outline the empirical experiment (Section 5) and the results (Section 6) before offering some conclusions (Section 7).

2 The Minimum Description Length Principle

The minimum description length principle is about finding the optimal balance between the size of a model and the size of some data given the model [25, 26]. Consider the information theoretical problem of encoding some data with a model, and then sending both the encoded data *and* the information needed to decode the data (the model) over a channel; the minimum description length would be the minimum number of bits sent over the channel. The encoded data can be interpreted as carrying the information necessary to disambiguate the ambiguities or uncertainties that the model has about the data. Theoretically, the model can *grow in size* and become *more certain* about the data, and it can *shrink in size* and become *less certain* about the data. An intuitive interpretation of this is that the exceptions, which are a part of the encoded data, can be moved into the model itself. By doing so, the size of the model increases, but there is no longer an exception that needs to be conveyed about the data. Some exceptions occur frequently enough that it is a good idea to incorporate them into the model, and some do not; finding the optimal balance minimizes the total description length. Formally, the description length (DL) is:

$$\mathrm{DL}\left(\Phi, D\right) = \mathrm{DL}\left(D|\Phi\right) + \mathrm{DL}\left(\Phi\right) \tag{1}$$

where Φ is the model and D is the data. Note the clear parallel to probabilities that have been moved into the logarithmic domain, but keep in mind that lengths do not necessarily have a probabilistic interpretation, whereas probabilities always have a length-in-bits interpretation [27].

In natural language processing, we never have complete data to train on, so we need our models to generalize to unseen data. A model that is very certain about the training data runs the risk of not being able to generalize to new data—we call this over-fitting. It is bad enough when estimating the parameters of a transduction grammar, and catastrophic when inducing the structure of the grammar. The key concept that we want to capture when learning the structure of a transduction grammar is *generalization*. This is the property that allow it to translate new, unseen, input. The challenge is to pin down what generalization actually is, and how to measure it.

One property of generalization for grammars is that it will lower the probability of the training data. This may seem counterintuitive, but can be understood as moving some of the probability mass away from the training data and putting it in unseen data. A second property is that rules that are specific to the training data can be eliminated from the grammar (or replaced with less specific rules that generate the same thing). The second property would shorten the description of the model, and the first would make the description of the data longer. That is: generalization raises the first term and lowers the second term in Equation 1. A good generalization will lower the total MDL, whereas a poor one will raise it.

2.1 Measuring the Length of a Corpus

The information-theoretic view of the problem gives a hint at the operationalization of description length of a corpus given a grammar. [27] stipulates that we can get a lower bound on the number of bits required to encode a specific outcome of a random variable. We thus define description length of the corpus given the grammar as:

$$\mathrm{DL}\left(D|\Phi\right) = -\lg P\left(D|\Phi\right)$$

2.2 Measuring the Length of a Transduction Grammar

Since information theory deals with encoding sequences of symbols, we need some way to serialize an inversion transduction grammar (ITG) into a message whose length can be measures; this section describes how we do this.

To serialize an ITG, we first need to determine the alphabet that the message will be written in. We obviously need one symbol for every nonterminal, L_0-terminal and L_1-terminal. We will also make the assumption that all these symbols are used in at least one rule, so that it is sufficient to serialize the rules in order to express the entire grammar. To serialize the rules, we need some kind of delimiter to know where one rule ends and the next rule begins; we will exploit the fact that we also need to specify whether the rule is straight or inverted (unary rules are assumed to be straight), and merge these two functions into one symbol. What we end up with is the union of the symbols of the grammar and the set $\{[], \langle\rangle\}$, where $[]$ signals the beginning of a straight rule, and $\langle\rangle$ signals the beginning of an inverted rule. The serialized format of a rule will be: rule type/start marker, followed by the left-hand side nonterminal, followed by all right-hand side symbols. The symbols on the right-hand sides are either nonterminals, **biterminals**—pairs of L_0-terminals and L_1-terminals that model translation equivalences. The serialized form of a grammar will be the serialized form of all rules concatenated.

Consider the following toy grammar:

$$S \rightarrow A,\ A \rightarrow \langle AA \rangle,\ A \rightarrow [AA]\,,\ A \rightarrow \text{have/yǒu},\ A \rightarrow \text{yes/yǒu},\ A \rightarrow \text{yes/shì}$$

Its serialized form would be: $[]SA\langle\rangle AAA[]AAA[]A$haveyou$[]A$yesyou$[]A$yessh. Now that we have a message made up of discrete symbols, we can, again turn to information theory to arrive at an encoding for this message. Assuming a uniform distribution over the symbols, each symbol will require $-\lg\left(\frac{1}{N}\right)$ bits to encode (where N is the number of different symbols—the type count). The above example grammar has 8 symbols, meaning that each symbol requires 3 bits; the entire message is 23 symbols long, which means that we need 69 bits to encode it.

3 Initializing the ITG

Rather than starting out with a fairly general transduction grammar and fitting it to the training data, we do the exact opposite: we start with a transduction grammar that fits the training data as well as possible, and generalize from there. The transduction grammar that fits the training data the best is the one where the start symbol rewrites to the full sentence pairs that it has to generate. It is also possible to add any number of nonterminal symbols in the layer between the start symbol and the bisentences without altering the probability of the training data. We take advantage of this by allowing for one intermediate symbol so that the grammar conforms to the normal form and the start symbol always rewrites to precisely one nonterminal symbol. This does violate the minimum description length principle, as the introduction of new symbols, by definition, makes the description of the model longer, but conforming to the normal form of inversion transduction grammars was deemed more important than strictly minimizing the description length. Our initial grammar thus looks like this:

$$S \rightarrow A, \quad A \rightarrow e_{0..T_0}/f_{0..V_0}, \quad A \rightarrow e_{0..T_1}/f_{0..V_1}, \quad \ldots, \quad A \rightarrow e_{0..T_N}/f_{0..V_N}$$

where S is the start symbol, A is the nonterminal, N is the number of sentence pairs in the training corpus, T_i is the length of the i^{th} output sentence (making $e_{0..T_i}$ the i^{th} output sentence), and V_i is the length of the i^{th} input sentence (making $f_{0..V_i}$ the i^{th} input sentence).

4 Generalizing the ITG

To generalize the initial inversion transduction grammar we need to identify parts of the existing biterminals that could be validly used in isolation, and allow them to combine with other segments. This is the very feature that allows a finite transduction grammar to generate an infinite set of sentence pairs; when we do this, we move some of the probability mass which was concentrated in the training data out to other data that are still unseen—the very definition of generalization. The over all strategy is to propose a number of sets of biterminal rules and a place to segment them, evaluate how the description length would change if we were to apply one of these sets of segmentations to the grammar, and commit to the best set. That is: we do a greedy search over the power set of possible segmentations of the rule set. As we will see, this intractable problem can be reasonable efficiently approximated, which is what we have implemented and tested.

The key component in the approach is the ability to evaluate how the description length would change if a specific segmentation was made in the grammar. This can then be extended to a set of segmentations, which only leaves the problem of generating suitable sets of segmentations.

The key to a successful segmentation is to maximize the potential for reuse. Any segment that can be reused saves model size. Consider the terminal rule:

$A \rightarrow$ five thousand yen is my limit/wǒ zùi dūo chū wǔ qīan rì yúan

This rule can be split into three rules:

$A \rightarrow \langle AA \rangle$
$A \rightarrow$ five thousand yen/ wǔ qīan rì yúan
$A \rightarrow$ is my limit/wǒ zùi dūo chū

Note that the original rule consists of 16 symbols (in our encoding scheme), whereas the new tree rules consists of $4 + 9 + 9 = 22$ symbols. Add to that the fact that three rules are likely to be less probable than one rule when parsing, which makes the training data longer as well. It is reasonable to believe that the bracketing inverted rule is present in the grammar already, but this still leaves 18 symbols, which is decidedly longer than 16 symbols—and we need to get the length to be shorter if we want to see a net gain. What we really need to do is find a way to reuse the lexical rules that came out of the segmentation. Now suppose the grammar also contained this terminal rule:

$A \rightarrow$the total fare is five thousand yen/
zǒng gòng de fèi yòng shì wǔ qīan rì yúan

This rule can also be split into three rules:

$A \rightarrow [AA]$
$A \rightarrow$the total fare is/zǒng gòng de fèi yòng shì
$A \rightarrow$five thousand yen/wǔ qīan rì yúan

```
G                   // The ITG
biaffixes_to_rules  // Maps biaffixes to the rules they occur in
do
   biaffixes_delta = []
   for each biaffix b :
      delta = eval_dl(b, biaffixes_to_rules[b], G)
      if (delta < 0)
         biaffixes_delta.push(b, delta)
   sort_by_delta(biaffixes_delta)
   real_delta = 0
   for each b:delta pair in biaffixes_delta :
      real_delta = eval_dl(b, biaffixes_to_rules[b], G)
      if (real_delta < 0)
         G = make_segmentations(b, biaffixes_to_rules[b], G)
while real_delta < 0
```

Fig. 1. Pseudocode for the top-down search algorithm using description length as learning objective

Again, we will assume that the structural rule is already present in the grammar, the old rule was 19 symbols long, and the two new terminal rules are $12+9=21$ symbols long. Again we are out of luck, as the new rules are longer than the old one, and three rules are likely to be less probable than one rule during parsing. The way to make this work is to realize that the two existing rules share a bilingual affix—a **biaffix**: "five thousand dollars" translating into "wǔ qīan rì yúan". If we make the two changes at the same time, we get rid of $16+19=35$ symbols worth of rules, and introduce a mere $9+9+12=30$ symbols worth of rules (assuming the structural rules are already in the grammar). Making these two changes at the same time is essential, as the length of the five saved symbols can be used to offset the likely increase in the length of the corpus given the data. And of course: the more rules we can find with shared biaffixes, the more likely we are to find a good set of segmentations.

Our algorithm takes advantage of the above observation by focusing on the biaffixes found in the training data. Each biaffix defines a set of lexical rules paired up with a possible segmentation. We evaluate the biaffixes by estimating the change in description length associated with committing to all the segmentations defined by a biaffix. This allows us to find the best set of segmentations, but rather than committing only to the one best set of segmentations, we will collect all sets which would improve description length, and try to commit to as many of them as possible. The pseudocode for our algorithm can be found in Figure 1. The pseudocode uses the methods `eval_dl`, `sort_by_delta` and `make_segmentations`. These methods evaluate the difference in description length, sorts candidates by these differences, and commits to a given set of candidates, respectively. To evaluate the description length of a proposed set of candidate segmentations, we need to calculate the difference in description length between the current model, and the model that would result from committing to the candidate segmentations:

$$\mathrm{DL}\left(\Phi', D\right) - \mathrm{DL}\left(\Phi, D\right) = \mathrm{DL}\left(D|\Phi'\right) - \mathrm{DL}\left(D|\Phi\right) + \mathrm{DL}\left(\Phi'\right) - \mathrm{DL}\left(\Phi\right)$$

The model lengths are trivial, as we merely have to encode the rules that are removed and inserted according to our encoding scheme and plug in the summed lengths in the above equation (making sure to add any one rule once only). This leaves the difference in data length, which is:

$$\mathrm{DL}\left(D|\Phi'\right) - \mathrm{DL}\left(D|\Phi\right) = -\mathrm{lg}\,\frac{P\left(D|\Phi'\right)}{P\left(D|\Phi\right)}$$

This lets us determine the probability through biparsing with the grammar being induced. Biparsing is, however, a very expensive operation, and we are making relatively small changes to the grammar, so we will further assume that we can estimate the description length difference in closed form based on the grammar parameters. Given that we are splitting the rule r_0 into the three rules r_1, r_2 and r_3, and that the probability mass of r_0 is distributed uniformly over the new rules, the new grammar parameters θ' will be identical to the old grammar parameters θ, except that:

$$\begin{aligned}
\theta'_{r_0} &= 0 \\
\theta'_{r_1} &= \theta_{r_1} + \frac{1}{3}\theta_{r_0} \\
\theta'_{r_2} &= \theta_{r_2} + \frac{1}{3}\theta_{r_0} \\
\theta'_{r_3} &= \theta_{r_3} + \frac{1}{3}\theta_{r_0}
\end{aligned}$$

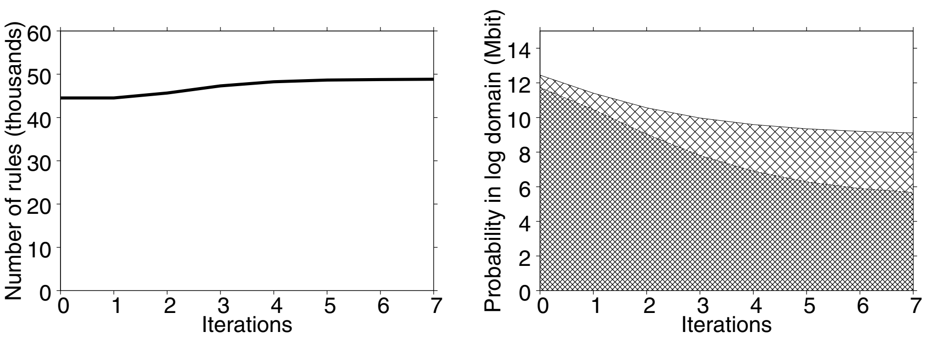
We estimate the probability of the corpus given this new parameters to be:

$$-\mathrm{lg}\,\frac{P\left(D|\Phi'\right)}{P\left(D|\Phi\right)} \approx -\mathrm{lg}\,\frac{\theta'_{r_1}\theta'_{r_2}\theta'_{r_3}}{\theta_{r_0}}$$

To generalize this to a set of rule segmentations, we construct the new parameters θ' to reflect all the changes in the set in a first pass, and then sum the differences in description length for all the rule segmentations with the new parameters in a second pass.

5 Experimental Setup

We have made the claim that iterative top-down segmentation guided by the objective of minimizing the description length is superior to iterative bottom-up chunking as a way of learning stochastic inversion transduction grammars in an unsupervised fashion. We have spent the paper so far outlining how this can be done in practice, and we are now about to show that the outlined method is indeed superior. To substantiate our claim, we will initialize a stochastic bracketing inversion transduction grammar (BITG) to rewrite it's one nonterminal symbol directly into all the sentence pairs of the training data (iteration 0). We will then segment the the grammar iteratively a total of seven times (iteration 1–7), after which the changes are negligible. For each iteration we will record



(a) Rule count (b) Impact of model structure

Fig. 2. Number of rules (a), and the impact of changes in the model structure (b) during the structure induction phase. The change in model structure is broken down into the size of the model (bottom) and the size of the data given the model (top).

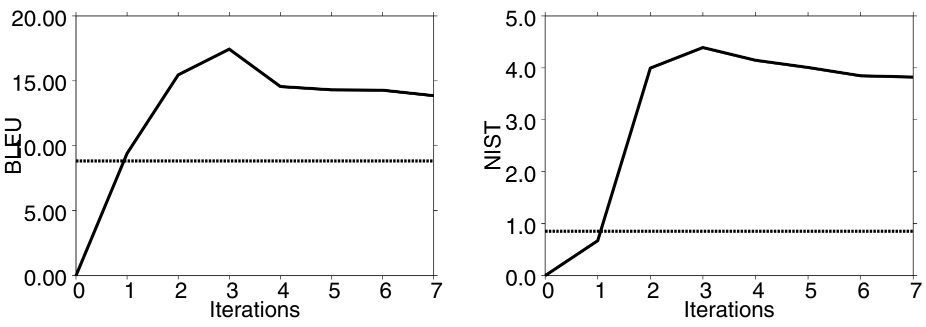
the change in description length and test the learned grammar. Each iteration requires us to biparse the training data to get the data probability of the data given the grammar the iteration starts with. We do this with our in-house implementation of the cubic time algorithm described in [28], with a beam width of 100.

As training data, we use the IWSLT07 Chinese–English data set [29], which contains 46,867 sentence pairs of training data and 489 Chinese sentences with 6 English reference translations each as test data; all the sentences are taken from the traveling domain.

To test the learned grammar as a translation model, we first tune the grammar parameters to the training data using expectation maximization [30] and parse forests acquired with the above mentioned in-house biparser, again with a beam width of 100. To do the actual decoding, we use our in-house ITG decoder. The decoder uses a CKY-style parsing algorithm [31–33] and cube pruning [34] to integrate the language model scores. The decoder builds an efficient hypergraph structure which is then scored using both the induced grammar and the language model. We use SRILM [35] for training a trigram language model on the English side of the training data. To evaluate the quality of the resulting translations, we use BLEU [36], and NIST [37].

6 Results

We claimed that our iterative top-down segmentation guided by the minimum description length objective is superior to iterative bottom-up guided by likelihood for unsupervised induction of inversion transduction grammars; by superior we mean that it produces a smaller model which gives better translation quality, and in the previous section we outlined an experiment to verify this claim.



(a) BLEU scores (b) NIST scores

Fig. 3. Variations in translation quality over different iterations. The dotted line represents the baseline [2].

Figure 2 shows the size of our model during induction, both in terms of rule count and in terms of description length. The initial ITG is at iteration 0, where the vast majority of the size is taken up by the model (DL (Φ), bottom), and very little by the data (DL $(D|\Phi)$, top)—just as we predicted. The trend over the induction phase is a sharp decline in model size, and a moderate increase in data size, with the overall size constantly decreasing. Note that, although the number of rules rises, the total description length decreases. Again, this is precisely what we expected. The size of the model learned by [2] is close to 30 Mbits, and far off the chart.

Figure 3 shows the translation quality of our model as it learns. Here we see a sharp early rise, and then levelling off and even some decline. The main point to focus on is, however, that the second iteration puts us firmly past the results published in [2].

7 Conclusions

We have presented an unsupervised learning method for inversion transduction grammars that iteratively segments the training data in a top-down fashion driven by the objective to minimize description length. This contrasts to our previous work where the learning is conducted bottom-up through chunking towards a maximum likelihood objective. Our experiments show that our new approach is superior.

Acknowledgements. This material is based upon work supported in part by the Defense Advanced Research Projects Agency (DARPA) under BOLT contract no. HR0011-12-C- 0016, and GALE contract nos. HR0011-06-C-0022 and HR0011-06-C-0023; by the European Union under the FP7 grant agreement no. 287658; and by the Hong Kong Research Grants Council (RGC) research grants

GRF620811, FSGRF13EG28, GRF621008, and GRF612806. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA, the EU, or RGC.

References

1. Wu, D.: Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora. Computational Linguistics 23(3), 377–403 (1997)
2. Saers, M., Addanki, K., Wu, D.: From finite-state to inversion transductions: Toward unsupervised bilingual grammar induction. In: COLING 2012: Technical Papers, Mumbai, India, pp. 2325–2340 (December 2012)
3. Koehn, P., Och, F.J., Marcu, D.: Statistical Phrase-Based Translation. In: HLT/NAACL 2003, Edmonton, Canada, vol. 1 (May/June 2003)
4. Chiang, D.: A hierarchical phrase-based model for statistical machine translation. In: ACL 2005, Ann Arbor, Michigan, pp. 263–270 (June 2005)
5. Brown, P.F., Della Pietra, S.A., Della Pietra, V.J., Mercer, R.L.: The Mathematics of Machine Translation: Parameter estimation. CL 19(2) (1993)
6. Vogel, S., Ney, H., Tillmann, C.: HMM-based Word Alignment in Statistical Translation. In: COLING 1996, vol. 2, pp. 836–841 (1996)
7. Och, F.J., Ney, H.: A Systematic Comparison of Various Statistical Alignment Models. Computational Linguistics 29(1), 19–51 (2003)
8. Johnson, H., Martin, J., Foster, G., Kuhn, R.: Improving translation quality by discarding most of the phrasetable. In: EMNLP/CoNLL 2007, Prague, Czech Republic, pp. 967–975 (June 2007)
9. Cherry, C., Lin, D.: Inversion transduction grammar for joint phrasal translation modeling. In: SSST-1, Rochester, New York, pp. 17–24 (April 2007)
10. Zhang, H., Quirk, C., Moore, R.C., Gildea, D.: Bayesian learning of non-compositional phrases with synchronous parsing. In: ACL/HLT 2008, Columbus, Ohio, pp. 97–105 (June 2008)
11. Blunsom, P., Cohn, T., Dyer, C., Osborne, M.: A Gibbs sampler for phrasal synchronous grammar induction. In: ACL/IJCNLP 2009, Singapore (August 2009)
12. Haghighi, A., Blitzer, J., DeNero, J., Klein, D.: Better word alignments with supervised itg models. In: ACL/IJCNLP'09, Suntec, Singapore (August 2009)
13. Saers, M., Wu, D.: Improving phrase-based translation via word alignments from stochastic inversion transduction grammars. In: SSST-3, Boulder, CO (June 2009)
14. Saers, M., Wu, D.: Principled induction of phrasal bilexica. In: EAMT 2011, Leuven, Belgium, pp. 313–320 (May 2011)
15. Blunsom, P., Cohn, T.: Inducing synchronous grammars with slice sampling. In: HLT/NAACL 2010, Los Angeles, California, pp. 238–241 (June 2010)
16. Burkett, D., Blitzer, J., Klein, D.: Joint parsing and alignment with weakly synchronized grammars. In: HLT/NAACL 2010, Los Angeles, CA (June 2010)
17. Riesa, J., Marcu, D.: Hierarchical search for word alignment. In: ACL 2010, Uppsala, Sweden, pp. 157–166 (July 2010)
18. Saers, M., Nivre, J., Wu, D.: Word alignment with stochastic bracketing linear inversion transduction grammar. In: HLT/NAACL 2010, Los Angeles, California, pp. 341–344 (June 2010)
19. Neubig, G., Watanabe, T., Sumita, E., Mori, S., Kawahara, T.: An unsupervised model for joint phrase alignment and extraction. In: ACL/HLT 2011, Portland, Oregon (June 2011)

20. Neubig, G., Watanabe, T., Mori, S., Kawahara, T.: Machine translation without words through substring alignment. In: ACL 2012, Jeju, Korea (July 2012)
21. Galley, M., Graehl, J., Knight, K., Marcu, D., DeNeefe, S., Wang, W., Thayer, I.: Scalable inference and training of context-rich syntactic translation models. In: COLING/ACL 2006, Sydney, Australia (July 2006)
22. Stolcke, A., Omohundro, S.: Inducing probabilistic grammars by bayesian model merging. In: Carrasco, R.C., Oncina, J. (eds.) ICGI 1994. LNCS, vol. 862, pp. 106–118. Springer, Heidelberg (1994)
23. Grünwald, P.: A minimum description length approach to grammar inference in symbolic. In: Wermter, S., Scheler, G., Riloff, E. (eds.) IJCAI-WS 1995. LNCS (LNAI), vol. 1040, pp. 203–216. Springer, Heidelberg (1996)
24. Si, Z., Pei, M., Yao, B., Zhu, S.-C.: Unsupervised learning of event and-or grammar and semantics from video. In: IEEE ICCV 2011 (November 2011)
25. Solomonoff, R.J.: A new method for discovering the grammars of phrase structure languages. In: IFIP Congress, pp. 285–289 (1959)
26. Rissanen, J.: A universal prior for integers and estimation by minimum description length. The Annals of Statistics 11(2), 416–431 (1983)
27. Shannon, C.E.: A mathematical theory of communication. The Bell System Technical Journal 27, 379–423, 623–656 (1948)
28. Saers, M., Nivre, J., Wu, D.: Learning stochastic bracketing inversion transduction grammars with a cubic time biparsing algorithm. In: IWPT 2009, Paris, France, pp. 29–32 (October 2009)
29. Fordyce, C.S.: Overview of the IWSLT 2007 evaluation campaign. In: IWSLT 2007 (2007)
30. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society. Series B (Methodological) 39(1), 1–38 (1977)
31. Cocke, J.: Programming languages and their compilers: Preliminary notes. Courant Institute of Mathematical Sciences, New York University (1969)
32. Kasami, T.: An efficient recognition and syntax analysis algorithm for context-free languages. Technical Report AFCRL-65-00143, Air Force Cambridge Research Laboratory (1965)
33. Younger, D.H.: Recognition and parsing of context-free languages in time n^3. Information and Control 10(2), 189–208 (1967)
34. Chiang, D.: Hierarchical phrase-based translation. CL 33(2) (2007)
35. Stolcke, A.: SRILM – an extensible language modeling toolkit. In: ICSLP 2002, Denver, Colorado, pp. 901–904 (September 2002)
36. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: BLEU: a method for automatic evaluation of machine translation. In: ACL 2002, Philadelphia, PA (July 2002)
37. Doddington, G.: Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In: HLT 2002, San Diego, California, pp. 138–145 (2002)

Comparing Resources for Spanish Lexical Simplification

Horacio Saggion, Stefan Bott, and Luz Rello

University Pompeu Fabra,
Departament of Information and Communication Technologies
C/Tanger 122, 08018 Barcelona, Spain
{stefan.bott,horacio.saggion,luz.rello}@upf.edu

Abstract. In this paper we study the effect of different lexical resources and strategies for selecting synonyms in a lexical simplification system for the Spanish language. The resources used for the experiments are the Spanish EuroWordNet, the Spanish Open Thesaurus and a combination of both. As for the synonym selection strategies, we have used both local and global contexts for word sense disambiguation. We present a novel evaluation framework in lexical simplification that takes into account the level of ambiguity of the word to be simplified. The evaluation compares various instances of the lexical simplification system, a gold standard, and a baseline. On the basis of our results we recommend different resources and word sense disambiguation methods depending on the ambiguity level of the target word to be simplified.

Keywords: Lexical Simplification, Text Simplification, Spanish, Word Sense Disambiguation, Word Vector Model, Lexical Simplification Evaluation.

1 Introduction

Lexical Simplification aims at replacing difficult words with easier synonyms, while preserving the meaning of the original text segments. It is usually considered as an essential part of text simplification, which might target other aspects of textual complexity, such as the syntactic complexity of sentences. Text Simplification can be used as a linguistic preprocess in order to improve other NLP tasks [9, 23], but it can potentially help people with various types of reading comprehension problems [1, 7, 21].

Lexical simplification requires the solution of at least two tasks: First, the finding of a set of synonymic candidates for a given word generally relying on a dictionary and, second, replacing the target word by a synonym which is easier to read and understand in the given context. For the first task different resources are generally used such as WordNet [18]. For the second task, different strategies of word sense disambiguation (WSD) and simplicity computation are requited.

Even if there is a considerable number of approaches to lexical simplification in different languages, an estimation of how different lexical resources and WSD strategies impact the task have not yet been studied. There is also no previous work which addresses the question in how far the level of ambiguity of a word influences the degree of success in automatic lexical simplification. The goal of this paper is to address these gaps using LexSiS [5], a system for Spanish lexical simplification which uses as parameter a lexical resource which provides word senses and lists of synonyms. Hence, the main contributions of this paper are:

– A comparison of the performance of our lexical simplification system with two different lexical resources (Open Thesaurus and EuroWordNet), in addition to a combined version of the two.
– A comparison of two different strategies for word sense disambiguation, one which only considers the local context of a target word and another which assumes that each target word has only one meaning per text and takes all local contexts for a given target into account.
– An evaluation that assesses the performance of the system depending on different levels of the ambiguity of target words.

The rest of the paper is organized as follows: In Section 2 we discuss the related work and the context in which our proposal has to be seen. In Section 3 we describe our system, including the alternative resources it can work with and alternative strategies to perform word sense disambiguation. Section 4 explains the evaluation framework and presents the experimental results while Section 5 draws some conclusions on the use of different lexical resources and disambiguation method. Section 6 concludes the paper with a summary of the main results and an outlook on future work.

2 Related Work

In this paper, we are only interested in lexical simplification as one of the various aspects of text simplification. Lexical simplification requires, at least, two things: a way of finding synonyms (or, in some cases, hyperonyms), and a way of measuring lexical *complexity* (or simplicity). Many approaches to lexical simplification [6, 7, 16] used WordNet in order to find appropriate word substitutions. Bautista *et al.* [3] use a dictionary of synonyms. De Belder *et al.* [11] apply explicit word sense disambiguation, with a Latent Words Language Model, in order to tackle the problem that many of the target words to be substituted are polysemic. As a measure of lexical simplicity most of the cited approaches [6, 7, 16] have relied on word frequency, with the exception of Bautista *et al.* [3], who use word length as a predictor for lexical simplicity. Since both word frequency and word length have been shown to correlate to the cognitive effort in reading [20], Bott *et al.* [5] use a weighted simplicity metric which combines length and frequency.

More recently, the availability of the Simple English Wikipedia (SEW) [10], in combination with the "ordinary" English Wikipedia (EW), made a new generation of text simplification approaches possible, which use primarily machine learning techniques [10, 28–30, 32]. This includes some new approaches to lexical simplification, which are the most important points of reference for our work.

Yatskar *et al.* [31] use edit histories for the SEW and the combination of SEW and EW in order to create a set of lexical substitution rules. Biran *et al.* [4] also rely on the SEW/EW combination (without the edit history of the SEW), in addition to the explicit sentence alignment between SEW and EW. They use WordNet as a filter for possible lexical substitution rules but do not apply explicit word sense disambiguation, their approach is *context-aware*, since they use a cosine-measure of similarity between a lexical item and a given context, in order to filter out possibly harmful rule applications which would select word substitutes with the wrong word sense.

Finally, there is a recent tendency to use statistical machine translation techniques for text simplification (defined as a monolingual machine translation task). Coster and Kauchak,[10] and Specia [24], drawing on work by Caseli *et al.*[8], use standard statistical machine translation machinery for text simplification. In this case lexical simplification is treated as an implicit part of the machine translation problem. The former uses a dataset extracted from the SEW/EW combination, while the latter is noteworthy for two reasons: first, it is one of the few statistical approaches that targets a language different from English (namely Brazilian Portuguese); and second, it is able to achieve good results, although for a limited range of phenomena, with a surprisingly small bi-data-set of only 4,483 sentences.

3 Spanish Lexical Simplification in LexSiS

LexSiS tries to find the best substitution candidate (a word lemma) for every word which has an entry in a lexical resource which is a parameter of the simplification process. The substitution operates in two steps: first the system tries to find the most appropriate sense for a given word, and then it tries to find the best substitution candidate within the list of synonyms of this sense. Here the *best* candidate is defined as the simplest and most appropriate synonym word in the given context. In order to perform word sense dissambiguation we rely on a word vector space model while for the simplicity criterion we apply a combination of word length and word frequency. In the rest of this section we provide the details of the resources and methods used by LexSiS.

3.1 Lexical Resources

As already mentioned, some approaches to lexical simplification make use of WordNet [18] in order to measure the semantic similarity between lexical items and to find an appropriate substitute. Spanish is one of the languages represented in EuroWordNet [27], although its scope is more modest[1]. We have tried three lexical resources in LexSiS: the **Spanish Open Thesaurus** (SOT), the **Spanish**

[1] The Spanish part of EuroWordNet contains only 50,526 word meanings and 23,370 synsets, in comparison to 187,602 meanings and 94,515 synsets in the English WordNet 1.5.

EuroWordNet (SWN), and **combination of SWN and SOT** (SWN+SOT). We describe each of them below.

The Spanish Open Thesaurus lists 21,831 target words (lemmas) and provides a list of word senses for each word. Each word sense is, in turn, a list of substitute words (and we shall refer to them as *substitution sets* hereafter). There is a total of 44,353 such word senses. The substitution candidate words may be contained in more than one of the substitution sets for a target word. The entry in SOT for the word *hoja* is as in (a).

(a) hoja|3
- |acero|espada|puñal|arma blanca
- |bráctea|hojilla|hojuela|bractéola
- |lámina|plancha|placa|tabla|rodaja|película|chapa|lata|viruta|loncha|lonja|capa|...

The first line of the entry represents the target word and states that there are three different meanings. The three lines that follow list synonyms for the three word meanings (*blade*, *leaf* and *sheet* in English).

A second resource we use is the Spanish EuroWordNet. However, for its use with LexSiS we represented synset of SWN in the same format as the Spanish Open Thesaurus, additionally enriching each entry with hyperonymes (e.g. *organo_de_una_planta/plant organ* in the last sense below) of the word. The SWN entry for *hoja* is given in (b)[2].

(b) hoja|4
- |instrumento_cortante
- |folio|cuartilla|pliego|hoja_de_papel|papel
- |folio|folio|cuartilla|pliego|hoja_de_papel
- |follaje|órgano|órgano_de_una_planta|órgano_vegetal

The word *hoja* is also semantically ambiguous here and can mean *blade*, *leaf* or *sheet of paper*. Here the sense for *sheet of paper* is represented by two synsets (second and third lines).

Finally, we are interested in whether a combination of SWN and SOT is able to produce better substitutions since this combination provides more substitution candidates to choose from. For this end we used a union of SWN synsets and SOT substitution sets and let LexSiS choose freely from the alternative lists of synonym words stemming from the two resources. The combined (SOT+SWN) representation for *hoja* contains all the lines contained in (a) and in (b).

3.2 Word Vector Space Model

In order to measure lexical similarity between words and contexts, we used a Word Vector Space Model [22]. Word Vector Space Models are a good way of modelling lexical semantics [26], since they are robust, conceptually simple and

[2] It can be seen in this example that SWN lists many multi-word expressions. At the moment we do not have a module that can detect the same kind of multi-word expressions in the linguistic pre-process, so we have to ignore these entries.

mathematically well defined. The 'meaning' of a word is represented as the contexts in which it can be found. A word vector can be extracted from contexts observed in a corpus, where the dimensions represent the words in the context, and the component values represent their frequencies. The context itself can be defined in different ways, such as an n-word window surrounding the target word. Whether two words are similar in meaning can be measured as the cosine distance between the two corresponding vectors. Moreover, vector models are sensitive to word senses. For example, vectors for word senses can be built as the sum of word vectors which share one meaning.

We trained a vector model on a 8M word corpus of Spanish online news. We lemmatized the corpus with FreeLing [19] and for each lemma type in the corpus we constructed a vector, which represents co-occurring lemmas in a 9-word (actually 9-lemma) window (4 lemmas to the left and to the right). The vector model has n dimensions, where n is the number of lemmas in the lexicon. The dimensions of each vector in the model (i.e. the vector corresponding to a target lemma) represent the lemmas found in the contexts, and the value for each component represents to number of times the corresponding lemma has been found in the 9-word context. In the same process, we also calculated the absolute and relative frequencies of all lemmas observed in this training corpus.

3.3 Word Sense Dissambiguation in LexSiS

We implemented two different methods to carry out word sense disambiguation, which we call the *local* and the *global* method. The local method only looks at the local context of a target word assuming that the local context provides enough information for disambiguation [15], while the global method takes all the occurrences of each target word within a text and constructs a combined representation of the contexts in which they are found, assuming the one sense per discourse hypothesis [14].

For the **local method**, we check for each lemma if it has alternatives in our lexical resource. If this is the case, we extract a vector from the surrounding 9-word window. Since each word is a synonym to itself (and might actually be the simplest word among all alternatives), we include the original word lemma in the list of words that represent the word sense. We construct a common vector for each of the word senses listed in the thesaurus by adding all the vectors of the words listed in each word sense. Then, we select the word sense with the lowest cosine distance to the context vector. In the second step, we select the best candidate within the selected word sense, assigning a simplicity score and applying several thresholds in order to eliminate candidates which are either not much simpler or seem to differ too much from the context.

The **global method** works largely like the local method, with one difference. We assume that each target word has only one meaning in each text it appears. So, instead of extracting a local context vector for each target instance of a word w, we extract all of the local vectors for w found in the text. Then we sum over all of these local vectors, and obtain a global vector for w and compare it to the vectors representing word senses.

3.4 Simplicity Computation and Filtering in LexSiS

As a measure for simplicity we use the metric proposed by Bott et al. [5], which combines word length and word frequency. This metric weights scores for length and frequency and combines them into a single simplicity score. The authors give arguments for the inclusion of word length into the calculus on the basis of a corpus study.

We also apply three thresholds in order to reduce the amount of bad simplification candidates proposed by LexSiS. First of all, we do not want to simplify frequent words, even if our resources (SOT or SWN) list them. So we set a cut-off point for frequent words, such that LexSiS does not try to simplify words with a frequency higher than 0.001% (calculated on the training corpus we used to build the vector model). We also discard substitutes where the difference in the simplicity score with respect to the original word is lower than 0.5, because such words can be expected not to be significantly simpler. We achieved this latter value through experimentation. Since many of the alternatives proposed by LexSiS are not acceptable substitutes, we try to filter out words that do not fit into the context by discarding all candidates whose word vector has a distance with a cosine inferior to 0.013, another value achieved through experimentation. This last threshold is also an attempt to remedy some shortcomings of the lexical resource, especially SOT, which often has entries which are far from being perfect.

4 Evaluation

In this section we present the experimental set-up employed to evaluate the different resourses and word sense disambiguation strategies for LexSiS. The evaluation was conducted thoroughly, rating the degree of simplification and the preservation of meaning of the substitutions.

Baseline: As baseline we use the method of [12]. It replaces a word with its most frequent synonym, presumed to be the simplest. This frequency baseline was also used in SemEval-2012 shared task for lexical simplification [25].

Gold Standard: We have an in-house corpus of parallel texts, consisting of 160 news texts (718 sentences) and manually simplified versions of these text, aligned on the sentence level. As the gold standard we used the manual lexical simplifications we found in this corpus.

Evaluation Dataset: The dataset is composed of the total of lexical simplification substitutions from our gold standard (55), together with the corresponding synonyms generated by LexSiS using the different resources (55 lexical substitutions each), giving a total of 275 lexical substitutions: baseline substitutions (**FREQ**), **SWN** substitutions, **SOT** substitutions, **SWN+SOT** substitutions, **Gold** manual lexical substitutions.

For each of the methods we used two different WSD strategies, having as a result 275 simplifications using a local strategy (**Local WSD**) and 275 simplifications using a global strategy (**Global WSD**).

Since we wanted to evaluate the meaning presentation as well as the simplification, each of the substitutions were inserted in their original sentences. In total we had 550 lexical substitutions to be compared with the original target words. We manually corrected the ungrammatical examples[3] and deleted the duplicated lexical substitutes giving a total of 456 unique lexical substitutions. We believe this is a reasonable size for an evaluation dataset, in [31] they use a total of 200 simplification examples, and in [4] 130 sentences were used. Below, we show two examples of a sentence with its original word (O) and the lexical substitution proposed by our system using SWN+SOT.

(O) Se encuentra a favor de la lucha contra la DESIGUALDAD y la pobreza.
'It is in favor of the fight against inequality and poverty.'
(SWN+SOT) Se encuentra a favor de la lucha contra la IRREGULARIDAD y la pobreza.
'It is in favor of the fight against irregularity and poverty.'

Ambiguity Bands: We divided the target words of the dataset in three levels of difficult depending on their degree of ambiguity. For measuring the degree of ambiguity we considered the average of senses per word given by WordNet and Open Thesaurus. Hence, our dataset has three ambiguity bands, low (from 0.5 to 1.5 senses, 49.08% of the dataset), medium (from 2 to 2.5 senses, 25.09 % of the dataset) and high (3 senses or more, 25.84% of the dataset).

Design: We created a multiple choice questionnaire presenting two sentences for each item. The test included all the unique lexical substitutions. Each item contained one sentence with a simplification example and the same sentence with the original word. These sentences were presented in counterbalanced order to the annotator (i.e., either as Original *vs.* SYSTEM or SYSTEM *vs.* Original). For each pair of sentences, the annotators were asked two questions to choose one option in each of then, one regarding the meaning preservation: "the sentences above have the same meaning" vs. "the sentences above do not have the same meaning", and another one regarding the simplicity degree: "the first of the sentence above is simpler than the second" vs. "the first of the sentence above is not simpler than the second". Five annotators with no previous annotation experience performed the tests using an on-line form. They were all Spanish native speakers, frequent readers and were not the authors of this paper. The five participants annotated all the instances of the datasets, achieving a Fleiss' kappa score of 0.332. Hence, we can assume we have a fair agreement [13, 17], comparable with other inter-annotator agreements in related work, where kappa score was between 0.35 and 0.53 [4].

4.1 Results

Table 1 shows a direct comparison of the performance of LexSiS with different resources and the two different WSD methods, the baseline and the gold standard.

[3] The correction only affected inflections and agreement errors, since we could not use a morphological generator in the experimental setting.

Table 1. Results for the different resources using local and global WSD

Method	**WSD**	**Synonym**	**Simpler**
SWN	Local	**63.2**	68.2
SWN	Global	62.2	**69.2**
SOT	Local	62.0	66.7
SOT	Global	62.0	66.7
SWN + SOT	Local	58.6	66.4
SWN + SOT	Global	60.6	68.2
Baseline	-	52.6	**71.1**
Gold	-	**75.9**	69.3

Table 2. Results for meaning preservation for different ambiguity levels

Ambig.		Meaning preservation			Simpler synonyms		
Band	**WSD**	**SWN**	**SOT**	**SOT+SWN**	**SWN**	**SOT**	**SOT+SWN**
Low	Local	60.0	62.5	56.0	70.0	**70.0**	68.7
Low	Global	61.0	62.5	56.9	70.3	**70.0**	68.9
Med.	Local	65.5	51.1	56.9	72.2	60.9	70.3
Med.	Global	61.7	51.1	63.1	**73.0**	60.9	**70.8**
High	Local	**67.4**	**70.4**	**66.7**	60.6	65.8	58.3
High	Global	65.3	**70.4**	66.1	62.5	65.8	64.1

In Table 2 we show the results for WSD and simplicity by ambiguity level. The scores for simplicity were calculated over those data points which were judged as being synonymous in order to be able to achieve independent scores for synonymity and simplicity. The WSD methods of *local* and *global* correspond to the two ways of constructing context vectors described in Section 3.

5 Discussion

We observe (Table 1) that LexSiS shows consistently much higher synonymity scores than the baseline. For the whole dataset, without distinction of levels of ambiguity LexSiS with SWN achieves a score of 63.16% in comparison to 52.59% produced by the baseline. When LexSiS uses other resources (SOT or SWN+SOT) the scores are only slightly lower. The score for the gold standard is higher (75.92%), but surprisingly it does not even get close to 100%, which shows that human judges are reluctant to accept alternatives as being fully synonymous and gives an idea of the difficulty of the task. A surprise is that the combination of the two resources (SWN+SOT) performs much worse than the two resources on their own. We hoped that the word disambiguation component would perform better with the availability of more synonym sets, because the cosine distance between the vectors for these sets should lead to a selection of the set with the most coherent meaning, penalising sets which include words that are not coherent with the rest of the set. This expectation was not met. A possible alternative would be to align the resources so that equivalent synsets are merged together providing additional synonym choices for equivalent senses. We will investigate this in future work.

Turning to the question whether the use of SOT shows a worse performance than that of SWN, we can observe a slightly better performance of LexSiS+SWN than LexSiS+SOT in both categories (Table 1), but none of the differences is statistically significant. This is an interesting result, because it suggests that a simpler resource like SOT can lead to nearly the same level of performance than the use of a more sophisticated resource like SWN, whose quality is controlled in a much stricter way. In SOT we often observed incoherent synonyms sets, cases in which more than one synonym sets appear to represent the same word sense and word senses which are not represented by a single synonym set. Nevertheless, we think that the use of the word sense disambiguation component in combination with the threshold that filters out candidates with a high cosine distance to the context can partially remedy the shortcomings of the thesaurus.

The left part of Table 2 (meaning preservation) suggests that words with a higher degree of ambiguity are easier to disambiguate, which might come as another surprise. We suspect that this reveals a possible shortcoming of the resources used: words which are listed with only one word sense are often still ambiguous, while the entries of words with many listed senses tend to be disambiguated better in the dictionary entry.

Turning to the production of synonyms which are perceived as being actually simpler than the original, again SWN outperforms SOT and SWN+SOT (cf. Table 1). In right part of Table 2 (simpler synonyms) we can observe that the success of producing simpler synonyms depends very much on the level of ambiguity of the target word. Highly ambiguous words are harder to simplify, while words with low ambiguity are easier. Words with a medium level ambiguity show a curious behaviour: for SWN and the combination of SWN and SOT these words appear to be easier to simplify.

Table 2 also shows that the global method of choosing synonyms (one synonym for each target per text) systematically outperforms the local method in its ability to produce simpler substitutes. We attribute this to the fact that the summed vectors for target word contexts are present much richer context information and are much more reliable than the rather sparse vectors for individual contexts. The assumption that each target word has only one meaning per text proves to be quite helpful. For example, in the following sentence the original word *noción* (*notion*) has been substituted with the word *representacíon*(*representation*) with the local method and the more acceptable word *idea* with the global method, using SWN+SOT.

(O) ...vivimos en un mundo en el que se ha perdido la NOCIÓN de autoridad.
'...we live in a world where the NOTION of authority has been lost'

In Table 1 the baseline outperforms LexSiS and even the gold standard in the simplification task, but it has to be taken into account that the simplicity scores were calculated only over those data instances that were actually perceived by the annotators as being synonymous. This amounts to saying that the frequency baseline would perform extraordinarily if all the non-synonym productions were filtered out, which is first impossible and would result in a much lower coverage (i.e. much less substitutions produced) than LexSiS. As a curious matter of fact, only 51.85% of the gold standard cases were both judged as being synonymous and being simpler, which illustrates the difficultly of the combined task.

Turning to significance, we only found a significant effect between different methods on the meaning preservation; the gold standard preserved significantly more meaning in their substitutions that the rest of the methods, $(F(9, 2262) = 4.062, p < 0.01)$. This finding is hardly surprising, given the difficulty of the word sense disambiguation task. It is probably more interesting to note that, while the gold standard achieves higher scores for simplicity, this score is not much higher than the score for LexSiS with the use of SWN. Also, even if the scores for LexSiS with different configurations are lower, the difference to the gold standard could not be shown to be statistically significant.

6 Conclusions and Outlook

In this paper we compared the effect of using different lexical resources and disambiguation strategies in a lexical simplification system for the Spanish language. In particular we have instantiated experiments with the Spanish WordNet and the Spanish Open Thesaurus as lexical resources. Where disambiguation methods are concerned, we have tried local and global disambiguation strategies.

The comparison of two different lexical resources shows how far the quality of the resource used influences the quality of the lexical simplifications the system produces. Since Open Thesaurus is an open collaborative effort, the quality of the thesaurus entries is not strongly controlled, a factor which we could see reflected in poorly separated word senses and even missing representation for some senses. We could find differences in the performance depending on the lexical resource used, but it was surprisingly low and not statistically significant. This is a good result because the main bottleneck for the most language dependent part of a lexical simplification system like LexSiS is the availability of lexical resources. Our evaluation suggests that thesauri may be a good substitute for more sophisticated lexical ontologies.

Another contribution of this paper is the comparison of two WSD methods: one based on local context and the second global method based on summed local context on the text level. We could show that the global method performs better for the lexical substitution task. The choice of the lexical resource is only one of a list of possible optimizations for the LexSiS system. There are other possibilities we would like to explore in the future, such as the use of TF*IDF weights and the investigation of in how far the size of the window which represents the context influences the system performance. Our lexical simplification system could also help to normalize paraphrases to the simplest word choice, which could be useful in plagiaism detection [2].

Acknowledgments. The research described in this paper is partially funded by Ministerio de Industria, Turismo, y Comercio, Spain (Avanza Competitiveness, with file number TSI-020302-2010-84) and by fellowship RYC-2009-04291 from Programa Ramón y Cajal 2009 and project number TIN2012-38584-C06-03 from Ministerio de Economía y Competitividad, Secretaría de Estado de Investigación, Desarrollo e Innovacíon, Spain.

References

1. Aluísio, S.M., Gasperin, C.: Fostering digital inclusion and accessibility: the PorSimples project for simplification of Portuguese texts. In: NAACL/HLT, Young Investigators Workshop on Computational Approaches to Languages of the Americas, YIWCALA 2010, pp. 46–53. ACL, Stroudsburg (2010), `http://dl.acm.org/citation.cfm?id=1868701.1868708`
2. Barrón-Cedeño, A., Vila, M., Martí, M., Rosso, P.: Plagiarism meets paraphrasing: Insights for the next generation in automatic plagiarism detection. Computational Linguistics 39(4) (2013)
3. Bautista, S., León, C., Hervás, R., Gervás, P.: Empirical identification of text simplification strategies for reading-impaired people. In: European Conference for the Advancement of Assistive Technology (2011)
4. Biran, O., Brody, S., Elhadad, N.: Putting it simply: A context-aware approach to lexical simplificaion. In: ACL/HLT, pp. 496–501. ACL, Portland (2011), `http://www.aclweb.org/anthology/P11-2087`
5. Bott, S., Rello, L., Drndarević, B., Saggion, H.: Can Spanish Be Simpler? LexSiS: Lexical Simplification for Spanish. In: CoLing, December 8-16 (2012)
6. Burstein, J., Shore, J., Sabatini, J., Lee, Y.W., Ventura, M.: The automated text adaptation tool. In: NAACL/HLT (Demonstrations), pp. 3–4 (2007), `http://dblp.uni-trier.de/db/conf/naacl/naacl2007.html#BursteinSSLV07`
7. Carroll, J., Minnen, G., Canning, Y., Devlin, S., Tait, J.: Practical Simplification of English Newspaper Text to Assist Aphasic Readers. In: Proc. of AAAI 1998 Workshop on Integrating Artificial Intelligence and Assistive Technology, pp. 7–10 (1998)
8. Caseli, H.M., Pereira, T.F., Specia, L., Pardo, T.A.S., Gasperin, C., Aluísio, S.M.: Building a Brazilian Portuguese parallel corpus of original and simplified texts. In: CICLing (2009)
9. Chandrasekar, R., Doran, D., Srinivas, B.: Motivations and methods for text simplification. In: CoLing, vol. 2, pp. 1041–1044 (1996)
10. Coster, W., Kauchak, D.: Learning to simplify sentences using Wikipedia. In: Proceedings of the Workshop on Monolingual Text-To-Text Generation. ACL (2011)
11. De Belder, J., Deschacht, K., Moens, M.F.: Lexical simplification. In: Proceedings of Itec 2010: 1st International Conference on Interdisciplinary Research on Technology, Education and Communication (2010), `https://lirias.kuleuven.be/handle/123456789/268437`
12. Devlin, S., Unthank, G.: Helping aphasic people process online information. In: The International ACM SIGACCESS Conference on Computers and Accessibility, pp. 225–226 (2006)
13. Fleiss, J.L.: Measuring nominal scale agreement among many raters. Psychological Bulletin 76(5), 378–382 (1971)

14. Gale, W.A., Church, K.W., Yarowsky, D.: One sense per discourse. In: Proceedings of the Workshop on Speech and Natural Language, pp. 233–237. HLT (1992)
15. Krovetz, R.: More than one sense per discourse. In: NEC Princeton NJ Labs. Research Memorandum (1998)
16. Lal, P., Ruger, S.: Extract-based summarization with simplification. In: Proceedings of the ACL 2002 Automatic Summarization/DUC 2002 Workshop (2002)
17. Landis, J., Koch, G.: The measurement of observer agreement for categorical data. Biometrics, 159–174 (1977)
18. Miller, G., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.: Introduction to WordNet: An on-line lexical database. International Journal of Lexicography 3(4), 235–244 (1990)
19. Padró, L., Collado, M., Reese, S., Lloberes, M., Castellón, I.: FreeLing 2.1: Five years of open-source language processing tools. In: LREC, Valletta, Malta (2010)
20. Rayner, K., Duffy, S.A.: Lexical complexity and fixation times in reading: Effects of word frequency, verb complexity, and lexical ambiguity. Memory & Cognition 14(3), 191–201 (1986)
21. Rello, L., Baeza-Yates, R., Dempere, L., Saggion, H.: Frequent words improve readability and short words improve understandability for people with dyslexia. In: INTERACT 2013, Cape Town, South Africa (2013)
22. Salton, G., Wong, A., Yang, C.: A vector space model for automatic indexing. Communications of the ACM 18(11), 613–620 (1975)
23. Siddharthan, A.: An architecture for a text simplification system. In: LREC 2002: Proceedings of the Language Engineering Conference, LEC 2002, pp. 64–71 (2002)
24. Specia, L.: Translating from complex to simplified sentences. In: Pardo, T.A.S., Branco, A., Klautau, A., Vieira, R., de Lima, V.L.S. (eds.) PROPOR 2010. LNCS, vol. 6001, pp. 30–39. Springer, Heidelberg (2010), http://dx.doi.org/10.1007/978-3-642-12320-7_5
25. Specia, L., Jauhar, S., Mihalcea, R.: Semeval-2012 task 1: English lexical simplification. In: SemEval 2012 (2012)
26. Turney, P., Pantel, P.: From frequency to meaning: Vector space models of semantics. Journal of Artificial Intelligence Research 37(1), 141–188 (2010)
27. Vossen, P. (ed.): EuroWordNet: A Multilingual Database with Lexical Semantic Networks, vol. 17. Oxford Univ. Press (2004)
28. Woodsend, K., Feng, Y., Lapata, M.: Generation with quasi-synchronous grammar. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pp. 513–523. ACL (2010)
29. Woodsend, K., Lapata, M.: WikiSimple: Automatic simplification of wikipedia articles. In: Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence (AAAI), pp. 927–932 (2011)
30. Wubben, S., van den Bosch, A., Krahmer, E.: Sentence simplification by monolingual machine translation. In: ACL (2012)
31. Yatskar, M., Pang, B., Danescu-Niculescu-Mizil, C., Lee, L.: For the sake of simplicity: Unsupervised extraction of lexical simplifications from wikipedia. In: ACL, pp. 365–368 (2010)
32. Zhu, Z., Bernhard, D., Gurevych, I.: A monolingual tree-based translation model for sentence simplification. In: CoLing, pp. 1353–1361 (2010)

Context-Aware Correction of Spelling Errors in Hungarian Medical Documents

Borbála Siklósi[2], Attila Novák[1,2], and Gábor Prószéky[1,2]

[1] MTA-PPKE Language Technology Research Group,
[2] Pázmány Péter Catholic University, Faculty of Information Technology,
50/a Práter street, 1083 Budapest, Hungary
{siklosi.borbala,novak.attila,proszeky.gabor}@itk.ppke.hu

Abstract. In our paper, we present a method for automated correction of spelling errors in Hungarian clinical records. We model the problem of spelling correction as a translation task, where the source language is the erroneous text and the target language is the corrected one using an SMT decoder to perform the error correction. Since no orthographically correct proofread text from this domain is available, we cannot use such a corpus for training the system, instead a spelling correction generation and ranking system is used to create translation models. In addition, a language model is used in order to model lexical context. We show that our system outperforms the first candidate accuracy of the baseline ranking system.

Keywords: spelling correction, agglutinating languages, medical text processing.

1 Introduction

Processing medical texts is an emerging topic in natural language processing. There are existing solutions mainly in English to extract knowledge from medical documents, which thus becomes available for researchers and medical experts. However, locally relevant characteristics of applied medical protocols or information relevant to locally prevailing epidemic data can be extracted only from documents written in the language of the local community.

In Hungarian hospitals, clinical records are created as unstructured texts, without any automated proofing control (e.g. spell checking). Moreover, the language of these documents contains a high ratio of word forms not commonly used, such as Latin medical terminology, abbreviations and drug names. Many of the authors of these texts are not aware of the standard orthography of this terminology. Thus processing such documents is not an easy task and automatic correction of the documents is a prerequisite of any further linguistic processing.

We investigated anonymized clinical records of a Hungarian clinic, in which we found errors due to the frequent (and apparently intentional) use of non-standard orthography, unintentional mistyping, inconsistent word usage and ambiguous misspellings (e.g. misspelled abbreviations), some of which are very hard to interpret and correct even for a medical expert. Besides, there is a high number of real-word errors, i.e. otherwise correct word forms, which are incorrect in the actual context. Many misspelled words never or hardly ever occur in their orthographically standard form in our corpus of clinical records. One possible solution for this problem is creating and using a quasi-standard representation (i.e. each concept represented by the same string for all occurrences) even if that representation does not correspond to the academic standard.

In our paper, we present a method, built on a baseline system for generating correction candidates, for considering textual context when recognizing and correcting spelling errors. We show that our system is able to correct certain errors with high accuracy, and, due to its parametrization, it can be tuned to the actual task. Thus the presented method is able to automatically correct single errors in words, making a firm base for extending it to the correction of multiple errors as well, and creating a normalized version of the clinical records corpus in order to apply higher level processing.

2 Spelling Errors

A characteristic of clinical documents is that they are usually created in a rush without proofreading. The medical records creation and archival tools used at most Hungarian hospitals provide no proofing or structuring tools. Thus the number of spelling errors is very high and a wide variety of error types occur. These errors are not only due to the complexity of the Hungarian language and orthography, but also to characteristics typical of the medical domain and the situation in which the documents are created. The most frequent types of errors are the following:

- mistyping, accidentally swapping letters, inserting extra letters or just missing some,
- lack or improper use of punctuation (e.g. no sign of sentence boundaries, missing commas, no space between punctuation and the neighboring words),
- grammatical errors,
- sentence fragments,
- domain-specific and often ad hoc abbreviations, which usually do not correspond to any standard
- Latin medical terminology not conforming to orthographical standards.

A common characteristic of these phenomena is that the prevailing errors vary with the doctor or assistant typing the text. Thus it can occur that a certain word is mistyped and should be corrected in one document while the same word is a specific abbreviation in another one, which does not correspond to the same concept as the corrected one. Latin medical terms usually have a standard form based on both Latin and Hungarian orthography, however what we find in the documents is often an inconsistent mixture of the two (e.g. tensio/tenzio/tensió/tenzió). Even though the spelling of these forms is standardized, doctors tend to develop their own customs which they use inconsistently.

Another difficulty is the complete lack of correctly written clinical documents that could be used for creating the appropriate language and error models.

3 Related Work

Much research has been done on spelling correction. Kukich [9] partitions the problem to three subproblems as (a) non-word error detection; (b) isolated-word error correction; and (c) context-dependent word correction. However, most of the described techniques rely on a lexicon-based approach that is not applicable to agglutinating languages such as Hungarian. The problems of spelling correction in agglutinative languages is described in [12]. One way of handling an infinite vocabulary is applying finite state automata or transducers, which are used in [14], [10] and [16].

In our work, we aim at solving all of the three problems in one step, that is, recognizing and correcting misspellings in context. A current trend is to apply statistical or hybrid approaches that outperform the previously prevailing rule-based methods. A widespread solution is to apply the noisy-channel model. The systems described in [5] and [3] apply variants of this model using different error models and probability scoring. The work described in [2] emphasizes the beneficial use of a contextual language model in the case of spelling correction while adopting the noisy-channel model. Another work applies a graph-based approach in a very strict domain-specific solution described in [1]. In [20], misspelled words are identified by comparing them to some predefined list of words, but this baseline method is extended by doing prevalence analysis, i.e. determining the frequency ratio of a word and its one edit distance alternatives in the corpus.

The problem of spelling correction in the clinical domain has been addressed in a number of publications. A research published in [15] uses several knowledge bases of English clinical terms besides applying statistical methods. In [6], a solution is implemented for rescoring the ranked candidates of different correction suggestion methods.

Although the idea of the noisy-channel model, which is explored in several works, is the basis of statistical machine translation algorithms as well, only very few works use SMT implementations directly. Mass noun errors in English as a Second Language texts are corrected in [4], which is a grammatical rather than an orthographic problem. The work most similar to our approach is that described in [7], where the traditional SMT algorithm is applied to the problem of spelling error correction. However, in that implementation, the translation model is based on a parallel corpus of proofread and erroneous texts into which errors were introduced artificially. One problem with this approach is that the random errors introduced into the corpus might not model well the types of errors people actually introduce. Another problem from our perspective is that one needs a correctly written corpus in the first place, which we do not have.

Regarding related literature, none of the solutions that use a predefined lexicon are applicable in our case due to the problems caused by an agglutinating language.

4 Application of a Statistical Machine Translation System for Spelling Correction

Our goal was to improve a baseline system presented in [18] that is able to generate correction suggestions for misspelled words considering them as isolated single words ignoring their context. The ranking in this baseline system is based on statistics built from domain-specific and general corpora in addition to grammaticality judgment of a wide coverage Hungarian morphological analyzer [11], [17]. The system is parametrized to assign much weight to frequency data coming from the domain-specific corpus, which ensures not coercing medical terminology into word forms frequent in general out-of-domain text. The baseline system was able to recognize most spelling errors and the list of the ten highest ranked automatically generated corrections contained the actually correct one in 98% of the corrections in the test set.

Since our goal is to create fully automatic correction, rather than offering the user a set of corrections that they can choose from, the system should be able to automatically find the most appropriate correction. In order to achieve this goal, the ranking of the baseline system based on morphology and word frequency data is not enough. To improve the accuracy of the system, lexical context also needs to be considered. To satisfy these two requirements, we applied Moses [8], a widely used statistical machine translation (SMT) toolkit. During "translation", we consider the original erroneous text as the source language, while the target is its corrected, normalized version. In this case, the input of the system is the erroneous sentence: $E = e_1, e_2 \dots e_k$, and the corresponding correct sentence $C = c_1, c_2 \dots c_k$ is the expected output. Applying the noisy-channel model terminology to our spelling correction system: the original message is the correct sentence and the noisy signal received at the end of the channel data is the corresponding sentence containing spelling errors. The output of the system trying to decode the noisy signal is the sentence $\hat{C}$, where

$$\hat{C} = argmaxP(C|E) = argmax\frac{P(E|C)P(C)}{P(E)} \tag{1}$$

conditional probability takes its maximal value. Since $P(H)$ is constant, the denominator can be ignored, thus the product in the numerator can be derived from the statistical translation and language models.

These models in a traditional SMT task are built from a parallel corpus of the source and target languages based on the probabilities of phrases corresponding to each other. The language model responsible for checking how well each candidate generated by the translation model fits the actual context is built using the SRILM toolkit [19].

4.1 Translation Models

In our system, we applied three translation (correction) models according to three categories of words and errors. The first one handles possible abbreviations, the second one can split erroneously joined words, and the third one handles all other errors. In the following sections, we describe each of these models starting with the last one.

Translation Model for General Words and Errors. The translation model is based on the output of the baseline system described in [18]. For each word, except for abbreviations and some stopwords, we considered the first 20 suggestions. The original suggestion system is also extended in a way that it can also generate suggestions by splitting words. Considering more than 20 candidates would have caused noise rather than increasing the quality of the system. The scores used for ranking these suggestions are normalized as a quasi-probability distribution, so that the sum of the probabilities of all possible corrections for a word is 1. We applied this method instead of learning these probabilities from a parallel corpus as no such corpus is available. It should be noted, that though suggestions are generated for each word in the sentences, these suggestions usually include the original form. The scoring ensures that if the original form was correct, then it will receive a high score, thus the decoder will not modify the word.

Table 1 contains a common word that is misspelled in the input text. The word *hosszúságu* should be written as *hosszúságú* 'of length ...'. Another word form, *hosszúsági* 'longitudinal' is ranked higher by the original context-insensitive scoring algorithm, since it is also a correct and more frequent Hungarian word, and since the *u:i* correspondence is also a frequent error beside *u:ú* since *u* and *i* are neighboring letters on the keyboard. Though the rest of the words in the example are also correct candidates, they received a lower score.

Without considering the context, all the others would also be correct at the word level. Our language model will be responsible for making the contextually optimal choice.

Table 1. A fragment of the translation model for a misspelled common word, its possible candidate corrections and their probabilities

hosszúságu	hosszúsági	0.01649
hosszúságu	hosszúságú	0.01560
hosszúságu	hosszúsága	0.01353
hosszúságu	hosszúságuk	0.01317
hosszúságu	hosszúságul	0.01292
hosszúságu	hosszúságé	0.01284
hosszúságu	hosszúság	0.01034

Translation Model for Abbreviations. Clinical documents contain much more abbreviations than general texts. (The ratio of abbreviations is 8.49% in our sample of clinical documents, while 0.36% in general texts.) Applying the above models to abbreviations is difficult due to two main reasons. On the one hand, the same word or phrase usually appears in several different abbreviated forms in the text according to the taste of the author or just due to accidental variation. On the other hand, most abbreviations are very short, and, in the case of most of them, the suggestion generator would prefer to transform the original abbreviation to a very frequent similar common word. Due to their high frequency and the fact that the morphology would also affirm their correctness, such "corrections" would practically ruin the semantics of the original text.

To eliminate these problems, we collected the possible abbreviations from the clinical corpus with automatic methods and after manually filtering this list, integrated some of these into the morphology. However, this alone does not solve the problem, since the same abbreviation might have several variations in the texts. Thus we collected possible variations of each potential abbreviation from the corpus together with their frequencies and used these values as maximum likelihood estimates. An alternative translation model was created this way. These abbreviations are not present in the first translation model in order to prevent the system transforming them to other words. We then applied the decoder of the SMT system so that the translation model of the abbreviations is given priority ensuring that abbreviations are transformed to their correct form rather than to other words. Table 2 shows some combinations of abbreviation corrections in the translation model.

Table 2. A portion of the translation model for abbreviations

conj.	conj.	0.6078
conj	conj.	0.8696
conj	conj	0.1303
mko	mko	0.4891
mko	mko.	0.9970
mko.	mko.	0.9993

Handling Joining Errors. Since the Moses SMT toolkit is usually used as a phrase-based translation tool in traditional translation tasks, a general feature of the translation models is that the translation of one (or more) words can also be more than one word. Thus the system can be used to generate multi-word suggestions for a single word in a straightforward manner, this way our system can split erroneously joined words. Probability estimates for these phrases are also derived from the scores assigned by the suggestion generation system. When inserting a space into a word, the models used for creating the ranking scores are calculated for both words separately and the geometric mean of these values is assigned to the phrase as a score. This final score then corresponds to the scale of the rest of the single word suggestions. An example for correction candidates for erroneously joined words is shown in Table 3.

Table 3. Extract from the translation model for multiword errors

soronkívül	soron kívül	0.02074
soronkívül	soronkívül	0.01459

4.2 The Language Model

The language model is responsible for taking the lexical context of the words into account. In order to have a proper language model, it should be built on a correct, domain specific corpus by acquiring the required word n-grams and the corresponding probabilities. Since the only manually corrected portion of our corpus was the test set, we could not build such a language model. Though there are orthographically correct texts of other, mostly general domains, the n-gram statistics of these would not correspond to the characteristics of the clinical domain. That is why we did not use such texts to build our language model. However, we found that our clinical corpus contains several very frequent word sequences, but there are a relatively smaller amount of different n-grams compared to general texts. See Table 4 for the difference in the number of n-grams in a general corpus and our clinical documents corpus.

Table 4. The number of different n-grams in a 1,000,000-word general and clinical corpus of Hungarian

	General text	Clinical text
1-grams	127784	106113
2-grams	505253	416362
3-grams	781917	622053

We assumed that the frequency of correct occurrences of a certain word sequence can be expected to be higher than that of the same sequence containing a misspelled word. Of course, the test set that we used for evaluation was separated from the corpus prior to building the language model. Otherwise the word sequences would have corresponded to these, and no correction would have been made.

The documents in the corpus were split into sentences at hypothesized sentence boundaries (finding sentence boundaries was often quite challenging in our corpus) along with applying tokenization as a preprocessing step. The average length of these-quasi-sentences is 8.58 tokens. We used a 3-gram language model due to the relatively short sentences, longer mappings cannot be expected. Our measurements confirmed this: choosing a higher n resulted in worse accuracy.

4.3 Decoding

The result of formula (1) is determined by the decoding algorithm of the SMT system based on the above models. To carry out decoding, we used the widely used Moses toolkit. The parameters of decoding can be set in the Moses configuration file, thus they can be changed easily in order to adapt the system to new circumstances and weighting schemes. During decoding, each input sentence is corrected by creating the translation models based on the suggestions generated for the words occurring in the actual sentence, and using the pre-built abbreviation translation model and the language model. The parameters for decoding were set as follows:

– **Weights of the Translation Models:** since the contents of the phrase tables do not overlap, their weights can be set independently. As mentioned earlier, the correction of the texts was meant as a normalization process rather than adjusting them to a strict orthographic standard. In the case of correcting abbreviations, the goal was to choose the same form for each abbreviation with high probability, but giving a slight chance for each form for special cases. We guaranteed this by giving the abbreviation translation model a higher weight.
– **Language Model:** a 3-gram language model was applied, which was given a lower weight than the translation models in order to prevent the harmful effect of the possibly erroneous n-grams due to the incorrect word forms in the corpus that we used for building this model.
– **Reordering Constraint:** when translating between different languages in a traditional translation task, the reordering of some words within a sentence might be necessary. However, word order changes are not allowed in our application, since modifications can only occur within words or by splitting some words, but cannot change the structure of the sentence. Thus a monotone decoding was applied.
– **Penalty for Difference in the Length of Sentences Before and After Correction:** since the length of a sentence measured in number of tokens cannot change significantly during correction, there is no need to apply a penalty factor of the decoder for this parameter. (The theoretical maximum in the change of the length for a sentence is doubling it by inserting a space to every word, but the necessary number of space insertions was at most two per sentence in the test set.)

5 Results

In order to evaluate our system, a manually corrected test set of clinical documents was necessary. We randomly selected 2000 sentences and sentence fragments from the corpus, from various clinical departments and corrected these texts regarding both tokenization and spelling. The remaining part of the corpus contained 978,000 sentences and that was used for creating the language model. Both sets of sentences only contained free-text parts of clinical reports. Tabular laboratory data, measurement results, headers, ICD codes and other structured content were previously filtered out. In spite of this, there were still a high number of sentences both in the training and test sets that contained hardly any real words, consisting of sequences of abbreviations and numbers while having a clearly Hungarian syntax.

Moreover, we had to accept some non-standard forms that were consistently used throughout the whole corpus, without the standard form appearing at all. We believe that the retrieval of concepts in the texts and their normalization do not require that the normalized version of each word be the orthographically standard form, but mapping variants to a single representation is sufficient. Therefore, we regarded all of these considerations when creating the test set and created multiple corrected references with possible corrections. The baseline to which we compared our contextual error correction system was an implementation that simply replaced every word with the first ranked suggestion that the suggestion generation system generated.

We evaluated how well each system performed on correcting erroneous words in the test set. The size of the test set was 19148 tokens (6744 types), of which 1289 tokens (847 types) were misspelled or in a non-standard form. More than half of these erroneous words are potential abbreviations of length less than 4 characters. Table 5 shows the performance of each system. The overall accuracy (i.e. the ratio of the well corrected words relative to the number of erroneous words) of the baseline system was 72.5%, while the SMT system improved the results significantly to 88.28%. We performed several experiments with different parameter settings that resulted in worse overall quality, but handled some special phenomena better. Table 6 shows some originally erroneous sentences with their corrections and the reference correction as well. The examples are chosen so that they contain different types of sentences occurring in the corpus.

Table 5. Evaluation results of the context-aware system and the 1-best baseline

System	Accuracy
Baseline (1-best)	72.5%
SMT-based context-aware	88.28%

We also performed manual evaluation of the results, where we found that even though there are several cases, where none of the correction systems is able to find the correct form of a word, the SMT-based context-aware system resulted in

Table 6. Originally erroneous sentences with the automatic correction of the baseline and the SMT systems and the manually corrected reference

Original sentence	csppent előírés szerint ,
Baseline correction	cseppent előír és szerint ,
SMT correction	cseppent előírás szerint ,
Reference	cseppent előírás szerint ,
Original sentence	th : mko tovább 1 x duotrav 3 ü-1 rec , íb : 2 x azoipt 3 ü-1 rec
Baseline correction	th : mko tovább 1 x duotrav 3 ü-1 sec , kb : 2 x azoipt 3 ü-1 sec
SMT correction	th. : mko tovább 1 x duotrav 3 ü-1 rec , kb : 2 x azopt 3 ü-1 rec
Reference	th. : mko. tovább 1 x duotrav 3 ü-1 rec , kb. : 2 x azopt 3 ü-1 rec
Original sentence	/alsó m?fogsor .
Baseline correction	/alsó műfogsor .
SMT correction	alsó műfogsor .
Reference	alsó műfogsor .
Original sentence	vértelt nyállkahártyák , kp erezett conjuctiva , fehér sclera .
Baseline correction	vértelt nyálkahártyák , kp erezett conjunctiva , fehér sclera .
SMT correction	vértelt nyálkahártyák , kp. erezett conjunctiva , fehér sclera .
Reference	vértelt nyálkahártyák , kp. erezett conjunctiva , fehér sclera .

Table 7. Examples for the transformation of a correct sentence to another correct sentence with very similar meaning, but different words

Original sentence	homályos látást panaszol .	*(s/he complains about blurred vision)*
SMT correction	homályos látás panaszok .	*(complaints of blurred vision)*
Original sentence	panasz nem volt .	*(there were no complaints)*
SMT correction	panasza nem volt .	*(s/he didn't have any complaints)*

such words that are much "closer" to the real correction. Even in cases when an originally correct word is modified, the second system results in an appropriate word, which is also correct in its context, however the baseline system usually replaces these words with some meaningless strings. These are usually real-word errors, when an originally correct word form is transformed to another correct word or if the original form is not correct, it might be corrected to a word that is correct and grammatically appropriate in the sentence, nevertheless it is still not the actually expected correction. Some examples are in Table 7. These effects originate mainly from the language model that also contains some improper n-grams.

6 Conclusion

In our paper, we presented a method to correct single spelling errors with high accuracy in Hungarian clinical records written in a special variant of domain specific language containing a lot of abbreviations. Besides applying morphological rules and statistics on the word level, lexical context is also considered during correction. Due to the lack of a corpus normalized to proper standard orthography, a practical goal in our work was to consider frequently used word forms as a quasi-standard. Applying our method to raw clinical free-text data, a normalized representation can be achieved that is of crucial importance for further processing steps.

Our method is not perfect, in our paper, we presented some difficult situations that the system is not able to handle yet. We have some future plans of utilizing a richer model containing lemmas and part-of-speech tags and we also expect some improvement if the language model is built iteratively from already corrected texts.

We showed that applying an SMT framework as a spelling correction system is appropriate and can achieve high accuracy.

Acknowledgement. This work was partially supported by TÁMOP – 4.2.1.B – 11/2/KMR-2011-0002 and TÁMOP-4.2.2./B-10/1-2010-0014.

References

1. Bao, Z., Kimelfeld, B., Li, Y.: A graph approach to spelling correction in domain-centric search. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, HLT 2011, vol. 1, pp. 905–914. Association for Computational Linguistics, Stroudsburg (2011)

2. Boswell, D.: CSE 256 (Spring 2004) language models for spelling correction (2004)
3. Brill, E., Moore, R.C.: An improved error model for noisy channel spelling correction. In: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics, ACL 2000, pp. 286–293. Association for Computational Linguistics, Stroudsburg (2000)
4. Brockett, C., Dolan, W.B., Gamon, M.: Correcting ESL Errors Using Phrasal SMT Techniques. In: Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, pp. 249–256. Association for Computational Linguistics, Sydney (2006)
5. Church, K.W., Gale, W.A.: Probability scoring for spelling correction. Statistics and Computing 1(2), 93–103 (1991)
6. Crowell, J., Zeng, Q., Ngo, L., Lacroix, E.: A frequency-based technique to improve the spelling suggestion rank in medical queries. J. Am. Med. Inform. Assoc. 11(3), 179–85
7. Ehsan, N., Faili, H.: Grammatical and context-sensitive error correction using a statistical machine translation framework. Softw., Pract. Exper. 43(2), 187–206 (2013)
8. Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., Herbst, E.: Moses: Open Source Toolkit for Statistical Machine Translation. In: Proceedings of the ACL 2007 Demo and Poster Sessions, pp. 177–180. Association for Computational Linguistics, Prague (2007)
9. Kukich, K.: Techniques for automatically correcting words in text. ACM Comput. Surv. 24(4), 377–439 (1992)
10. Noeman, S., Madkour, A.: Language independent transliteration mining system using finite state automata framework. In: Proceedings of the 2010 Named Entities Workshop, NEWS 2010, pp. 57–61. Association for Computational Linguistics, Stroudsburg (2010)
11. Novák, A.: What is good Humor like? In: I. Magyar Számítógépes Nyelvészeti Konferencia, pp. 138–144. SZTE, Szeged (2003)
12. Oflazer, K., Güzey, C.: Spelling correction in agglutinative languages. In: Proceedings of the Fourth Conference on Applied Natural Language Processing, ANLC 1994, pp. 194–195. Association for Computational Linguistics, Stroudsburg (1994)
13. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: BLEU: a method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL 2002, pp. 311–318. Association for Computational Linguistics, Stroudsburg (2002)
14. Park, Y.A., Levy, R.: Automated whole sentence grammar correction using a noisy channel model. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, HLT 2011, vol. 1, pp. 934–944. Association for Computational Linguistics, Stroudsburg (2011)
15. Patrick, J., Nguyen, D.: Automated proof reading of clinical notes. In: Gao, H.H., Dong, M. (eds.) PACLIC, pp. 303–312. Digital Enhancement of Cognitive Development, Waseda University (2011)
16. Pirinen, T.A., Lindén, K.: Finite-state spell-checking with weighted language and error models. In: Proceedings of the Seventh SaLTMiL Workshop on Creation and Use of Basic Lexical Resources for Less-resourced Languages, Valletta, Malta, pp. 13–18 (2010)

17. Prószéky, G., Kis, B.: A unification-based approach to morpho-syntactic parsing of agglutinative and other (highly) inflectional languages. In: Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics, ACL 1999, pp. 261–268. Association for Computational Linguistics, Stroudsburg (1999)
18. Siklósi, B., Orosz, G., Novák, A., Prószéky, G.: Automatic structuring and correction suggestion system for Hungarian clinical records. In: 8th SaLTMiL Workshop on Creation and Use of Basic Lexical Resources for Less-resourced Languages, pp. 29–34 (2012)
19. Stolcke, A., Zheng, J., Wang, W., Abrash, V.: SRILM at sixteen: Update and outlook. In: Proc. IEEE Automatic Speech Recognition and Understanding Workshop, Waikoloa, Hawaii (December 2011)
20. Turchin, A., Chu, J.T., Shubina, M., Einbinder, J.S.: Identification of misspelled words without a comprehensive dictionary using prevalence analysis. In: AMIA Annual Symposium Proceedings, pp. 751–755 (2007)

Pronunciation Extraction from Phoneme Sequences through Cross-Lingual Word-to-Phoneme Alignment

Felix Stahlberg[1], Tim Schlippe[1], Stephan Vogel[2], and Tanja Schultz[1]

[1] Karlsruhe Institute of Technology, Cognitive Systems Lab.
Adenauerring 4, 76131 Karlsruhe, Germany
`felix.stahlberg@student.kit.edu`, {`tim.schlippe,tanja.schultz`}`@kit.edu`
[2] Qatar Foundation, Qatar Computing Research Institute
Al-Nasr Tower A, 21st Floor, Doha, Qatar
`svogel@qf.org.qa`

Abstract. With the help of written translations in a source language, we cross-lingually segment phoneme sequences in a target language into word units using our new alignment model *Model 3P* [17]. From this, we deduce phonetic transcriptions of target language words, introduce the vocabulary in terms of word IDs, and extract a pronunciation dictionary. Our approach is highly relevant to bootstrap dictionaries from audio data for Automatic Speech Recognition and bypass the written form in Speech-to-Speech Translation, particularly in the context of under-resourced languages, and those which are not written at all.

Analyzing 14 translations in 9 languages to build a dictionary for English shows that the quality of the resulting dictionary is better in case of close vocabulary sizes in source and target language, shorter sentences, more word repetitions, and formal equivalent translations.

Keywords: pronunciation dictionary, under-resourced languages, speech-to-speech translation, word segmentation.

1 Introduction

There are over 7,000 living languages and dialects in the world [8]. Automatic Speech Recognition (ASR) and Machine Translation (MT) systems exist only for few of them. Porting rapidly and economically language technology to new unseen and under-resourced languages is in particular required in situations where languages with few linguistic resources suddenly appear in the focus of interest. Another challenge is the merely spoken nature of many languages and dialects, some of which are widespread despite the lack of a written script [16,13]. However, language technology generally requires a written script nowadays.

In [17] and in this work, we take first steps towards gathering training data for ASR and MT systems for an unseen target language rapidly and at low cost: We segment phoneme sequences into word units using information from another language. We then deduce word pronunciations from these units, introduce the

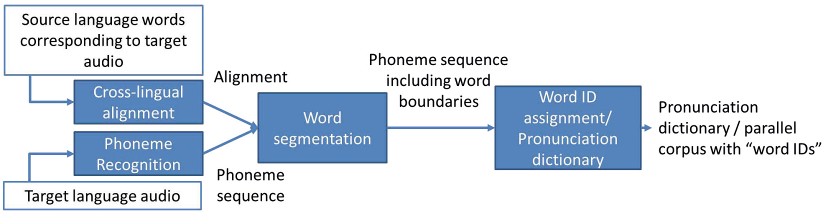

Source language words corresponding to target audio
Cross-lingual alignment
Phoneme Recognition
Target language audio
Alignment
Phoneme sequence
Word segmentation
Phoneme sequence including word boundaries
Word ID assignment/ Pronunciation dictionary
Pronunciation dictionary / parallel corpus with "word IDs"

Fig. 1. Long-term scenario

vocabulary in terms of word IDs, and extract a pronunciation dictionary. Dictionaries are used to train speech processing systems by describing the pronunciation of words in manageable units such as phonemes [12]. As dictionaries are so fundamental, much care has to be taken to select a dictionary that is as free of errors as possible. Thus our approach is highly relevant for Speech-to-Speech Translation (S2S) of under-resourced languages, and those which are not written.

We explore 14 translations in 9 languages to build a dictionary for English. Our method benefits from the fact that written sentences are available in several economically viable languages such as Spanish. We assume that a speaker is available who understands Spanish and who speaks translations of the Spanish sentences in his or her mother tongue. This is a weak assumption, since human simultaneous translations happen frequently in the real world, e.g. in the context of humanitarian aid operations or in multilingual parliament sessions [7]. Our goal is to exploit the phonetic output of such human translators, so that the following scenario comes within reach (Fig. 1):

1) We recognize the spoken translations with a language independent phoneme recognizer. 2) We build an alignment between words in the written Spanish sentence and phonemes in the corresponding recognized phoneme sequence in the target language. 3) Using this cross-lingual alignment, we segment the phoneme sequence into word units. 4a) The word segmentation induces phonetic transcriptions of target language words, which are used in a pronunciation dictionary for ASR systems. 4b) The segmented phoneme sequence is replaced by a sequence of word IDs. This results in a parallel training corpus on the word level for a Statistical MT (SMT) system as described in [2]. Our final goal is to bootstrap an S2S system without any linguistic knowledge of the target language.

While we have focused on *step 2* and *3* in [17], we tackle *step 4a* in this paper – the pronunciation extraction. We test our algorithms on parallel data from the Christian Bible since it is available in many different languages in written form and in some languages also as audio recordings. A variety of linguistic approaches to Bible translation [21] enables us to compare different translations within the same source language. In our experiments, English takes the role of the under-resourced target language. English is by no means under-resourced and comprehensive pronunciation dictionaries are readily available [24]. However, for this exploratory work we feel that understanding the target language gives a deeper insight in the strengths and weaknesses of our algorithms.

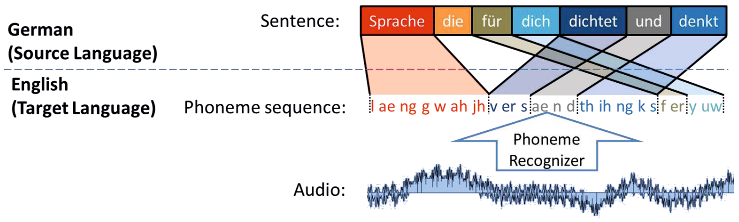


Fig. 2. Word segmentation through word-to-phoneme alignment

2 Word Segmentation

Cross-lingual word-to-phoneme alignments introduced in [2,19,20] and tackled by us with our new alignment model *Model 3P* [17] are the basis for our pronunciation extraction algorithm in Sec. 3. Therefore, this section summarizes the concepts of [17] in condensed form. The word segmentation problem describes the task of segmenting phoneme sequences into word units. We have shown in [17] that unsupervised learning of word segmentation is more accurate when information of another language is used. *Model 3P*[1] for cross-lingual word-to-phoneme alignment extends the generative process of IBM Model 3 by a word length step and additional dependencies for the lexical translation probabilities. Those alignments can be used for the segmentation task as illustrated in Fig. 2. Using *Model 3P* for the alignment between English words and correct Spanish phoneme sequences on the BTEC corpus [10] resulted in 76.5% F-Score (90.0% accuracy [22]) and thus outperformed a state-of-the-art monolingual word segmentation approach [9] by 42% absolute in F-Score (18.2% in accuracy).

3 Word Pronunciation Extraction

3.1 Formal Framework

Let V_{src} be the vocabulary of the source language and $PhonemeSet_{trgt}$ the phoneme set of the target language. The data source we explore in our scenario is a set $DB \subset {V_{src}}^+ \times {PhonemeSet_{trgt}}^+$ of pairs containing a written sentence in the source language and its spoken translation in the target language. As described in Sec. 2, we use *Model 3P* to find word-to-phoneme alignments for each sentence-phoneme sequence pair in DB. An alignment $A_{s,t}$ consists of a mapping between the words in the source language sentence $s \in {V_{src}}^+$ and the phonemes in the target language phoneme sequence $t \in {PhonemeSet_{trgt}}^+$ segmented into word units. We formalize $A_{s,t}$ as a word over an alphabet containing pairs of source language words and target language phoneme sequences.

$$A_{s,t} \in ({PhonemeSet_{trgt}}^+ \times V_{src})^+$$

[1] A multi-threaded implementation is available at `http://pisa.googlecode.com/`

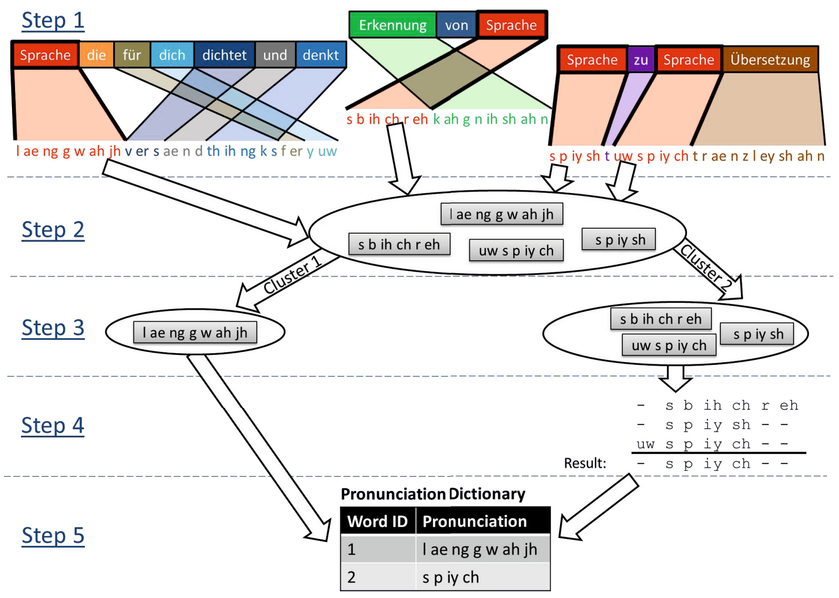


Fig. 3. Steps 1-5 on a German-English example ("Sprache zu Sprache Übersetzung" → "Speech to speech translation", "Sprache die für dich dichtet und denkt" → "Language verses and thinks for you", "Erkennung von Sprache" → "Speech recognition")

Each element in $A_{s,t}$ contains a hypothetical target language *word* represented by its phonemes and the source language word aligned to it. We postulate that the source language words are elements in s, and that concatenating all target language words results in the complete phoneme sequence t.

3.2 Pronunciation Extraction Algorithm

We extract pronunciations based on the assumption, that phoneme sequences, that are aligned to the same source language word, are likely to represent the same target language word. They only differ due to phoneme recognition and alignment errors. From the linguistic point of view, this is not always the case: in Fig. 3, the German word *Sprache* has two different English translations (*Speech* and *Language*). *Step 3* of our algorithm addresses this special case.

We build the pronunciation dictionary iteratively by repeating the following steps until all source language words are marked. The steps are visualized in Fig. 3 with German as source language and English as target language.

1. Select the most frequent unmarked source language word $v \in V_{src}$ and mark it.
2. Collect the set $P \subset PhonemeSet_{trgt}{}^{+}$ of all phoneme sequences, which are aligned to v (hypothetical target language words):[2]

[2] For technical reasons, we define the $\in$ sign for a symbol $x \in \Sigma$ and a word $w \in \Sigma^{+}$ as $x \in w :\Leftrightarrow \exists i \in \mathbb{N} : x = w_i$

$$P \leftarrow \{h | \exists (s,t) \in DB : (h,v) \in A_{s,t}\}$$

3. Group the phoneme sequences into clusters $C \subset 2^P$. We applied the clustering algorithm DBSCAN [6] ($\epsilon = 1$, $minPts = 3$) implemented in the ELKI [1] environment with the Levenshtein distance metric. This step aims to separate elements in P from each other, which do not represent the same target language word.
4. At this point, the clusters should contain phoneme sequences representing the same target language word, but differing due to alignment and phoneme recognition errors. Thus we try to reconstruct the correct phoneme sequence for each cluster by merging its elements with the `nbest-lattice` [18] program.We obtain a set $H \subset PhonemeSet_{trgt}{}^{+}$ of phoneme sequences, which are now assumed to correspond to real target language words.
5. For each pronunciation $h \in H$, we choose a new word ID $id_h \in \mathbb{N}$ and add both to the pronunciation dictionary $Dict$.

When we apply the general algorithm above to the example in Fig. 3, the variables have following values:

1. $v = \texttt{Sprache}$
2. $P = \{\texttt{s b ih ch r eh}, \texttt{l ae ng g w ah jh}, \texttt{uw s p iy ch}, \texttt{s p iy sh}\}$
3. $C = \{\{\texttt{l ae ng g w ah jh}\}, \{\texttt{s b ih ch r eh}, \texttt{uw s p iy ch}, \texttt{s p iy sh}\}\}$
4. $H = \{\texttt{l ae ng g w ah jh}, \texttt{s p iy ch}\}$
5. $Dict = \{(1, \texttt{l ae ng g w ah jh}), (2, \texttt{s p iy ch})\}$

4 Experiments

4.1 Corpus

We tested our pronunciation extraction algorithm on parallel data from the Christian Bible. A variety of linguistic approaches to Bible translation (Dynamic equivalence, formal equivalence, and idiomatic translation [21]) enables us to compare different translations within the same source language. In our experiments, English takes the role of the under-resourced target language. For this exploratory work we feel that understanding the target language gives a deeper insight in the strengths and weaknesses of our algorithms. The English Standard Version (ESV) [5] is a literal English translation of the Christian Bible [3]. Half of the words in the vocabulary occur three times or more in the text, 30.5% have only one occurrence. High word frequencies are suitable for our extraction algorithm since we merge more phoneme sequences in *Step 4* which leads to better error correction as shown in Sec. 4.4. Verses in the ESV Bible are identified by unique verse numbers (such as *Galatians 5:22*), which are consistent with verse numbers in other Bible translations. Based on these numbers, we extracted a parallel and verse-aligned corpus consisting of 30.6k English Bible verses (target language) and 14 written translations of them (Tab. 1).

To generate the target language phoneme sequences, we replaced the words in the ESV Bible with their canonical pronunciations and removed word boundary markers. Thereby we simulate the output of a perfect phoneme recognizer (0% Phoneme Error Rate) and refrain from dealing with pronunciation variants and phoneme recognition errors. However, we design our algorithms to be robust against recognition errors. The pronunciations were taken from the CMUdict [24] or generated with a grapheme-to-phoneme model trained on it (39 phonemes).

4.2 Evaluation Measures

We measure the quality of the word segmentation (Sec. 2) in terms of **accuracy** [22]. Additionally, we suggest 3 different evaluation measures, which address different aspects of the quality of the extracted dictionary.

Let I be the set of all word IDs in the extracted dictionary $Dict : I \rightarrow PhonemeSet_{trgt}{}^{+}$. We measure the structural quality of $Dict$ by the **Out-Of-Vocabulary rate (OOV)** on a subset of the English ESV Bible. The OOV rate can not be calculated directly since $Dict$ contains word IDs instead of written words consisting of graphemes like in the ESV Bible. Therefore, a mapping between the word IDs and the written words is required. Let V_{trgt} be the vocabulary of the ESV Bible (written words) and $Dict_{ref} : V_{trgt} \rightarrow PhonemeSet_{trgt}{}^{+}$ the reference dictionary with the correct pronunciations. The mapping $m : I \rightarrow V_{trgt}$ assigns each word ID to the written word with the most similar pronunciation.

$$m(n) = \arg\min_{v \in V_{trgt}} d_{edit}(Dict(n), Dict_{ref}(v)) \quad (1)$$

where d_{edit} denotes the edit distance. The set $m(I)$ of matched vocabulary entries in $Dict_{ref}$ is then used to calculate the OOV rate.

Table 1. Overview of used Bible translations

ID	Language	Full Bible Version Name	Number of running words
bg	Bulgarian	Bulgarian Bible	643k
cs	Czech	Bible 21	547k
da	Danish	Dette er Biblen på dansk	653k
de1	German	Schlachter 2000	729k
de2	German	Luther Bibel	698k
es1	Spanish	Nueva Versión Internacional	704k
es2	Spanish	Reina-Valera 1960	706k
es3	Spanish	La Biblia de las Américas	723k
fr1	French	Segond 21	756k
fr2	French	Louis Segond	735k
it	Italian	Nuova Riveduta 2006	714k
pt1	Portugese	Nova Versão Internacional	683k
pt2	Portuguese	João Ferreira de Almeida Atualizada	702k
se	Swedish	Levande Bibeln	595k
en	*English*	*English Standard Version*	*758k*

While the OOV rate indicates the coverage of *Dict* on a Bible text, the **Phoneme Error Rate (PER)** reflects the quality of the extracted pronunciations on the phoneme level. It is defined as the average edit distance between the entries in *Dict* and the closest entry in the reference dictionary $Dict_{ref}$:

$$PER = \frac{\sum_{n \in I} d_{edit}(Dict(n), Dict_{ref}(m(n)))}{|I|} \tag{2}$$

The **Hypo/Ref ratio** indicates how many hypothesis entries in *Dict* are mapped by m to a single reference entry in $Dict_{ref}$ on average ($|I|$ divided by $|m(I)|$). The higher the Hypo/Ref ratio, the more pronunciations are extracted unnecessarily.

4.3 Which Source Translation Is Favorable?

Fig. 4 shows the distribution of the edit distances between the extracted pronunciations and the closest entries in the reference dictionary (pairs $(n, m(n))$) for each of the 14 translations in Tab. 1. For example, the length of the dark blue bar above the *es3* label shows, that using the Spanish *La Biblia de las Américas* translation, 4,464 of the 21,561 extracted pronunciations (20.7%) contain no or only minor phoneme errors (edit distance lower than 0.1). The translations are sorted by accuracy (descending from left to right). We can observe, that the red bar (interval $[0.1, 0.2)$) is small, because a word has to contain at least 6 phonemes (and 1 phoneme error) to fall into this class and English words are usually shorter. Apart from these side effects, the edit distance usually seems to be approximately uniformly distributed in $[0, 0.6)$, and only a few outliers have higher edit distances. Exceptions are *bg* and *cs*. The red line marks the actual size of the ESV Bible vocabulary. Fig. 5 breaks down the extracted pronunciations by the differently colored absolute number of insertions, deletions, and substitutions. 20% of all entries contain no phoneme error, 50% no more than one error. Only about 30% of all entries contain 3 or more phoneme errors.

We investigate the impact of four factors to our evaluation measures.

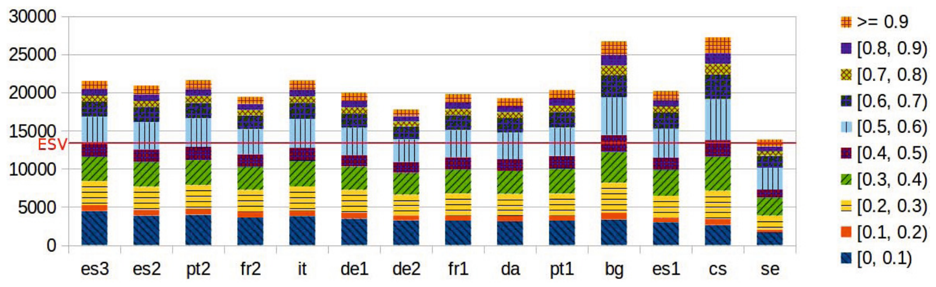


Fig. 4. Distribution of the edit distances between the extracted pronunciations and the nearest entry in the reference dictionary for all 14 source translations

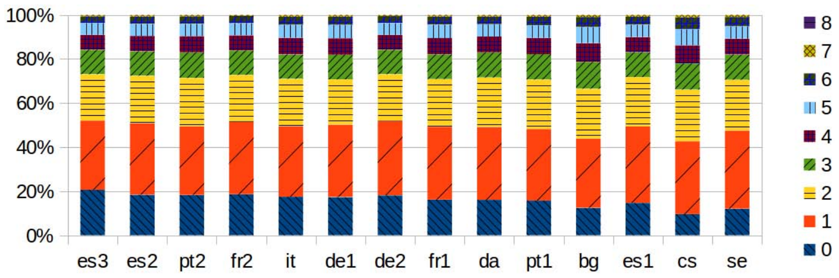


Fig. 5. Distribution of the absolute number of insertions, deletions, and substitutions between the extracted pronunciations and the nearest entry in the reference dictionary

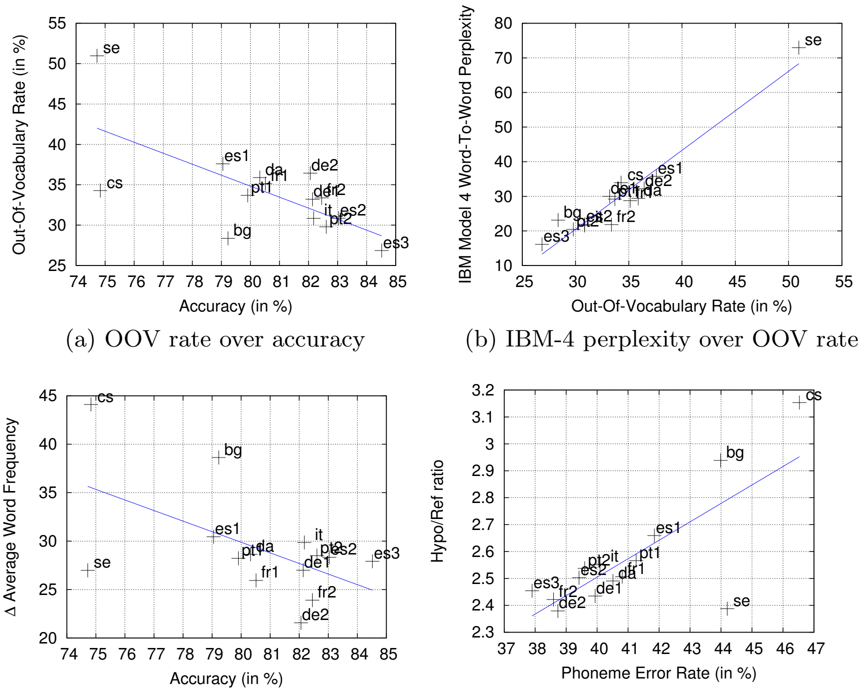
- Δ **Vocabulary Size.** The difference between the vocabulary size of the source translation and the size of the ESV vocabulary.
- Δ **Average Number of Words per Verse.** The difference between the average verse length in the source translation and in the ESV Bible.
- Δ **Average Word Frequency** The difference between the average number of word repetitions in the source language and in the ESV Bible.
- **IBM-4 PPL.** To measure the general correspondence of the translation to IBM-Model based alignment models, we run GIZA++ [14] with default configuration on the word level and use the final perplexity of IBM Model 4 [4].

Tab. 2 shows the Pearson's correlation coefficient $|r|$ [15] between those four factors and our evaluation measures from Sec. 4.2. Fig. 6 plots some of the point clouds with their regression line. We observe a rather weak linear correlation between the OOV rate and the word segmentation accuracy in Fig. 6 (a) ($r = 0.68$): The better the word segmentation, the closer the extracted and the reference dictionary structurally. The dominant factor for the OOV rate is the IBM-4 PPL (Fig. 6 (b), $r = 0.96$). This suggests, that a literal translation is more important than cross-lingual linguistic dissimilarities. This hypothesis is supported by the

Table 2. Absolute correlation coefficients $|r| \in [0, 1]$ between our evaluation measures and different influencing factors (high $|r|$ - high linear correlation)

$\|r\|$	**Accuracy**	**PER**	**Hypo/Ref ratio**	**OOV rate**
Δ Vocabulary size	0.47	0.71	0.98	0.31
Δ Average number of words	0.59	0.72	0.85	0.06
Δ Average word frequency	0.55	0.79	0.97	0.21
IBM-4 PPL	0.77	0.54	0.10	0.96
PER	0.94	-	-	-
Hypo/Ref Ratio	0.53	0.77	-	-
OOV rate	0.68	0.40	0.24	-

wide variance of the evaluation measures between different translations within the same source language: *es3* has 5.5% higher accuracy, 10.7% lower OOV rate, and 3.9% lower PER (absolute) than *es1* since *es3* is a very literal translation [11]. Similar results can be observed for French and Portuguese.There is only a weak linear correlation of the average word frequency with the accuracy (Fig. 6 (c)), but a stronger correlation with the PER. Consequently, frequent word repetitions improve the quality of the extracted pronunciations on the phoneme level since *Step 4* in our extraction algorithm in Sec. 3.2 merges many phoneme sequences and can correct errors more effectively. The Hypo/Ref ratio is highly correlated with both the vocabulary size and the average word frequency. This suggests, that *Step 3* in our extraction algorithm needs to be improved: Often one single cluster per source language word is generated, and *Step 4* merges words which are different in the target language. This high correlation also uncovers another point for improvement: Pronunciations extracted from different source language words can not be merged. For example, all three German definite articles are translated to *the*, so there are at least three dictionary entries for *the* alone.



(a) OOV rate over accuracy

(b) IBM-4 perplexity over OOV rate

(c) Divergence of vocabulary size over accuracy

(d) Hypo/Ref ratio over Phoneme Error Rate

Fig. 6. Different influencing factors for the evaluation measures in Sec. 4.2

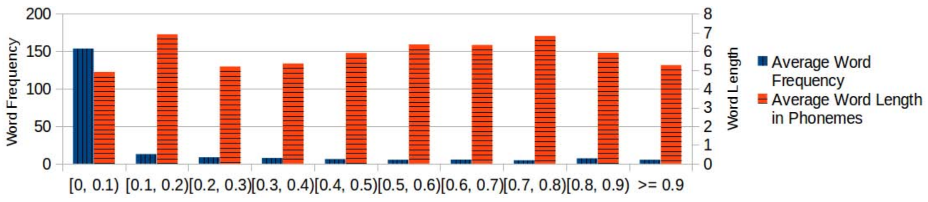


Fig. 7. Average word frequency and number of phonemes per word over the PER (*es3*)

4.4 Which Words Are Extracted Correctly?

This section describes the characteristics of words which are likely to be extracted correctly when the source translation *es3* is used. Experiments with other source translations show similar results. Fig. 7 indicates, that frequently repeated words tend to contain no or only minor errors on the phoneme level (blue bar) while there is no such clear correlation with the number of phonemes per word (red bar). A look at some extracted pronunciations reveals two major sources of errors for words with only 1-2 phoneme errors:

1. Single phonemes are added or dropped in the beginning or end of a word because of off-by-one alignment errors:
   - `z f ih s t s` instead of `f ih s t s` (fists)
   - `ih k s t` instead of `f ih k s t` (fixed)
2. Different words with the same stem are merged together:
   - `s ih d uw s ih t` instead of `s ih d uw s t` (seduced) or `s ih d uw s ih ng` (seducing)
   - `ih k n aa l ih jh m` instead of `ih k n aa l ih jh` (acknowledge) or `ih k n aa l ih jh m ah n t` (acknowledgement)

Entries with two phoneme errors or more often contain two words because of missing word boundaries between words often occurring in the same context:

- `w er ih n d ih g n ah n t` (were indignant)
- `f ih n ih sh t ih t` (finished it)

We assume that this kind of errors would not be very critical when using the dictionary in an S2S system since those words are likely to be stuck together as *phrase* later in the training process of the translation model anyway.

4.5 Combining Multiple Translations

In case of several written translations, we first extract the pronunciation dictionary with each source translation separately, and then combine all of them in a single dictionary. To combine the set of dictionaries, we first add the translation tags (i.e. *es3*, *de2*. . .) to the word IDs to obtain globally unique IDs. Second, we concatenate all dictionaries and remove homophones. Starting out from the *es3* dictionary, we successively combined more dictionaries of other translations ordered descending by the word segmentation accuracy. Fig. 8 suggests, that the

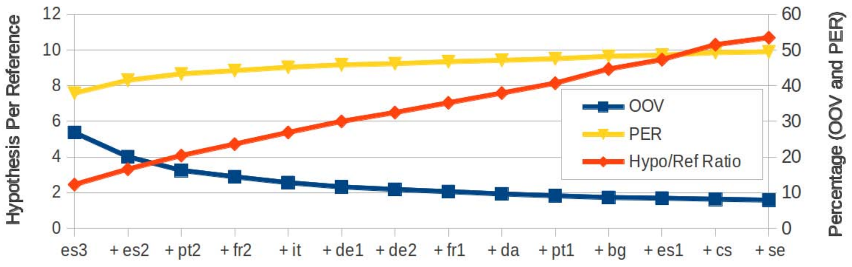


Fig. 8. Evaluation measures over the number of combined source translations

OOV rate decreases slightly exponentially with the number of combined translations. At the same time, the Hypo/Ref ratio increases linearly. The PER only increases slightly. Combining all 14 translations results in a dictionary with only 7.9% OOV rate, but more than 9 of 10 dictionary entries are extracted unnecessarily (Hypo/Ref ratio 10.7:1). Such a dictionary is far too noisy for practical use, but it shows, that experiments with different source translations extract different English words. Therefore, our future work will also focus on how to remove this noise and explore the synergy of multiple translations.

5 Conclusion and Future Work

Using written translations in one or many source languages, we cross-lingually segmented phoneme sequences in a target language using our alignment model *Model 3P* [17]. We proposed a new algorithm for extracting a pronunciation dictionary with word IDs from these segmentations and alignments, which can be used in an S2S system bypassing the written form of a non-written or under-resourced target language. In our exploratory experiments, we extracted English pronunciations by using 14 different translations in 9 languages. With a Spanish translation (*es3*), we built a dictionary for the ESV Bible [5] with 26.9% OOV rate, in which most of the pronunciations contain not more than one wrong phoneme. Combining dictionaries from multiple translations drops the OOV rate to 7.9%, but increases the number of unnecessary entries. This shows, that depending on the used translation, different English words are extracted.

In the future, we plan to enhance our pronunciation extraction algorithm based on the results from Sec. 4.3: *Step 3* needs to be improved to separate pronunciation variants and different words with the same translation more reliably. The algorithm needs to be adjusted to allow merging of pronunciations generated by distinct source language words. Off-by-one pronunciation errors due to alignment errors may be reduced by reinforcing the alignments with the extracted pronunciations after each iteration of our algorithm. Monolingual word segmentation methods as in [9] may give additional hints. When combining multiple dictionaries, a mechanism is to be found to filter accurate entries and benefit from the lower OOV rate while keeping the Hypo/Ref ratio constant. In a next step, we will use a phoneme recognizer to obtain the phoneme sequences. Such a phoneme recognizer can be bootstrapped using recognizers from other languages and adaptation techniques as presented in [23]. Furthermore, we intend to use the extracted dictionaries in a speech recognizer for a truly under-resourced language. The final goal is to build an S2S system without any linguistic knowledge of the target language.

References

1. Achtert, E., Goldhofer, S., Kriegel, H.P., Schubert, E., Zimek, A.: Evaluation of Clusterings–Metrics and Visual Support. In: ICDE (2012)
2. Besacier, L., Zhou, B., Gao, Y.: Towards Speech Translation of Non-Written Languages. In: SLT (2006)
3. Borland, J.A.: The English Standard Version-A Review Article. Faculty Publications and Presentations, 162 (2003)
4. Brown, P.F., Pietra, V.J.D., Pietra, S.A.D., Mercer, R.L.: The Mathematics of Statistical Machine Translation: Parameter Estimation. Computational Linguistics 19(2), 263–311 (1993)
5. Crossway: The Holy Bible: English Standard Version (2001)
6. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases With Noise. In: KDD (1996)
7. Gollan, C., Bisani, M., Kanthak, S., Schlüter, R., Ney, H.: Cross Domain Automatic Transcription on the TC-STAR EPPS Corpus. In: ICASSP (2005)
8. Gordon, R.G., Grimes, B.F.: Ethnologue: Languages of the World, 15th edn. SIL International (2005)
9. Johnson, M., Goldwater, S.: Improving Non-Parameteric Bayesian Inference: Experiments on Unsupervised Word Segmentation with Adaptor Grammars. In: HLT-NAACL (2009)
10. Kikui, G., Sumita, E., Takezawa, T., Yamamoto, S.: Creating Corpora for Speech-to-Speech Translation. In: Eurospeech (2003)
11. Lockman: La Biblia de las Américas (1986), `http://www.lockman.org/lblainfo/` (accessed on February 28, 2013)
12. Martirosian, O., Davel, M.: Error Analysis of a Public Domain Pronunciation Dictionary. In: PRASA (2007)
13. Nettle, D., Romaine, S.: Vanishing Voices: The Extinction of the World's Languages. Oxford University Press (2000)
14. Och, F.J., Ney, H.: A Systematic Comparison of Various Statistical Alignment Models. Computational Linguistics 29(1), 19–51 (2003)
15. Rodgers, J.L., Nicewander, W.A.: Thirteen Ways to Look at the Correlation Coefficient. The American Statistician 42(1), 59–66 (1988)
16. Schultz, T., Kirchhoff, K. (eds.): Multilingual Speech Processing. Academic Press, Amsterdam (2006)
17. Stahlberg, F., Schlippe, T., Vogel, S., Schultz, T.: Word Segmentation Through Cross-Lingual Word-to-Phoneme Alignment. In: SLT (2012)
18. Stolcke, A., Konig, Y., Weintraub, M.: Explicit Word Error Minimization in N-best List Rescoring. In: Eurospeech (1997)

19. Stüker, S., Waibel, A.: Towards Human Translations Guided Language Discovery for ASR Systems. In: SLTU (2008)
20. Stüker, S., Besacier, L., Waibel, A.: Human Translations Guided Language Discovery for ASR Systems. In: Interspeech (2009)
21. Thomas, R.L.: Bible Translations: The Link Between Exegesis and Expository Preaching. The Masters Seminary Journal 1, 53–74 (1990)
22. VIM: International Vocabulary of Basic and General Terms in Metrology. International Organization, pp. 09–14 (2004)
23. Vu, N.T., Kraus, F., Schultz, T.: Rapid Building of an ASR System for Under-Resourced Languages Based on Multilingual Unsupervised Training. In: Interspeech (2011)
24. Weide, R.: The Carnegie Mellon Pronouncing Dictionary 0.6 (2005)

Can Statistical Tests Be Used for Feature Selection in Diachronic Text Classification?

Sanja Štajner and Richard Evans

Research Group in Computational Linguistics
University of Wolverhampton, UK
{sanjastajner,R.J.Evans}@wlv.ac.uk

Abstract. In spite of the great number of diachronic studies in various languages, the methodology for investigating language change has not evolved much in the last fifty years. Following the progressive trends in other fields, in this paper, we argue for the adoption of a machine learning approach in diachronic studies, which could offer a more efficient analysis of a large number of features and easier comparison of the results across different genres, languages and language varieties. We suggest the use of statistical tests as an initial step for feature selection in an approach which uses the F-measure of the classification algorithms as a measure of the extent of diachronic changes. Furthermore, we compare the performance of the classification task after the feature selection made by statistical tests and the CfsSubsetEval attribute selection algorithm. The experiments were conducted on the British part of the biggest existing diachronic corpora of 20th century written English language – the 'Brown family' of corpora, using 23 different stylistic features. The results demonstrated that the use of the statistical tests for feature selection can significantly increase the accuracy of the classification algorithms.

1 Introduction

Approaches to text classification continue to develop from those based on knowledge engineering techniques that prevailed in the 1980s, in which classifiers were defined manually by domain experts. In the 1990s, these methods were superseded by those relying on machine learning which provided high levels of efficacy, cost effectiveness, in terms of time and manual effort, and easy adaptation for use in different scenarios and domains [26]. Continuous methodological improvements in the field of text classification has led to the adoption of more effective and less labour intensive approaches in place of those requiring a large amount of human annotation. By contrast, approaches to the linguistic study of stylistic variation and change were more conservative and did not follow the progressive trends in other related fields.

Early work in the field of stylistic variation and change was based on historical and sociolinguistic approaches, e.g. [14,1,4]. The next generation of stylistic variation studies, e.g. [7,8] employed a corpus-based methodology and the multi-dimensional framework presented in [5,6]. The same methodology was used in a great number of subsequent diachronic studies, e.g. [32,33]. Another set of corpus-based diachronic studies was initiated by the emergence of the diachronic part of the 'Brown family' of corpora in the 1990s. These corpora offered a possibility for diachronic comparison of various lexical, grammatical, syntactic and stylistic features in two major English language varieties – British and American [21] in the period 1961–1991/2. Many diachronic studies of these corpora (e.g. [23,22,24]), shared the same methodology. The corpora were POS tagged, change was presented in terms of absolute and relative differences between the corpora and the statistical significance of that change was measured using the log likelihood function. The first attempt at completely automated feature extraction from the raw text version of these 'Brown family' of corpora in diachronic studies was reported in [30]. The corpora were parsed with Connexor's Machinese Syntax parser[1] and the features were automatically extracted from the parser's output. Statistical significance of the results was measured by the t-test.

In this paper, we adopt the hypothesis that diachronic language change could be seen as a classification problem and therefore addressed by machine learning techniques. To illustrate, if we wish to investigate the degree of change in certain features between the texts published in 1961 and 1991, we could train a classifier on a representative set of labeled texts (using the selected features as variables) and then classify a set of randomly selected unlabeled texts using this classifier. The performance of the classifier (in terms of the F-measure), would then represent the extent of diachronic change in the selected features. In the cases where diachronic changes were most pronounced, the F-measure obtained by the classification algorithm will be at its highest level. More importantly, by using the machine learning approach, we could also take advantage of existing attribute selection algorithms in order to single out from a large set of initial features, those features which underwent the most extensive changes over the observed period. In this paper, we wanted to investigate whether statistical tests and the CfsSubsetEval attribute selection algorithm [15] would improve the accuracy of diachronic classification and whether they would select the same subsets of features. In order to do so, we applied several well-known classification algorithms (Naïve Bayes and different versions of Logistic and Support Vector Machines functions) in Weka[2] on the texts from the British part of the 'Brown family' of corpora, using different subsets of the 23 initial features.

2 Related Work

Altmann et al. [3] and Kroch [19] proposed the logistic function as the underlying S-shaped curve of linguistic change. Although the correctness of this choice was not proved at the time, it was generally considered appropriate to use this function in statistical studies of changing percentages of alternating forms over

[1] http://www.connexor.eu

[2] http://www.cs.waikato.ac.nz/ml/weka/

time ([2,29] in [18]). Twenty years later, Geisler [13] used logistic regression in the study of relativisation variation in Ulster English [12]. Therefore, we decided to include the classifier based on the logistic function for our experiments.

A survey of previous diachronic studies of the 'Brown family' of corpora motivated the development of our initial feature set. Leech and Smith [22] reported a reduction in the use of passive voice between 1961 and 1991/2 in both British and American English. Stajner and Mitkov [30] investigated diachronic changes of four stylistic features: average sentence length, Automated Readability Index [27], lexical density and lexical richness [11]. The results revealed statistically significant changes in these features between 1961 and 1991/2 in both varieties of English and across all four main text categories (Press, Prose, Learned and Fiction). Although in both cases the authors differentiated only between texts across the four main text categories, it is reasonable to expect that some significant changes of these features would also be reported in a separate investigation of the sub-genres (A–R, see Table 1). Mair et al. [23] compared the frequency of occurrence of words with particular parts of speech in the British part of the corpora. They reported a significant increase in the number of nouns and adjectives and a decrease in the frequency of occurrence of pronouns in all four main text categories over the observed period (1961–1991). Usage of verbs underwent a significant increase in the Press and Science categories, and a significant decrease in the Prose and Fiction categories. In the study reported in the current paper, we investigated nine different POS tags. We differentiated between texts across sub-genres (A–R) and calculated two different types of tag frequencies – tag frequency as a percentage of the selected tag in the whole text and tag frequency as an average per sentence. Stajner and Mitkov [31] reported some significant changes in sentence complexity in the period 1961–1991 in three genres of the British part of the corpora.

Identifying the best set of features for a particular classification task is one of the central problems in machine learning. The CfsSubsetEval attribute selection algorithm uses a correlation based approach to the feature selection problem. It is based on the idea that "good feature sets contain features that are highly correlated with the class, yet uncorrelated with each other" [15]. When compared with a wrapper, the CfsSubsetEval gave similar results to the wrapper and even outperformed the wrapper on small datasets [15].

3 Methodology

The corpora, features and experimental settings used in this study are presented in the following three subsections.

3.1 Corpora

We used only the British part of the aforementioned 'Brown family' of corpora [21]:

- the Lancaster-Oslo/Bergen Corpus of British English (LOB);
- the Freiburg-LOB Corpus of British English (F-LOB).

These two corpora are mutually comparable [21] and contain texts published in 1961 and 1991, respectively.[3] Each corpus consists of approximately 1,000,000 words (500 texts of about 2000 running words each). The texts cover fifteen different text genres (Table 1), which could be further grouped into four, more generalised, categories: Press (A–C), Prose (D–H), Learned (J) and Fiction (K–R). The corpora were used in their untagged, raw text versions and parsed with

Table 1. Structure of the corpora

Category	Code	Genre	# texts
Press	A	Press: Reportage	44
	B	Press: Editorial	27
	C	Press: Review	17
General Prose	D	Religion	17
	E	Skills, Trades and Hobbies	38
	F	Popular Lore	44
	G	Belles Lettres, Biographies, Essays	77
	H	Miscellaneous	30
Learned	J	Science	80
Fiction	K	General Fiction	29
	L	Mystery and Detective Fiction	24
	M	Science Fiction	6
	N	Adventure and Western	29
	P	Romance and Love Story	29
	R	Humour	9

Connexor's Machinese Syntax parser in order to achieve consistent, highly accurate sentence splitting, tokenisation, lemmatisation and part-of-speech, syntactic and functional tagging.

3.2 Features

Twenty-three stylistic features (automatically extracted from the parser's output) were exploited (Table 2). Nine different POS tags were considered: N (noun)[4], A (adjective), PRON (pronoun), DET (determiner), ADV (adverb), V (verb)[5], CC (coordinative conjunction), CS (subordinate conjunction), PREP (preposition). Each POS tag was represented by two separate features: (1) the percentage of tokens tagged with that POS in each text; and (2) the average number of tokens tagged with that POS per sentence. Therefore, the last two rows in Table 2 account for 18 different features in total.

Connexor's Machinese Syntax parser was reported to achieve 99.3% accuracy in POS tagging on Standard Written English (benchmark from the Maastricht Treaty) [10]. Details of the parser's tokenisation and lemmatisation processes can be found in [30], while the details of passive and finite predicator marking procedures can be found in [31].

[3] Both corpora are publicly available as a part of the ICAME corpus collection at `http://www.hit.uib.no/icame`

[4] The ABBR morphological tag was counted as occurrence of a noun (N).

[5] The morphological tags ING (present participle) and EN (past participle) were counted as occurrences of a verb (V).

Table 2. Features (Key: c – total number of characters in a text; w – total number of words in a text; s – total number of sentences in a text; tokens – total number of tokens in a text; passive – total number of passive constructions in a text; active – total number of active constructions in a text; simple_s – total number of sentences in a text which have 1 finite predicator at the most; complex_s – total number of sentences in a text which have 2 or more finite predicators)

Feature	Code	Formula
Average sentence length	ASL	ASL = w/s
Coleman-Liau readability index	CLI	CLI = 5.89(c/w) - 29.5(s/w) - 15.8
Lexical richness	LR	LR = (unique lemmas)/(unique tokens)
Passive voice (%)	PASS	PASS = passive/(passive+active)
Sentence complexity	COMPL	COMPL = (simple_s)/(complex_s)
Part-of-Speech (%)	POS_per	POS_per = POS/tokens
Part-of-Speech (on average per sentence)	POS_av	POS_av = POS/s

3.3 Experimental Settings

First, we wanted to explore whether it is reasonable to expect that these 23 stylistic features would differ between the texts published in 1961 and those published in 1991, if we investigate them in each sub-genre (A–R) separately. Therefore, we conducted two sets of preliminary experiments. The Shapiro-Wilk's W test (offered by SPSS) was applied in order to determine whether the features follow the normal distribution across all thirteen genres in the two observed years. Additionally, the skewness and the existence of outliers was examined by using the box-plot. As the results demonstrated that the distribution of certain features in certain genres was significantly different from the normal distribution, we were not able to apply the t-test as a measure of statistical significance of the changes in all cases. In the cases where the distribution of the features was not approximately normal in both samples, we applied a non-parametric statistical test (Kolmogorov-Smirnov test).[6] The results of these statistical tests revealed significant differences in all 23 features, though in different subsets across the thirteen analysed genres. After these two preliminary experiments, which justified the use of the 23 initial features, we applied several Machine Learning algorithms in Weka Experimenter [34]: Support Vector Machines [25,17], Naïve Bayes [16], Logistic [9] and Simple Logistic [20,28] to classify the texts according to the year of publication – 1961 or 1991, using 5-fold cross-validation with 10 repetitions. The experiments were conducted separately for each text genre (A–P, excluding M)[7], thus enabling a comparison of diachronic changes in the period 1961–1991 across these thirteen text genres. We conducted three sets of experiments which differed in the subset of features they used:

– Experiment I: Using all 23 features;
– Experiment II: Using only the features marked as significant (at a 0.05 level of significance) by the statistical tests;

[6] We followed the same method for deciding on the appropriate statistical test as described in [31].

[7] Genres M and R were excluded from our analysis as they contain less than 10 texts in each corpus which is insufficient for the Machine Learning approach.

– Experiment III: Using only the features selected by the CfsSubsetEval attribute selection algorithm [15].

The comparison of the results obtained from these three experiments allowed us to further explore the potential of such a machine learning approach in diachronic studies. The goal was to answer the following questions:

1. Could the use of the statistical tests as a preprocessing (feature selecting) step improve the classification accuracy? (Comparison of the results of the first and second experiment).
2. Would the classification accuracy be improved if only the features selected by the CfsSubsetEval attribute selection algorithm were used? (Comparison of the results of the first and third experiment).
3. Would the CfsSubsetEval attribute selection algorithm be consistent with the results of the statistical tests? (Comparison of the results of the second and third experiment).

4 Results and Discussion

The results of the classification experiments are presented in Table 3. Column 'Exp.' contains the label of the experiment (I, II, III or III^{+}). While running the CfsSubsetEval attribute selection algorithm in the third experiment, it was noted that in the cases when it actually cannot find the best subset of features, the algorithm returns the first feature in the given list of all features as the best one. In those cases, the value of 'the merit of best subset found' is zero, while in the case of successful feature selection 'the merit of best subset found' has a value greater than zero. Therefore, in the first of these cases, an additional classification experiment was carried out – Exp. III^{+}– on the features selected by the CfsSubsetEval algorithm applied only on the subset of the initial set of features (those features reported as significant by the statistical tests). Columns 'NB', 'Log.', 'SLog.', 'SMO(s)', and 'SMO(n)' contain the F-measures of the five following classification algorithms: Naïve Bayes, Logistic, Simple Logistic, Support Vector Machines (with previous normalisation of the data), Support Vector Machines (with previous standardisation of the data) used in 5-fold cross-validation with 10 repetitions. Column '#feat.' contains the number of features used in each experiment. The highest obtained F-measure in each genre is shown in bold. As each genre contains the same number of texts published in 1961 and those published in 1991, the baseline accuracy in all genres could be considered to be 0.5. All comparisons between the results of experiment I and any other experiment were done pairwise using the paired t-test at a 0.05 level of significance. The statistically significant differences are shown in bold, with significantly lower results presented with an '*', and significantly higher results presented with a 'v'.

From the results presented in Table 3 it can be noted that in all cases where a statistically significant difference between the results of the first and second experiments was reported (genres B and N), the F-measure was lower in experiment I which uses all features. This indicates that the use of statistical tests

Table 3. Results of the classification experiments

Code	Genre	Exp.	NB	Log.	SLog.	SMO(n)	SMO(s)	#feat.
		I	0.69	0.74	0.81	0.79	**0.84**	23
A	Press: Reportage	II	0.76	0.81	0.78	0.78	0.77	8
		III	0.76	0.74	0.72	0.74	**0.73***	3
		I	0.62	0.68	0.74	0.75	0.66	23
B	Press: Editorial	II	**0.80v**	**0.81v**	0.79	0.77	0.78	4
		III	0.72	0.73	0.73	0.72	0.73	1
		I	0.61	0.72	0.73	0.74	0.76	23
C	Press: Review	II	0.69	0.71	0.72	0.73	0.71	2
		III	0.75	0.72	0.74	**0.78v**	0.75	1
		I	0.71	0.72	0.76	0.74	0.74	23
D	Religion	II	0.78	0.63	0.79	0.79	0.77	5
		III	**0.84**	0.80	0.81	0.81	0.82	1
		I	0.56	0.62	**0.66**	0.64	0.62	23
E	Skills, Trades and Hobbies	II	0.62	0.62	0.62	0.54	0.61	2
		III	0.61	0.61	0.61	0.59	0.61	1
		III$^+$	0.61	0.61	0.61	0.59	0.61	1
		I	0.45	0.54	0.65	0.61	0.61	23
F	Popular Lore	II	0.53	0.62	0.64	0.56	0.61	3
		III	0.59	**0.67**	**0.67**	0.50	0.66	1
		I	0.57	0.70	**0.72**	0.68	0.71	23
G	Belles Lettres, Biographies...	II	0.60	0.67	0.67	0.63	0.66	6
		III	0.63	0.62	0.61	0.62	0.64	2
		I	0.55	0.60	**0.65**	0.59	0.61	23
H	Miscellaneous	II	0.55	0.57	0.62	0.55	0.58	3
		III	0.63	0.62	0.62	0.64	0.61	1
		III$^+$	0.50	0.62	0.62	0.35	0.55	1
		I	0.65	**0.74**	0.71	0.69	**0.74**	23
J	Science	II	0.70	0.72	0.72	0.69	0.71	6
		III	0.71	0.73	0.72	0.69	0.73	3
		I	0.52	0.47	0.48	0.55	0.50	23
K	General Fiction	II	0.63	**0.65**	0.64	**0.65**	0.64	3
		III	0.54	0.55	0.56	0.43	0.51	1
		III$^+$	0.59	0.61	0.60	0.50	0.55	1
L	Mystery and Detective Fiction	I	0.35	0.57	**0.58**	0.54	0.56	23
		III	0.50	0.46	**0.58**	0.37	0.42	1
		I	0.69	0.57	0.55	0.58	0.45	23
N	Adventure and Western	II	0.70	0.71	0.68	0.69	**0.69v**	2
		III	0.69	0.67	0.68	**0.72**	**0.70v**	1
		I	0.60	0.56	0.54	0.56	0.51	23
P	Romance and Love Story	II	0.65	0.64	0.63	0.66	0.63	2
		III	0.63	0.63	0.62	0.58	0.59	1
		III$^+$	0.66	**0.68**	**0.68**	0.67	0.67	1

as a preprocessing step could enhance the diachronic classification of texts. In comparison with the results of the first experiment (Exp. I), the use of the CfsSubsetEval attribute selection algorithm (Exp. III) significantly increased the classification performance in two cases (genres C and N), while it significantly decreased the classification accuracy in genre A.

The use of the classification algorithms based on the logistic function (columns 'Log.' and 'SLog.') led to the highest F-measure in 9 genres (B, C, E–L, and P), while the classification algorithms based on Support Vector Machines (columns 'SMO(n)' and 'SMO(s)') led to the highest results only in 5 genres (A, C, J, K, and N). This might be interpreted as support for the idea that the diachronic change is best presented by the logistic function [3,19].

The results presented in Table 3 also indicate that the stylistic changes (in terms of these 23 initial features) were most pronounced in the Press category (genres A–C). Genres belonging to the Prose category underwent less extensive stylistic changes than those in the Press genre, as the F-measures are significantly lower in Prose than in the Press category. It can also be noted that within the Prose category, genre D (Religion) stands out in with the highest classification accuracy which leads to the conclusion that the stylistic changes were more pronounced in that genre than in the other four, thus making this genre an outlier in its category.

A more detailed analysis of features marked as significant by statistical tests and those returned by the CfsSubsetEval attribute selection algorithm as part of the best subset of features (Table 4) revealed that the features selected by the CfsSubsetEval algorithm are a subset of features marked as significant by statistical tests, in all cases where the CfsSubsetEval algorithm was successful ('the merit of the best subset found' above zero). In the cases when the CfsSubsetEval algorithm is unable to find the best subset of features, the algorithm selects the first feature in the given list of features, with 'the merit of best subset found' equal to zero.

Table 4. Selected features in experiments II, III and III$^+$

Genre	**Exp. II**	**Exp. III**	**Exp. III$^+$**
A	ASL, LR, PASS, COMPL, V_per, V_av, N_av, det_per	LR, COMPL, v_av	/
B	LR, det_per, prep_per, sc_av	LR	/
C	LR, COMPL	LR	/
D	n_av, prep_av, adj_per, adj_av, CLI	CLI	/
E	LR, CLI	CLI*	CLI*
F	pron_per, pron_av, CLI	CLI	/
G	LR v_per, n_per, sc_per, sc_av, CLI	CLI, n_per	/
H	ASL, det_av, prep_av	CLI*	ASL*
J	PASS, det_per, det_av, prep_per, prep_av, CLI	CLI, prep_per, det_per	/
K	COMPL, adv_av, cc_per	CLI*	adv_av
L	/	CLI*	/
N	n_per, CLI	CLI*	CLI*
P	LR, adv_per	CLI*	LR*

The results of experiment III$^+$ were found to be significantly better than those of experiment III when CfsSubsetEval:

– fails to find the best subset of the initial features (selected feature in column 'Exp. III' in Table 4 is marked by an '*'),
– succeeds in finding the best subset of those features reported as significant by statistical tests (selected feature in column 'Exp. III$^+$' in Table 4 is not marked by an '*').

Although in our data set we found only one such case (genre K), we could still say that the safest way to use the CfsSubsetEval attribute selection algorithm would be to apply it only to a subset of initial features (only those features which were marked as significant by the statistical tests).

5 Conclusions

The results presented in this study indicated that the stylistic diachronic changes of written British English in the period 1961–1991 were significantly more pronounced in the Press category than in three other text categories (Table 3). They also demonstrated that the genres within the same broad text category are very heterogeneous. In each of them, different groups of features underwent a significant diachronic change (Table 4) and the extent of those changes differed significantly across them (Table 3). The results also indicated that lexical richness (LR) and the Coleman-Liau readability index were the features which significantly changed in most of the investigated genres (Table 4).

On the basis of the comparison of the results of different experiments, we can conclude that the use of the statistical tests as a preprocessing (feature selection) step, significantly increases the classification accuracy in several cases, while in others it does not have any significant influence. Therefore, we suggest the use of the statistical tests as a preprocessing step in other diachronic text classification tasks. When compared with the CfsSubsetEval attribute selection algorithm, the statistical test achieved significantly better or equal performance (with the only exception in genre D, for the Naïve Bayes classification algorithm). In most cases, this was due to the fact that the CfsSubsetEval algorithm selects the first feature in the given list of features in the cases when it is not able to find a subset with 'the merit of best subset found' greater than zero. The use of the CfsSubsetEval attribute selection algorithm on the subset of features previously selected by the statistical tests, significantly improves the classification accuracy (genre K) or it leaves it unchanged. The statistical tests when used in the preprocessing step on their own, either significantly improve the classification accuracy (genres B and N) or they do not lead to any significant differences. Therefore, we suggest either the use of the statistical tests on their own or the combination of the CfsSubsetEval attribute selection algorithm with them in the preprocessing step of diachronic text classification.

Most importantly, the presented study demonstrated various possibilities that the machine learning approach can offer to the investigation of language change. By partially automating the process, it can speed up and facilitate the initial phases of language change studies, by providing a broad overview of possible changes and selecting the most important features from a potentially large initial set, which would be the subject of closer investigation. A machine learning approach could also offer an easier comparison of diachronic changes across different genres, languages and language varieties.

References

1. Adolph, R.: The Rise of Modern Prose Style. M.I.T. Press, Cambridge (1966)
2. Aldrich, J., Nelson, F.: Linear probability, logit, and probit models. Quantitative applications in the social sciences. Sage, London (1984)
3. Altmann, G., von Buttlar, H., Rott, W., Strau, U.: A law of change in language. In: Brainerd, B. (ed.) Historical Linguistics, pp. 104–115. Brockmeye, Bochum (1983)

4. Bennett, J.R.: Prose Style: A Historical Approach through Studies. Chandler, San Francisco (1971)
5. Biber, D.: Investigating Macroscopic Textual Variation through Multifeature/Multidimensional Analyses. Linguistics 23, 337–360 (1985)
6. Biber, D.: Variation across speech and writing. Cambridge University Press, Cambridge (1988)
7. Biber, D., Finegan, E.: An Initial Typology of English Text Types. In: Aarts, J., Meijs, W. (eds.) Corpus Linguistics H: New Studies in the Analysis and Exploitation of Computer Corpora, pp. 19–46. Rodopi, Amsterdam (1986)
8. Biber, D., Finegan, E.: Drift and the evolution of English style: A history of three genres. Language 65, 487–517 (1989)
9. le Cessie, S., van Houwelingen, J.: Ridge Estimators in Logistic Regression. Applied Statistics 41(1), 191–201 (1992)
10. Connexor: Machinese language analysers (2006)
11. Corpas Pastor, G., Mitkov, R., Afzal, N., Pekar, V.: Translation Universals: Do they exist? A corpus-based NLP study of convergence and simplification. In: Proceedings of the AMTA, Waikiki, Hawaii (2008)
12. Geisler, C.: Relativization in Ulster English. In: Poussa, P. (ed.) Relativisation on the North Sea Littoral (LINCOM Studies in Language Typology 07), pp. 135–146. Lincom Europa, München (2002)
13. Geisler, C.: Statistical reanalysis of corpus data. ICAME Journal 32, 35–46 (2008)
14. Gordon, I.A.: The Movement of English Prose. Indiana University Press, Bloomington (1966)
15. Hall, M.A., Smith, L.A.: Practical feature subset selection for machine learning. In: McDonald, C. (ed.) Computer Science 1998 Proceedings of the 21st Australasian Computer Science Conference, ACSC 1998, pp. 181–191. Springer, Berlin (1998)
16. John, G.H., Langley, P.: Estimating Continuous Distributions in Bayesian Classifiers. In: Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence, pp. 338–345 (1995)
17. Keerthi, S.S., Shevade, S.K., Bhattacharyya, C., Murthy, K.R.K.: Improvements to Platt's SMO Algorithm for SVM Classifier Design. Neural Computation 13(3), 637–649 (2001)
18. Kroch, A.: Function and grammar in the history of English: Periphrastic "do". In: Fasold, R. (ed.) Language Change and Variation, pp. 133–172. Benjamins, Amsterdam (1989)
19. Kroch, A.: Reflexes of grammar in patterns of language change. In: Language Variation and Change, vol. 1, pp. 199–244 (1989)
20. Landwehr, N., Hall, M., Frank, E.: Logistic Model Trees. Machine Learning 59, 161–205 (2005)
21. Leech, G., Smith, N.: Extending the possibilities of corpus-based research on English in the twentieth century: a prequel to LOB and FLOB. ICAME Journal 29, 83–98 (2005)
22. Leech, G., Smith, N.: Recent grammatical change in written English 1961-1992: some preliminary findings of a comparison of American with British English. In: Renouf, A., Kehoe, A. (eds.) The Changing Face of Corpus Linguistics, pp. 186–204. Rodopi, Amsterdam (2006)
23. Mair, C., Hundt, M., Leech, G., Smith, N.: Short term diachronic shifts in part-of-speech frequencies: a comparison of the tagged LOB and F-LOB corpora. International Journal of Corpus Linguistics 7, 245–264 (2002)
24. Mair, C., Leech, G.: Current change in English syntax. In: Aarts, B., MacMahon, A. (eds.) The Handbook of English Linguistics, ch. 14. Blackwell, Oxford (2006)

25. Platt, J.C.: Fast Training of Support Vector Machines using Sequential Minimal Optimization. In: Schoelkopf, B., Burges, C., Smola, A. (eds.) Advances in Kernel Methods – Support Vector Learning. The MIT Press, London (1998)
26. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys 34(1), 1–47 (2002)
27. Senter, R.J., Smith, E.A.: Automated readability index. Tech. rep., University of Cincinnati. Ohio, Cincinnati (1967)
28. Sumner, M., Frank, E., Hall, M.: Speeding up Logistic Model Tree Induction. In: Jorge, A.M., Torgo, L., Brazdil, P.B., Camacho, R., Gama, J. (eds.) PKDD 2005. LNCS (LNAI), vol. 3721, pp. 675–683. Springer, Heidelberg (2005)
29. Tukey, J.: Exploratory data analysis. Addison-Wesley, Reading (1977)
30. Štajner, S., Mitkov, R.: Diachronic Stylistic Changes in British and American Varieties of 20th Century Written English Language. In: Proceedings of the RANLP 2011 Workshop "Language Technologies for Digital Humanities and Cultural Heritage", pp. 78–85 (2011)
31. Štajner, S., Mitkov, R.: Diachronic Changes in Text Complexity in 20th Century English Language: An NLP Approach. In: Calzolari, N., Choukri, K., Declerck, T., Dogan, M.U., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S. (eds.) Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC 2012), Istanbul, Turkey (May 2012)
32. Westin, I.: Language Change in English Newspaper Editorials. Rodopi, Amsterdam (2002)
33. Westin, I., Geisler, C.: A multi-dimensional study of diachronic variation in British newspaper editorials. ICAME Journal 26, 133–152 (2002)
34. Witten, I.H., Frank, E.: Data mining: practical machine learning tools and techniques. Morgan Kaufmann Publishers (2005)

Factored Semantic Sequence Kernel for Sentiment Polarity Classification

Luis Trindade, Hui Wang, William Blackburn, and Niall Rooney

University of Ulster, School of Computing and Mathematics
Faculty of Computing and Engineering, UK
{Trindade-L,H.wang,Wt.blackburn,Nf.rooney}@ulster.ac.uk

Abstract. Sentiment analysis is an area of research that has gained considerable attention in recent years due to the increasing availability of opinionated information online. The majority of the work in sentiment analysis considers the polarity of word terms rather than the polarity of specific senses of the word but different senses of a word can have different opinion-related properties. In order to address this issue we consider novel semantic features of words in the context of a sentence. We take a sentence as a sequence of words augmented with features based on word sense disambiguation and sentiment lexicons with sense specific opinion-related properties. We then use a factored version of the sequence kernel in a support vector machine, and apply it to sentiment classification of sentences. We evaluate this sentiment analysis methodology on three publicly available corpuses. We also evaluate the effectiveness of several publicly available sense specific polarity lexicons and combinations. Experiments show that our factored approach offers improvements over the surface words baseline and other state-of-the-art kernels.

Keywords: Information Retrieval, Social Media, Sentiment Analysis, Opinion Mining, Polarity Classification, Kernel Methods, Word Sense Disambiguation.

1 Introduction

Textual information can easily be seen as consisting of facts and/or opinions. Facts are an objective expression about entities, events and their properties, and opinions are a subjective expression of people's sentiments, appraisals or feelings toward entities, events and their attributes [1]. Determining the opinion contained within a piece of text is the aim of sentiment analysis (or opinion mining), which is assisted by natural language processing (NLP), information retrieval (IR) and computational linguistics (CL).

Sentiment analysis has gained considerable attention in recent years, partially due to the many practical applications it supports. Examples include helping companies and organizations find customer opinions of commercial products or services; tracking opinions in online forums, blogs and social networks; or helping individuals decide on which product to buy or which movie to watch. The growing demand for automated sentiment analysis is supported by an increasing amount and availability of opinionated information online, mainly due to social media websites and e-commerce services [2], [3]. Some of the most common tasks in sentiment analysis include:

- Subjectivity Classification: Determining if a given piece of text is factual, describing a given situation or event without expressing an opinion, or opinionated [4]. This equates to a binary text classification task of classifying a given piece of text as either subjective or objective;
- Polarity Classification: Determining if a given piece of text expresses a positive, negative or neutral (or both) opinion or attitude [4];
- Polarity Intensity Classification: Determining the direction (polarity) and intensity of an opinion in a given piece of text. The intensity is usually in the form of a scale of star ratings [5];
- Feature/Aspect-based Sentiment Analysis: Discovering relevant aspects of entities in a given piece of text and the opinions or sentiments they express, and then determining the polarity of these opinions [6]. The features or aspects can be attributes or components of an object or entity.

These tasks can also be performed in combination, for example, one can start by classifying expressions as being either objective or subjective in nature; expressions classified as subjective can then be further classified as neutral or polar; and finally polar expressions can be classified as either positive, negative or both. Moreover polarity classification can be performed at various levels: word-level, phrase-level, sentence-level and document-level. Classifying the sentiment of documents is a very different task from recognizing the contextual polarity of words and phrases, where there is very little contextual information. This paper focuses on polarity classification at the sentence (and phrase) level.

Support vector machine (SVM) is a popular kernel method for text classification [7]. Kernel methods are based on the use of a kernel function, which allows the mapping of data from the original feature space into a higher dimensional linear space. The comparison of data can thus be done by computing the inner product in the higher dimensional space, albeit implicitly through the so-called kernel trick. The choice of kernel function depends on the application. Kernel methods can be applied to complex objects such as sequences, images, graphs and textual documents [8], based on an appropriate kernel function. This makes them well suited for structured NLP [9] and they have been applied to various tasks such as Question Answering, Summarization and Recognizing Textual Entailment. This paper focuses on the sequence kernel (SK), which has been successfully employed for sentiment analysis tasks [10], [11]. The SK aggregates the frequency of matching subsequences between two sequences. A common approach is to consider the words, or terms of a sentence as the individual objects in the sequence.

Despite recent efforts [12], [13], [14], [15], [16], [17] the majority of work in sentiment analysis still considers the polarity of word terms rather than the polarity of specific senses of the word. It is clear that different senses of a word can have different opinion-related properties, for example, the adjective "hysterical" can mean, upset or scared (e.g. she was frightened and out of control, she was hysterical) but it can also mean excited or hilarious (e.g. the comedian was so funny, he was hysterical). In order to address this issue we consider novel semantic features of words in the context of a sentence. To be precise we take a sentence as a sequence of words augmented with features based on word sense disambiguation (WSD) and sentiment lexicons with sense specific opinion-related properties. Namely the surface words with the corresponding WordNet [18] senses (defined as a concatenation of the word's lemma, its reduced part of speech (POS) tag and its sense number, see section 3.3) and their corresponding polarity. We consider individual features as well as a factored representation of features. We evaluate the quality of such feature sequences on two binary text classification tasks, the determination of whether a sentence is subjective or objective in nature (subjectivity classification); and whether a sentence expresses a positive or negative sentiment (sentiment polarity classification). Our evaluations show that the factored sequence kernel can be quite effective for the tasks of polarity and subjectivity classification, and performs better than other state-of-the-art techniques.

To the best of our knowledge no previous study has considered a factored version of the sequence kernel for sentiment classification, nor used sequences of WordNet senses and sequences of polarities.

The rest of this paper is structured as follows. Section 2 gives a brief introduction to sequence kernels and their factorization. Section 3 describes the text classification tasks as well as the experimental setting considered for evaluation of the kernel and features. Section 4 reports the experimental results. Section 5 concludes this paper with a discussion of the results and possible future work.

2 Sequence Kernels

Some of the first kernel methods represented documents as a bag-of-words until Lodhi et al. [19] developed what would become known as sequence kernels (SKs) (or string kernels). Rather than making use of features such as word frequencies, SKs use the number of all possible ordered subsequences of characters contained in a text document. Although such mechanisms reflect the structure of sentences, it was still not the ideal solution for comparing sentences within large corpora, due to the computational complexity of the kernel. Due to this downside, further developments extended the idea of SKs to process documents as sequences of words (word sequence kernel) increasing the number of symbols to consider but greatly reducing the average number of symbols per document. Since the kernel's dynamic programming algorithm depends only on sequence length this increases its computational efficiency significantly [20], [21]. This idea was extended to allow the comparison of complex objects, such as fixed length vectors of features, through a weighted factored representation [10]. Cancedda et al. in the latter study have shown that there are advantages in using the factored kernel, compared to a linear combination of kernels or a single kernel applied to each element in the vector separately.

Definition 1. Let I be a feature subsequence space of sequences set S, and $L_n(x)$ be the set of subsequences of x of size n, $G(s_x)$ be the number of gaps in the subsequence s_x, and λ be the gap penalization factor, so for x and y belonging to S and $u \in I$, the *gap-weighted subsequence kernel* of order n can also be defined as an inner product of vectors of $\phi_u(x)$:

$$sk(x,y) = \left(\phi(x), \phi(y)\right) = \sum_{u \epsilon I} \phi_u(x)\, \phi_u(y) \tag{1}$$

where

$$\phi_u(x) = \sum_{S_x \epsilon L_n(x)} 1(S_x = u)\lambda^{G(S_x)} \tag{2}$$

When λ is 1 there is no penalization of gaps, meaning every subsequence will contribute equally whether the elements in the sequence are contiguous or not. As λ decreases the gap penalization increases, meaning that as its value gets very close to 0 the kernel will be reduced to counting the number of consecutive subsequences.

A dynamic programming algorithm used for computing SK, requiring $O(n|x||y|)$ arithmetic operations, is presented in [19]. Note that when computing the order n kernel sk_n, this algorithm computes all kernels sk_i for $i \leq n$ allowing the computation of an $up\ to\ n$ kernel $sk = \sum_{i=1}^{n} sk_i$.

2.1 Factored Sequence Kernel

This paper considers more complex structures as the objects in the sequences, such as fixed length vectors of associated features/factors. One efficient way to combine the results of such kernels of these features is the factored combination of kernels.

Definition 2. Let u be defined as a tuple of p features $\{u^{(d)}\}_{d=1:p} \in \Sigma_d$, and the weights $w_d \geq 0$ control the relative contribution of the different factors in the global kernel. The factored SK is defined as a soft-matching kernel, where for a pair of objects (u, v) the kernel k_Σ is defined as

$$k_\Sigma(u,v) = \sum_{d=1}^{p} w_d 1(u^{(d)} = v^{(d)}) \tag{3}$$

that is

$$sk_n^{fact}(x,y) = \sum_{S_x \epsilon L_n(x)} \sum_{S_y \epsilon L_n(y)} \lambda^{G(S_x)+G(S_y)} \prod_{i=1}^{n} \sum_{d=1}^{p} w_d \left(S_x^{(d)}[i] = S_x^{(d)}[i]\right) \tag{4}$$

Note that whereas the linear combination of SK has p times the complexity of the single kernel version, the factored version has complexity $O((p+n)|x||y|)$ which can be quite significant for large sequences. Finally note that since the number of subsequences in a sequence increases with its length, the value of the kernels should be normalized to compensate for this effect. Then

$$\widehat{sk}(s,y) = \frac{sk(x,y)}{\sqrt{sk(x,x)sk(y,y)}} \tag{5}$$

3 Methodology

This paper presents a novel set of feature sequences based on word sense disambiguation (WSD) and sentiment lexicons with sense specific opinion-related properties. We consider a tridimensional factored representation in a similar fashion to Cancedda et al. [10], where each word in the sequences is also accompanied by its associated lemma and POS. In this paper we explore a novel set of features: the surface words, their WordNet sense, and their polarity:

$$x_i = \left[x_i^{(d=1)}, x_i^{(d=2)}, x_i^{(d=3)}\right] = [word_i, WordNet\ sense_i, polarity_i]$$

Table 1. Tridimensional factored representation of an example sentence "This movie is not good"

d \ i	1	2	3	4	5
1	This	movie	is	not	good
2	this#ND	movie#n#1	is#v#1	not#r#1	good#a#1
3	O	O	O	N	P

This allows the kernel to not only match the surface words but also senses of the words as well as the polarity of the word senses. The idea is that having a set of features that is tailored for the task will increase the performance of the system.

3.1 Sentiment Polarity Corpuses

We evaluate our approach on two binary text classification tasks, the determination of whether a sentence is subjective or objective in nature (subjectivity classification); and whether a sentence expresses a positive or negative sentiment (sentiment polarity classification). We conduct a series of 10-fold cross-validation tests on three publicly available corpuses. Namely:

- *Movie Reviews Polarity corpus* (sentence polarity dataset v1.0) [5] – This corpus contains 5331 positive and 5331 negative processed sentences/snippets taken from several movie reviews.
- *SemEval-2007 Affective Task corpus* [22] – This corpus contains 500 positive (valence ≥ 0) and 500 negative (valence < 0) news headlines, extracted from news web sites (such as Google news and CNN) and/or newspapers.
- *Movie Reviews Subjectivity corpus* (subjectivity dataset v1.0) [4] – This corpus contains 5000 subjective and 5000 objective processed sentences taken from several movie reviews.

3.2 Word Sense Disambiguation

We perform WSD with a WordNet-based method (WordNet::SenseRelate::AllWords [23]) in order to obtain the WordNet sense corresponding to the words in the corpus. We choose the same combination of parameters that achieved the best result reported in [23], using the Lesk measure [24] as the similarity function, which tends to result in much higher recall, (since it is able to measure the similarity between words with any

POS); and a window size of 15 (the number of words, to be taken into consideration when performing the WSD). In order to increase the compatibility of the sentences in the corpuses with WordNet::SenseRelate::AllWords, we replace contracted expressions with their full version (e.g. the term "won't" is replaced by "will not").

3.3 Sentiment Lexicons

Despite recent efforts, most work still makes use of the words' prior polarity in order to classify the polarity of sentences or documents. Often overlooking the fact that the polarity of a word depends on the context in which it is expressed [25]. In order to address this issue this paper makes use of several WordNet-based polarity lexicons that take into account the polarity of particular senses of the words, namely, SentiWordNet [2][26], Q-WordNet [27] Micro-WNOp [28].

This paper assigns each WordNet sense a value based on an aggregated score (A-score = P-score – N-score) similar to the approach taken by Agerri et al. [27]. Namely assigning the following values for the overall polarity:

- "**P**" to positive senses (A-score > 0) – e.g. true#a#2 which has a P-score of 1 and a N-score of 0;
- "**N**" to negative senses (A-score < 0) – e.g. cynical#a#1 which has a P-score of 0 and a N-score of 1; and
- "**O**" to objective and neutral senses (A-score = 0) – e.g. real#a#7 which has a P-score of 0 and a N-score of 0.

We also consider an alternative representation by assigning a value of B for senses that can have both polarities (A-score = 0, P-score $\neq$ 0, N-score $\neq$ 0, and P-score = N-score) – e.g. literal#a#1 which has a P-score of 0.25 and a N-score of 0.25. This alternative representation seems to have little to no effect in preliminary experiments, as such it is not considered for the final experiments.

We analyze the effectiveness and coverage of the polarities obtained from the different sentiment lexicons, by themselves and in combination as depicted in Table 2.

Table 2. Sentiment lexicons considered

ID	Lexicon	Senses
L1	Micro-WNOp (MWN)	2800
L2	Q-WordNet (QWN)	15511
L3	SentiWordNet (SWN)	49447
CL1	Micro-WNOp + Q-WordNet (MWN + QWN)	18062
CL2	Micro-WNOp + SentiWordNet (MWN + SWN)	51001
CL3	Q-WordNet + SentiWordNet (QWN+SWN)	60738
CL4	Micro-WNOp + Q-WordNet + SentiWordNet (MWN+QWN+SWN)	62194

The polarity lexicons are in the format Lemma#ReducedPart-of-SpeechTag# SenseNumber Polarity {P or N or O (or B)}. Note that the combined lexicon QWN+SWN (CL3), for example, does not have the same meaning as SWN+QWN. QWN+SWN is generated by using the polarities in Q-WordNet as a starting point and then adding to it the polarities extracted from SentiWordNet for words that are present in

SentiWordNet and not in Q-WordNet. This means that there are other possible combinations that are not featured in this table, since they proved to be less effective The most efficient combinations are those that give priority to the most finegrained and smallest lexicons especially when considering SWN, for example QWN (15511) + SWN (49447) results in 60738 unique WordNet sense polarities in total. This might be due to the fact that SWN was not manually annotated and some senses are misclassified, so by giving priority to the senses in MWN and QWN we reduce this negative influence.

3.4 Support Vector Machine

The SVM implementation chosen to run the classification tasks is libSVM [29]. In order to investigate the effects of the different features considered, we use the following combinations of weight vectors $w = \left[w_1^{(d=1)}, w_2^{(d=2)}, w_3^{(d=3)}\right]$:

- For the single feature experiments: $w = [1,0,0]$, $w = [0,1,0]$, $w = [0,0,1]$;
- for the two factor experiments: $w = [1,1,0]$, $w = [1,0,1]$, $w = [1,2,0]$, $w = [2,1,0]$, $w = [2,2,0]$, $w = [1,0,2]$, $w = [2,0,1]$, $w = [2,0,2]$;
- and for the full factored experiments: $w = [1,1,1]$, $w = [1,1,2]$, $w = [2,1,1]$, $w = [1,2,1]$ and w = [2,4,1]

For the SK's parameters n and λ, we use values from the sets {2, up to 2, 3, up to 3} and {0.1, 0.5, 1} respectively.

4 Experimental Evaluation

We evaluate the impact of the proposed methodology for sentiment classification tasks. Note that the results are for a combination of parameters where a given combination is denoted by P_i in Table 3.

Table 3. Parameter combinations

n \ λ	**0.1**	**0.5**	**1**
2	P_1	P_5	P_9
***up to* 2**	P_2	P_6	P_{10}
3	P_3	P_7	P_{11}
***up to* 3**	P_4	P_8	P_{12}

We start by evaluating the performance of the different sentiment lexicons considered, using only the polarities generated by each corresponding lexicon (Table 4). We found that the combined lexicon CL4 comprising Micro-WNop, Q-WordNet and SentiWordNet achieves the best performance in most cases. Since it also has the largest coverage, we choose it for the remaining tests.

Table 4. Sentiment lexicon evaluation

Lexicon	Movie Polarity	SemEval News	Movie Subjectivity
L1	50.81	54.70	53.00
L2	53.19	55.40	53.42
L3	56.27	58.90	63.52
CL1	53.73	56.40	53.97
CL2	56.19	59.20	**63.63**
CL3	57.35	59.60	62.59
CL4	**57.51**	**60.10**	63.17

We then evaluate the performance of the various features separately, as well as the SK parameter combinations (Table 5). Please note that these experiments equate to the simple SK where: $w = [1,0,0]$ means only the surface words contribute, $w = [0,1,0]$ means only the WordNet senses contribute and $w = [0,0,1]$ means that only the polarities contribute. We found that WSD provides an improvement over the surface words for the SemEval News and Movie Reviews Polarity corpuses. However for the Movie Reviews Polarity corpus there is a slight decline from using the WordNet senses. We also found that the polarity sequences are not very accurate, being much lower in accuracy than the surface words or WordNet senses.

Table 5. Single feature SK parameter evaluation

	Movie Polarity			SemEval News			Movie Subjectivity		
P \ w	[1,0,0]	[0,1,0]	[0,0,1]	[1,0,0]	[0,1,0]	[0,0,1]	[1,0,0]	[0,1,0]	[0,0,1]
P_1	76.22	75.32	57.48	68.00	69.40	58.10	89.63	89.77	63.04
P_2	76.08	75.43	57.40	68.50	**71.50**	58.10	89.79	89.73	63.13
P_3	76.11	75.24	**57.51**	68.90	70.10	57.00	89.86	89.99	63.07
P_4	**76.33**	75.70	57.48	67.00	69.70	57.90	**89.99**	89.93	**63.17**
P_5	76.21	75.59	57.39	68.30	70.40	57.30	89.72	89.80	63.09
P_6	76.30	75.55	57.39	68.00	69.00	57.30	89.68	89.70	63.07
P_7	76.19	75.41	57.34	67.00	70.10	57.10	89.83	89.73	63.01
P_8	76.00	75.33	57.44	67.30	69.10	57.10	89.80	89.92	63.02
P_9	76.21	75.53	57.28	67.60	69.50	58.20	89.83	**90.05**	62.98
P_{10}	76.27	**76.05**	57.30	**69.00**	70.90	58.50	89.64	89.97	63.00
P_{11}	76.00	75.43	57.36	67.30	69.90	59.30	89.69	89.86	63.03
P_{12}	75.92	75.40	57.41	67.80	70.70	**60.10**	89.93	89.87	62.95

Next we evaluate the performance of the factored SK using all of our features combined (Table 6) and using the best performing parameter combinations identified in Table 5. We found that for the SemEval News corpus, there is an improvement by using the factored features, however this results is still lower than when using the

WordNet senses by themselves. As for the Movie Reviews Polarity and Subjectivity corpuses there is a decline in performance when compared to only using the surface words.

Table 6. Full factored evaluation (Word + WordNet + Polarity)

	Movie Polarity		SemEval News		Movie Subjectivity	
w \ *P*	P_4	P_{10}	P_2	P_{10}	P_4	P_9
[1:1:1]	71.76	71.90	68.00	69.30	86.89	86.88
[1:1:2]	70.76	70.80	68.20	68.10	86.21	86.16
[1:2:1]	71.84	72.15	69.30	68.70	86.98	87.10
[2:1:1]	71.88	72.11	69.20	69.60	87.12	87.13
[2:4:1]	**72.75**	71.76	**70.70**	69.70	87.56	**87.60**

We postulate that the polarity feature is actually hindering the overall performance of the factored SK. In order to verify this we further evaluate the performance of the factored SK using combinations of two features only (Table 7). We found that using only the surface words combined with the WordNet senses improves the performance for the Movie Review Polarity and Subjectivity corpuses. However for the SemEval News corpus, despite a small improvement over the surface words, it still does not outperform the WordNet senses or even the full factored features.

Table 7. Evaluation of the factored sk using combinations of two features

	Movie Polarity		SemEval News		Movie Subjectivity	
w \ *P*	P_4	P_{10}	P_2	P_{10}	P_4	P_9
[1:0:1]	69.24	69.43	65.70	65.30	84.91	84.95
[1:0:2]	68.10	67.91	64.00	63.60	83.94	84.01
[2:0:1]	70.47	70.34	66.10	67.00	85.74	85.70
[1:1:0]	76.30	76.55	69.10	69.20	90.17	89.96
[1:2:0]	76.39	76.84	69.70	**69.80**	90.21	90.20
[2:1:0]	**76.93**	76.65	68.90	68.90	90.10	**90.24**

Finally by way of comparison we evaluate the performance of our approach and other popular kernels for sentiment classification tasks. Namely the Linear kernel with Bag of Words (BoW), the Subset Tree kernel [30] (SST) with Syntactic Parse trees and the Partial Tree kernel [31] (PT) with Dependency Parse trees. We found that our approach outperforms all the other kernels considered.

Table 8. Comparison of our approach and other popular kernels for sentiment classification tasks

Methodology	Movie Polarity	SemEval	Movie Subjectivity
Linear, BoW	50.47	54.10	53.71
SST, Syntactic Parse trees	71.70	62.60	89.01
PT, Dependency Parse trees	74.86	65.20	88.84
Factored Sequence Kernel, WSD	**76.93**	**71.50**	**90.24**

5 Conclusions and Outlook

The majority of the work in sentiment analysis considers the polarity of word terms rather than the polarity of specific senses of the word. It should be clear that different senses of a word can have different opinion-related properties. This work addressed this issue, by considering a sentence as a sequence of words augmented with their WordNet sense and sense specific opinion-related properties.

We evaluated three sentiment lexicons and four combinations of these. We found that the combined lexicon CL4 comprising Micro-WNop, Q-WordNet and Senti-WordNet achieves the best performance for the polarity classification tasks. For the subjectivity classification task, CL2 achieves the best performance but CL4 is very close. This might be due to the polarity sequences being more relevant for polarity classification tasks rather than subjectivity classification tasks.

Despite WSD being reportedly only about 50-70% accurate [14], [23], [17] the experimental evaluation shows that performing WSD provides an improvement over the surface words for the SemEval News and Movie Reviews Subjectivity corpuses, from 69% up to 71.5% and from 89.99% up to 90.05% respectively. As for the Movie Reviews Polarity corpus there is a slight decline from 76.33% down to 76.05%.

In the full factored (words + WordNet sense + polarity) evaluation, for the SemEval News corpus, there is an improvement by using the factored features (from 69% up to 70.7%), however this result is low when compared to only using the WordNet senses (71.5%); as for the Movie Review Polarity and Subjectivity corpuses there is substantial decline, from 76.33% down to 72.75% and from 89.99% down to 87.6% respectively. Note however, that the polarity sequences are not very accurate, being much lower in accuracy than the surface words or WordNet senses. For the Movie Reviews Polarity corpus there is an 18.82% decrease in accuracy, for the SemEval corpus there is an 11.4% decrease in accuracy and finally for the Movie Reviews Subjectivity corpus there is a decrease of 26.88% in accuracy. This suggests that the polarity sequences may not be a useful feature for discovering similarity (particularly for the subjectivity classification task). As there are only three (or four) forms of polarity value considered, subsequences of this feature may convey very little information for distinguishing between a sentence that is positive and one that is negative. For example two positive sentences may share little similar subsequences based on the polarity feature. Due to the lower accuracy the polarity sequences achieve, we postulate that they are actually hindering the overall performance when used in the factored SK. In order to investigate this we evaluated the performance of the factored SK using combinations of two features. We found that this improved the performance for the Movie Review Polarity and Subjectivity corpuses, when combining the surface words and the WordNet senses, from 76.33% up to 76.93% and 89.99% up to 90.24% respectively. For the SemEval News corpus, despite a small improvement over the surface words (from 69% up to 69.8%), it still does not surpass the performance of using only the WordNet senses (71.5%) or even the full factored performance (70.7%).

Our evaluations confirm previous findings that WSD offers improvements for sentiment classification tasks, however since the WSD is an intermediate task, disambiguation errors can affect the quality of the corresponding sense specific opinion-related properties and thus the classification quality. Finally our results show that the factored sequence kernel can be quite effective for sentiment classification tasks, and performs better than other state-of-the-art kernels.

Further work will possibly include: applying the methodology to other corpuses; exploring other polarity representations (e.g. ranking); handling polarity modification features (e.g. negation, intensification) in order to improve the polarity sequences; applying the methodology to 3-class and 5-class polarity classification problems; and expanding the methodology to classify documents.

References

1. Liu, B.: Sentiment Analysis and Subjectivity. In: Handbook of Natural Language Processing (2010)
2. Esuli, A., Sebastiani, F.: Sentiwordnet: a Publicly Available Lexical Resource for Opinion Mining. In: Proceedings of LREC, p. 417. Citeseer (2006)
3. Pang, B., Lee, L.: Opinion Mining and Sentiment Analysis. Foundations and Trends in Information Retrieval 2(1-2), 1–135 (2008)
4. Pang, B., Lee, L.: A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts. In: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics, p. 271. Association for Computational Linguistics (2004)
5. Pang, B., Lee, L.: Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, p. 115. Association for Computational Linguistics (2005)
6. Hu, M., Liu, B.: Mining and Summarizing Customer Reviews. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, p. 168. ACM (2004)
7. Sebastiani, F.: Machine Learning in Automated Text Categorization. ACM Computing Surveys (CSUR) 34(1), 1–47 (2002)
8. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press (2004)
9. Suzuki, J., Hirao, T., Sasaki, Y., Maeda, E.: Hierarchical Directed Acyclic Graph Kernel: Methods for Structured Natural Language Data. In: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics, vol. 1, pp. 32–39. Association for Computational Linguistics (2003)
10. Cancedda, N., Mahe, P.: Factored Sequence Kernels. Neurocomputing 72(7-9), 1407–1413 (2009)

11. Trindade, L.A., Wang, H., Blackburn, W., Rooney, N.: Text Classification Using Word Sequence Kernel Methods. In: 2011 International Conference on Machine Learning and Cybernetics (ICMLC), p. 1532. IEEE (2011)
12. Akkaya, C., Wiebe, J., Conrad, A., Mihalcea, R.: Improving the Impact of Subjectivity Word Sense Disambiguation on Contextual Opinion Analysis. In: Proceedings of the Fifteenth Conference on Computational Natural Language Learning, p. 87. Association for Computational Linguistics (2011)
13. Balamurali, A., Joshi, A., Bhattacharyya, P.: Robust Sense-Based Sentiment Classification. In: ACL HLT 2011, p. 132 (2011)
14. Balamurali, A.R., Joshi, A., Bhattacharyya, P.: Harnessing Wordnet Senses for Supervised Sentiment Classification. In: Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, July 27-31, p. 1081. Association for Computational Linguistics (ACL), Edinburgh (2011)
15. Carrillo De Albornoz, J., Plaza, L., Gervás, P.: A Hybrid Approach to Emotional Sentence Polarity and Intensity Classification. In: Proceedings of the Fourteenth Conference on Computational Natural Language Learning, p. 153. Association for Computational Linguistics (2010)
16. Martın-Wanton, T., Balahur-Dobrescu, A., Montoyo-Guijarro, A., Pons-Porrata, A.: Word Sense Disambiguation in Opinion Mining: Pros and Cons. Special Issue: Natural Language Processing and Its Applications, 119 (2010)
17. Rentoumi, V., Giannakopoulos, G., Karkaletsis, V., Vouros, G.A.: Sentiment Analysis of Figurative Language Using a Word Sense Disambiguation Approach. In: Proc. of the International Conference RANLP, p. 370 (2009)
18. Miller, G.A.: Wordnet: a Lexical Database for English. Communications of the ACM 38(11), 39–41 (1995)
19. Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., Watkins, C.: Text Classification Using String Kernels. Journal of Machine Learning Research 2(3), 419–444 (2002)
20. Bunescu, R., Mooney, R.: Subsequence Kernels for Relation Extraction. In: Advances in Neural Information Processing Systems, vol. 18, p. 171 (2006)
21. Cancedda, N., Gaussier, E., Goutte, C., Renders, J.: Word-Sequence Kernels. MIT Press Journals, 1059 (2003)
22. Strapparava, C., Mihalcea, R.: Semeval-2007 Task 14: Affective Text. In: Proceedings of Semeval, vol. 7 (2007)
23. Pedersen, T., Kolhatkar, V.: Wordnet:: Senserelate:: Allwords: a Broad Coverage Word Sense Tagger That Maximizes Semantic Relatedness. In: Proceedings of Human Language Technologies: the 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Demonstration Session, p. 17. Association for Computational Linguistics (2009)
24. Patwardhan, S., Banerjee, S., Pedersen, T.: Using Measures of Semantic Relatedness for Word Sense Disambiguation. In: Gelbukh, A. (ed.) CICLing 2003. LNCS, vol. 2588, pp. 241–257. Springer, Heidelberg (2003)
25. Wiegand, M., Klakow, D.: Convolution Kernels for Opinion Holder Extraction. In: Human Language Technologies: the 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, p. 795. Association for Computational Linguistics (2010)
26. Baccianella, S., Esuli, A., Sebastiani, F.: Sentiwordnet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In: Proceedings of the Seventh International Conference on Language Resources and Evaluation, LREC 2010, p. 2200. European Language Resources Association, ELRA (2008)

27. Agerri, R., Garc, A.: Q-Wordnet: Extracting Polarity From Wordnet Senses. In: Seventh International Conference on Language Resources and Evaluation, Malta (2009)
28. Cerini, S., Compagnoni, V., Demontis, A., Formentelli, M., Gandini, G.: Language Resources and Linguistic Theory: Typology, Second Language Acquisition, English Linguistics. Micro-Wnop: A Gold Standard for the Evaluation of Automatically Compiled Lexical Resources for Opinion Mining. Franco Angeli Editore, Milano (2007)
29. Chang, C.C., Lin, C.J.: LIBSVM: a Library for Support Vector Machines. Citeseer (2001)
30. Collins, M., Duffy, N.: New Ranking Algorithms for Parsing and Tagging: Kernels over Discrete Structures, and the Voted Perceptron. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, p. 263. Association for Computational Linguistics (2002)
31. Moschitti, A.: Efficient Convolution Kernels for Dependency and Constituent Syntactic Trees. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 318–329. Springer, Heidelberg (2006)

An Investigation of Code-Switching Attitude Dependent Language Modeling

Ngoc Thang Vu, Heike Adel, and Tanja Schultz

Institute for Anthropomatics, Karlsruhe Institute of Technology (KIT), Germany
thang.vu@kit.edu, heike.adel@student.kit.edu, tanja.schultz@kit.edu

Abstract. In this paper, we investigate the adaptation of language modeling for conversational Mandarin-English Code-Switching (CS) speech and its effect on speech recognition performance. First, we investigate the prediction of code switches based on textual features with focus on Part-of-Speech (POS) tags. We show that the switching attitude is speaker dependent and utilize this finding to cluster the training speakers into classes with similar switching attitude. Second, we apply recurrent neural network language models which integrate the POS information into the input layer and factorize the output layer into languages for modeling CS. Furthermore, we adapt the background N-Gram and RNN language model to the different Code-Switching attitudes of the speaker clusters which lead to significant reductions in terms of perplexity. Finally, using these adapted language models we rerun the speech recognition system for each speaker and achieve improvements in terms of mixed error rate.

Keywords: multilingual speech processing, code switch attitude, language model adaptation.

1 Introduction

Code-Switching (CS) speech is defined as speech that contains more than one language ('code'). It is a common phenomenon in multilingual communities where people of different cultures and language background communicate with each other [2]. The switch between languages can happen between or within an utterance. In this paper, we show that the decision whether and when a speaker changes the language is rather individual ('Code-Switching attitude').

For the automated processing of spoken communication in these scenarios, a speech recognition system must be able to handle code switches. However, the components of speech recognition systems are usually trained on monolingual data, particulary when there is a lack of multilingual training data. This is why the CS task appears to be difficult to solve.

While there have been promising research results in the area of acoustic modeling to handle Code-Switching, only few approaches so far address this challenge in the language model. Recently, it has been shown that recurrent neural network language models (RNNLMs) improve perplexity and error rates in speech recognition systems in comparison to traditional N-Gram approaches [9,10,15].

One reason for that is their ability to handle longer contexts. Furthermore, the integration of additional features as input is rather straight-forward due to their structure. Recently, we proposed an extended structure of recurrent neural networks in order to predict CS points. In this paper, we extend this work by showing that Code-Switching can be regarded as a speaker dependent phenomenon. Hence, it is possible to cluster speakers with similar Code-Switching attitudes to obtain more specific models. Our experimental results demonstrate that this clustering leads to significant improvements in terms of perplexity for each test speaker and that these improvements also transform into error rate reductions. Figure 1 illustrates our adaptation process.

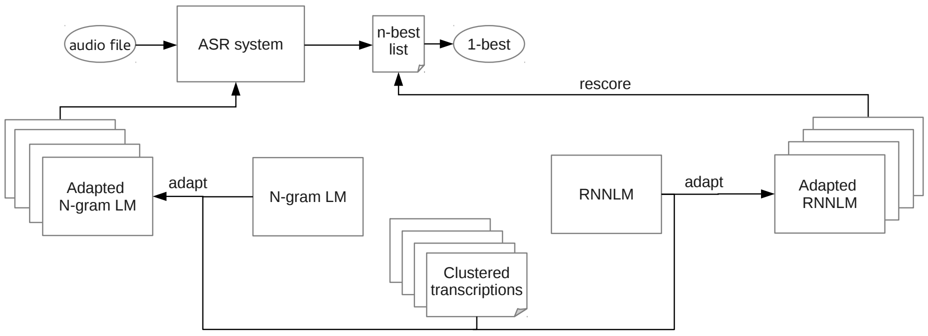


Fig. 1. Overview: language model adaptation to Code-Switching attitudes

The remainder of the paper is organized as follows: Section 2 reports on previous research in the area of Code-Switching, text clustering and language modeling using recurrent neural networks. Section 3 describes the SEAME corpus and analyzes it with focus on Part-of-speech tags triggering CS events. In section 4, we present how speakers can be clustered using the results of our analysis. Furthermore, we describe the adaptation of N-Gram and recurrent neural network language modeling for each Code-Switching attitude. In section 5, we present our experiments and results. The study is concluded in section 6.

2 Related Work

For this work, three different topics are investigated: 1) analysis of CS points and their integration into language models, 2) text clustering using similarity measures and 3) recurrent neural network language modeling and adaptation to more specific data. This section gives a short overview of prior work in these fields.

2.1 The Code-Switching Phenomenon

In [4,12,13], it is observed that code switches occur at positions in an utterance where they do not violate the syntactical rules of the involved languages. On the one hand, Code-Switching can be regarded as a speaker dependent phenomenon [3]. On the other hand, particular CS patterns are shared across speakers [14]. It is shown that speakers mainly switch to another language for nouns and object noun phrases. Therefore, the most frequent switches are between determiners and nouns and between verb phrases and object noun phrases.

In [16], different machine learning algorithms (for instance the Naive Bayes Classifier) trained on textual features are used to predict CS points. As features, word form, language identity, Part-of-Speech tags and the position of the word relative to the phrase are used. [5] compares four different kinds of N-Gram language models to predict Code-Switching. It is discovered that a class-based model which clusters all foreign words into their POS classes achieves the best performance. In [1], we show that the integration of POS tags into a neural network, which predicts the next language as well as the next word, leads to significant reductions in terms of perplexity.

2.2 Clustering Textual Documents

There are different text clustering techniques, such as hierarchical clustering (bottom-up or top-down) or k-means. While hierarchical clustering often provides better results, its time complexity is quadratic. On the other hand, k-means has a linear time complexity. Each technique requires a distance or similarity measure. The most common measure is the cosine measure [17].

2.3 Recurrent Neural Networks and Their Adaptation

In the last years, neural networks have been used for a variety of tasks. [9] introduces a refined form of neural networks for the task of language modeling. The so-called recurrent neural networks are able to handle long-term contexts since the input vector does not only contain the current word but also the previous output from the neurons in the hidden layer. It is shown that these networks outperform traditional language models, such as N-Grams which only contain very limited histories. In [10], the network is extended by factorizing the output layer into classes to accelerate the training and testing processes. Recently, further information has been added to neural networks. [15] augments the input layer to model features, such as topic information or Part-of-Speech tags. In [6], it is shown that adaptation of Recurrent Neural Network Language Models in form of one-iteration retraining leads to improvements in the word error rate when the adapted models are applied for rescoring.

3 Code-Switching Prediction Using Part-Of-Speech

This section describes the SEAME data corpus and the analyses which we performed on the data: 1) A speaker independent analysis in which we compute the CS rate after each Part-of-speech tag over all speakers and 2) A speaker dependent analysis in which the CS rate per speaker is calculated.

3.1 SEAME Corpus

SEAME (South East Asia Mandarin-English) is a conversational Mandarin-English Code-Switching speech corpus recorded from Singaporean and Malaysian speakers, created and collected by [7]. The corpus was used for the research project ‘Code-Switch’, jointly performed by Nanyang Technological University (NTU) and Karlsruhe Institute of Technology (KIT). The recordings consist of spontanously spoken interviews and conversations of about 63 hours of audio data. For this task, all hesitations are deleted and the transcribed words are divided into four categories: English words, Mandarin words, particles (Singaporean and Malaysian discourse particles) and others (other languages). These categories are used as language information in our neural networks. The average number of code switches between Mandarin and English is 2.6 per utterance. The duration of monolingual segments is very short: More than 82% English and 73% Mandarin segments last less than 1 second with an average duration of English and Mandarin segments of only 0.67 seconds and 0.81 seconds respectively. In total, the corpus contains 9,210 unique English and 7,471 unique Mandarin vocabulary words. The corpus is divided into three disjoint sets (training, development and test set). The data is assigned to them based on several criteria (gender, speaking style, ratio of Singaporean and Malaysian speakers, ratio of the four categories, and the duration in each set). Table 1 lists the statistics of the SEAME corpus in these sets.

Table 1. Statistics of the SEAME corpus

	Train set	Dev set	Eval set
# Speakers	139	8	8
Duration(hours)	59.2	2.1	1.5
# Utterances	48,040	1,943	1,018
# Token	525,168	23,776	11,294

3.2 Assigning POS Tags to Code-Switching Data

To be able to assign Part-of-Speech tags to our bilingual text corpus, we use two different taggers: On the one hand, the Stanford log-linear POS tagger for Mandarin and on the other hand, the Stanford log-linear POS tagger for English [19,20]. The tags are derived from the Penn Treebank POS tag set for Mandarin and English [8,22]. First, we determine Mandarin as matrix language (the main language of an utterance) and English as embedded language. If three or more words of the embedded language are detected, they are passed to the English tagger. The rest of the text is passed to the Chinese tagger, even if it contains foreign words. The idea is to provide the tagger as much context as possible.

However, most English words in the Mandarin segments are falsely tagged as nouns by the Chinese tagger. To avoid subsequent errors in the determination of trigger POS tags, we add a postprocessing step to the tagging process: We select all foreign words in the Mandarin segments and pass them to the English tagger in order to replace the wrong tags with the correct ones.

3.3 Speaker Independent Analysis

After having tagged the CS text, we select those tags that possibly predict CS points. The results are shown in table 2. First, we consider only those tags that appear in front of a CS point from Mandarin to English. Second, we investigate the tags predicting a CS point from English to Mandarin. In each case, only those tags are counted that occur more than 250 times in the text. Table 2 shows that CS points are most often triggered by determiners in Mandarin and by verbs and nouns in English. This seems reasonable since it is possible that a Mandarin speaker switches for the noun to English and afterwards back to Mandarin. It also corresponds to previous investigations as described in section 2.

Table 2. Mandarin and English POS that trigger a CS point

Tag	meaning	frequency	CS-rate
DT	determiner	11276	40.44%
DEG	associative 的	4395	36.91%
MSP	other particle	507	32-74%
VC	是	6183	25.85%
DEC	的 in a relative-clause	5763	23.86%
NN	noun	49060	49.07%
NNS	noun (plural)	4613	40.82%
RP	particle	330	36.06%
RB	adverb	21096	31.84%
JJ	adjective	10856	26.48%

3.4 Speaker Dependent Analysis

The previous analysis detects CS rates up to less than 50%. Thus, the triggering may not be reliable. A possible reason is that one speaker switches often after a specific tag while other speakers do not. Hence, a speaker dependent analysis is performed. The CS rate for each tag is computed for each speaker. Then, minimal, maximal and mean values and standard deviations are calculated. Indeed, the spread between minimal and maximal values is quite high for most of the tags. Figure 2 shows the distribution of the speaker dependent CS rates for all tags that appear more than 250 times in the text.

To sum up, whether a Part-of-speech tag triggers a CS event is rather speaker dependent. This corresponds to the previous investigations described in section 2. Hence, a model that includes all individual deviations cannot be very precise.

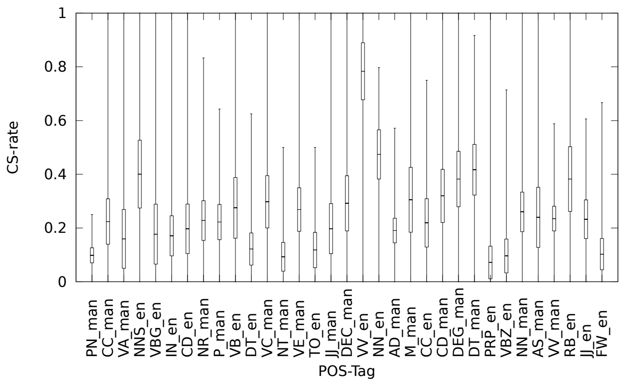


Fig. 2. Distribution of speaker dependent CS rates

4 Code-Switching Attitude Dependent Language Modeling

As shown in section 3, Code-Switching attitude is speaker dependent. Hence, we perform a clustering of the manual transcriptions of all speakers of our training data into K different groups to describe different Code-Switching attitudes. After that, we are able to adapt our language model to each class. Thus, we obtain K different language models that model Code-Switching more precisely and, therefore, achieve better recognition results.

4.1 Text Clustering

We apply the k-means algorithm to cluster the training transcriptions. As similarity measure, we chose the cosine distance because it was applied succesfully to cluster documents in the past. The following equation shows the computation of the cosine distance. d_1 denotes a vector representing document 1 and d_2 a vector for document 2.

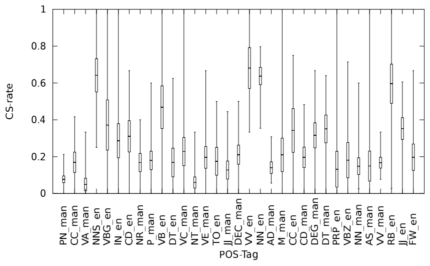
$$Dist(d_1, d_2) = (d_1.d_2)/(||d_1|| \cdot ||d_2||) \tag{1}$$

For the Code-Switching modeling, we define the document vectors d as follows:

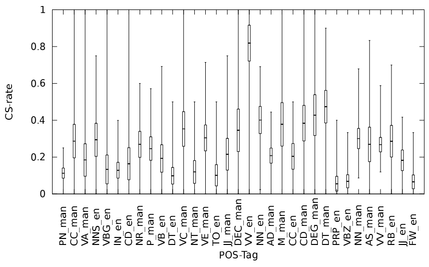
$$d = [f_{cs}(POS_1)/f(POS_1), ..., f_{cs}(POS_n)/f(POS_n)] \tag{2}$$

$f_{cs}(POS_i)$ defines the number of switches after the Part-of-Speech tag i in the given document while $f(POS_i)$ refers to the number of all occurences of the tag. After the clustering process converges (when there are no changes in the clusters), we use the mean vector of each cluster as representant.

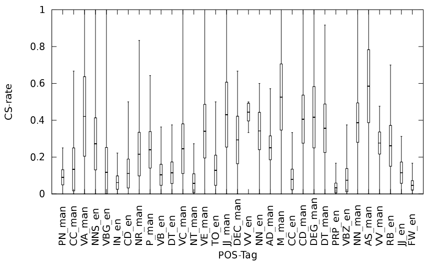
Figure 3 shows for the example of three classes that clustering helps to decrease the spread of the Code-Switching attitudes. There are still tags for which the clustered speakers show different attitudes but there are also tags for which



(a) distribution in class 1



(b) distribution in class 2



(c) distribution in class 3

Fig. 3. Distribution of speaker dependent Code-Switching rates after clustering

their attitude is quite similar. For example, the spread of the English tag 'NN' (noun) is discriminated into upper and lower values by the classes.

Further analyses show that, on the one hand, the classes divide different nationalities while, on the other hand, the gender of the speakers and the speaking style is similar in all classes. Hence, Code-Switching attitude seems to depend on the nationality but not on the gender or style. Table 3 summarizes the most important results for three classes.

Table 3. Analysis of the speakers that are clustered into one class

(con. denotes conversational speech, while iv. stands for interview)

Class	nationalities	gender	style
1	66 % Malaysia, 34 % Singapour	58 % female, 52 % male	5 % con., 95 % iv.
2	7 % Malaysia, 93 % Singapour	55 % female, 45 % male	47 % con., 53 % iv.
3	0 % Malaysia, 100 % Singapour	66 % female, 34 % male	29 % con., 71 % iv.

4.2 Language Modeling

Training. We train two different language models (LM): An N-Gram LM for the decoding process and a recurrent neural network LM for rescoring.

N-Gram Language Model for Code-Switching. We use the SRI Language Modeling Toolkit [18] to build trigram LMs from the SEAME training transcriptions. These models are interpolated with two monolingual language models that were created from 350k English sentences from NIST and 400k Mandarin sentences from the GALE project. The vocabulary of 30k entries contains all words in the transcriptions and the most frequent words in the monolingual corpora. Furthermore, characteristics of Code-Switching from the SEAME training transcriptions are analyzed and additional Code-Switching text is generated artificially as described in [21].

Recurrent Neural Network Language Modeling for Code-Switching. In this paragraph, we describe the original version of the RNNLM toolkit [11] and our extension to it which is illustrated in figure 4. Vector $w(t)$, which represents the current word using 1-of-N coding, forms the input of the RNN. Its dimension equals the size of the vocabulary. Vector $s(t)$ contains the state of the network and is called 'hidden layer'. The network is trained using back-propagation through time (BPTT), an extension of the back-propagation algorithm for RNNs. With BPTT, the error is propagated through recurrent connections back in time for a specific number of time steps. Hence, the network is able to remember information for several time steps. The matrices U, V and W contain the weights for the connections between the layers. They are learned during the training phase. Moreover, the output layer is factorized into classes to accelerate the training and testing processes. Every word belongs to exactly one class. The classes are formed during the training phase depending on the frequencies of the words. Vector $c(t)$ provides the probabilities for each class and vector $y(t)$ the probabilities for each word given its class. Hence, the probability $P(w_i|history)$ is computed as shown in equation 3.

$$P(w_i|history) = P(c_i|s(t))P(w_i|c_i, s(t)) \tag{3}$$

In our extension of the RNNLM, the output classes do not depend on word frequencies but on languages. We use the language categorization described in section 3.1. Therefore, our model consists of four classes: One class for English words, one for Mandarin words, one for other languages and one for particles. We do not only intend to predict the next word but also the next language in our bilingual corpus. Hence according to equation 3, the probability of the next language is computed first and then the probability of each word given the language. Furthermore, we add another vector $f(t)$ to the network which provides features corresponding to the current word and concantenate the former input layer with this vector. According to the analyses described in section 3, we use POS tags as features. Vector $f(t)$ consists of 67 elements (31 Mandarin POS tags, 34 English POS tags, one feature for words classified as other languages and one feature for particles). During the training and testing phases, not only the current word is activated but also its feature. Because the POS tags are integrated into the input layer, they are also propagated into the hidden layer $s(t)$. Thus, features from several previous time steps are stored in the history.

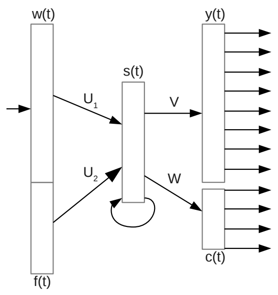


Fig. 4. RNNLM for Code-Switching based upon a figure in [10]

Hence in equation 3, the term $P(c_i|s(t))$ computes the next language c_i using not only information about previous words, but also previous features.

Adaptation. After the clustering, we retrain our models with the data of the different clusters. For the N-Gram model, we interpolate the baseline N-Gram model with an N-Gram model trained on the texts of one class. Hence, we obtain one N-Gram model per class. The interpolation weights are chosen to minimize the perplexity of the development set speakers that are similar to the class. Analog to this, we retrain one RNNLM per class. We perform one-iteration training using the texts of the different classes.

Rescoring. The decoding process contains two different passes. In the first pass, we run the speech recognition system to extract the N-best hypotheses using the speaker independent N-Gram LM. Based on the average score of the CS attitude dependent RNNLM on these N-bests, each speaker is assigned to a specific CS attitude. In the second pass, we rerun the decoding process using our CS attitude dependent N-Gram for each speaker and rescore the 100-best hypotheses using the CS attitude dependent RNNLM to obtain the best hypothesis.

5 Experiments and Results

This section reports the experiments and evaluations performed on the challenge of CS language modeling. Since the models are adapted to fit better to individual speakers, their perplexities are computed speaker-wise.

5.1 Clustering Experiments with K-Means

The most important parameter in the clustering process is the cluster size. Hence, different sizes are tested. Since rescoring experiments with the RNNLM are faster than decoding experiments with the N-Gram model, the following values are calculated using the 100-best lists of the speaker independent system and the RNNLM system to compute perplexities and rescore the hypotheses.

Table 4. Minimum and maximum perplexity on the development set speakers

Speaker	Baseline	2 classes	3 classes	4 classes	5 classes
Speaker 1	257.47	234.29 - 270.57	234.08 - 270.56	233.39 - 267.56	237.32 - 274.98
Speaker 2	221.00	194.78 - 218.96	194.66 - 219.04	194.41 - 216.52	196.96 - 222.16
Speaker 3	253.31	242.94 - 283.21	243.54 - 283.44	242.87 - 280.27	242.03 - 289.04
Speaker 4	201.28	186.14 - 213.38	186.70 - 213.55	185.96 - 212.29	188.37 - 217.14
Speaker 5	339.50	299.69 - 355.34	299.84 - 355.75	299.58 - 349.79	303.15 - 366.97
Speaker 6	151.92	135.00 - 156.76	135.05 - 156.81	134.92 - 156.67	135.49 - 160.82
Speaker 7	225.82	221.99 - 251.83	222.00 - 250.66	223.56 - 252.66	220.47 - 279.62
Speaker 8	194.35	189.30 - 206.97	189.30 - 206.32	188.97 - 207.64	191.10 - 222.73

Table 4 summons the minimum and maximum perplexity on the eight development set speakers in order to detect the most appropriate cluster size.

It can be noted that the results of two, three and four classes are quite similar and superior to a cluster size of five. Although the worst result per cluster performs worse than the baseline model, most of the classes of each cluster lead to an improvement of the perplexity. These results support the speaker dependent analysis: It is possible to adapt the language model to individual Code-Switching attitudes. The three best cluster sizes (2, 3, and 4 classes) are further evaluated regarding their word error rate reduction in the rescoring process. This results in a best cluster size of 3 classes. This is reasonable since two classes might not cover enough different speaker attitudes, while four or more classes do not contain enough training data per class. Hence, a cluster size of three is chosen for further evaluations.

5.2 Results

This subsection summons the results of our experiments, including the test of our models on the evaluation set speakers. Table 5 shows the minimum and maximum perplexities per speaker of all three adapted models and the baseline model for each the N-Gram and the RNN LM.

Again, the models adapted on the clustered training data can improve the performance in terms of perplexity for all speakers. Finally, the adapted models are used to decode and rescore the evaluation set speakers. We use the system in [21] to perform the decoding process. As performance measure, the Mixed Error Rate (MER) as proposed in [21] is calculated. It applies word error rates to English and character error rates to Mandarin and is the weighted average over all English and Mandarin parts of the speech recognition output. By applying character based error rates to Mandarin, the performance does not depend on the applied word segmentation algorithm for Mandarin and thus the performance can be compared across different segmentations. Table 6 shows the results on the SEAME development and evaluation set.

Table 5. Minimum and maximum perplexities on the evaluation set speakers

Speaker	N-Gram	Adapted N-Gram	RNNLM	Adapted RNNLM
Speaker 1	317.84	302.94 - 326.24	200.66	197.74 - 204.82
Speaker 2	265.77	253.73 - 270.81	181.60	175.85 - 185.48
Speaker 3	327.09	302.56 - 352.60	187.04	170.92 - 197.30
Speaker 4	232.83	213.33 - 248.30	174.13	160.58 - 185.28
Speaker 5	367.72	365.47 - 409.25	364.59	327.33 - 392.68
Speaker 6	175.28	162.02 - 181.83	275.89	253.67 - 299.37
Speaker 7	318.50	306.58 - 375.41	286.31	286.30 - 292.29
Speaker 8	292.53	281.57 - 331.04	256.99	241.69 - 268.23

Table 6. Mixed error rate results after decoding and rescoring with the adapted models

Model	development set	evaluation set
Speaker-independent N-Gram model	34.74%	29.23%
Adapted N-Gram model + RNNLM	34.47%	28.89%

6 Conclusions

In this paper, we showed that Code-Switching is a speaker dependent phenomenon. Therefore, we clustered similar Code-Switching attitudes using cosine-distances. Furthermore, we trained recurrent neural network language models for the Code-Switching task by adding POS information to the input layer and by factorizing the output layer into languages. Afterwards, we adapted our background N-Gram and RNN language model using the corresponding training texts of these clusters. We showed that this approach leads to significant reductions in terms of perplexity. Finally, we used these adapted language models to rerun and rescore the speech recognition system for each speaker and achieved some improvements in terms of mixed error rate.

References

1. Adel, H., Vu, N.T., Kraus, F., Schlippe, T., Schultz, T., Li, H.: Recurrent neural network language modeling for code switching conversational speech. In: ICASSP (2012)
2. Auer, P.: Code-switching in conversation. Routledge (1999)
3. Auer, P.: From codeswitching via language mixing to fused lects toward a dynamic typology of bilingual speech. International Journal of Bilingualism 3(4), 309–332 (1999)
4. Bokamba, E.G.: Are there syntactic constraints on code-mixing? World Englishes 8(3), 277–292 (1989)
5. Chan, J.Y.C., Ching, P., Lee, T., Cao, H.: Automatic speech recognition of Cantonese-English code-mixing utterances. In: Proc. of ICSLP (2006)
6. Kombrink, S., Mikolov, T., Karafiát, M., Burget, L.: Recurrent neural network based language modeling in meeting recognition. In: Proc. of ICSLP (2011)
7. Lyu, D.C., Tan, T.P., Chng, E.S., Li, H.: An analysis of a Mandarin-English code-switching speech corpus: Seame. In: ICSLP (2010)

8. Marcus, M.P., Marcinkiewicz, M.A., Santorini, B.: Building a large annotated corpus of English: The penn treebank. Computational Linguistics 19(2), 313–330 (1993)
9. Mikolov, T., Karafiát, M., Burget, L., Cernocky, J., Khudanpur, S.: Recurrent neural network based language model. In: Proc. of ICSLP (2010)
10. Mikolov, T., Kombrink, S., Burget, L., Cernocky, J., Khudanpur, S.: Extensions of recurrent neural network language model. In: ICASSP, pp. 5528–5531 (2011)
11. Mikolov, T., Kombrink, S., Deoras, A., Burget, L., Cernockỳ, J.: Rnnlm–recurrent neural network language modeling toolkit. In: Proc. of the 2011 ASRU Workshop, pp. 196–201 (2011)
12. Muysken, P.: Bilingual speech: A typology of code-mixing, vol. 11. Cambridge University Press (2000)
13. Poplack, S.: Syntactic structure and social function of code-switching, vol. 2. Centro de Estudios Puertorriqueños, City University of New York (1978)
14. Poplack, S.: Sometimes i'll start a sentence in spanish y termino en español: toward a typology of code-switching. Linguistics 18(7-8), 581–618 (1980)
15. Shi, Y., Wiggers, P., Jonker, C.M.: Towards recurrent neural networks language models with linguistic and contextual features. In: ICSLP (2012)
16. Solorio, T., Liu, Y.: Learning to predict code-switching points. In: Proc. of the EMNLP, pp. 973–981. Association for Computational Linguistics (2008)
17. Steinbach, M., Karypis, G., Kumar, V., et al.: A comparison of document clustering techniques. In: KDD Workshop on Text Mining, vol. 400, pp. 525–526 (2000)
18. Stolcke, A., et al.: Srilm-an extensible language modeling toolkit. In: Proc. of ICSLP, vol. 2, pp. 901–904 (2002)
19. Toutanova, K., Klein, D., Manning, C.D., Singer, Y.: Feature-rich part-of-speech tagging with a cyclic dependency network. In: Proc. of the HLT-NAACL, pp. 173–180 (2003)
20. Toutanova, K., Manning, C.D.: Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In: Proc. of the 2000 Joint SIGDAT Conference EMNLP/VLC, pp. 63–70 (2000)
21. Vu, N.T., Lyu, D.C., Weiner, J., Telaar, D., Schlippe, T., Blaicher, F., Chng, E., Schultz, T., Li, H.: A first speech recognition system for Mandarin-English code-switch conversational speech. In: ICASSP, pp. 4889–4892 (2012)
22. Xue, N., Xia, F., Chiou, F.D., Palmer, M.: The Penn Chinese treebank: Phrase structure annotation of a large corpus. Natural Language Engineering 11(2), 207 (2005)

Author Index