Evaluation of Topic Models for Supervised Word-Sense Disambiguation

Computerlinguistische Anwendungen

Robert Brünig
Fakultät für Informatik
Technische Universität München
robert.bruenig@in.tum.de

Jakob Buchgraber
Fakultät für Informatik
Technische Universität München
jakob.buchgraber@tum.de

Philipp Dowling
Fakultät f"ur Informatik
Technische Universität München
dowling@in.tum.de

***Abstract*— We present an evaluation of the performance that different topic models achieve when applied to the problem of supervised word-sense Disambiguation (WSD). Our algorithm was tested on the SensEval-3 English lexical sample task, using Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA) at different dimensionalities, as well as the Hirearchical Dirichlet Process (HDP). All models were trained on the entire Wikipedia corpus, and applied in a variant on the Lesk algorithm. Our approach is based on selecting the word sense that most often occurs in the top *k* most similar training instances.**

Our results are on par with those of many of the entries to the SensEval-3 evaluation exercise. We found LSA at 350 topics (*k = 12*) to perform best (*recall 0.64*), with LSA generally outperforming LDA, and both outperforming HDP, which could not be properly evaluated partially due to time contraints.

***Index Terms*— topic model, LSA, LDA, HDP, word-sense disambiguation**

I. Introduction

In this paper, we present an evaluation of the performance that different topic models achieve when applied to the problem of supervised word-sense disambiguation (WSD) in a variation on the Lesk algorithm.

A. Supervised Word-Sense Disambiguation

In virtually all languages, some words can have different meanings or senses. WSD is the problem of determining which sense of a word is being employed in a given context. In supervised WSD, both the words to be disambiguated and their possible senses are known beforehand, and labelled sample data is provided for each word sense. Consider the following example: The noun $atmosphere$ can, among other senses, refer to the air that surrounds the earth, as demonstrated by this sentence:

> *Rising levels of greenhouse gas emissions are damaging the atmosphere.*

The term can also refer to the general mood surrounding an event.

> *Most attendees were in agreement that the party had a very pleasant atmosphere to it.*

Given sufficient data, supervised WSD systems should automatically determine the sense of a certain word in a new context. This is commonly done by training a classifier with feature vectors generated from the labelled data. State of the art systems for WSD currently achieve accuracies of over 90% for certain tasks, however there are a number of different problem settings and top scores vary significantly based on the details of the task. For the SenseEval-3 English lexical sample task, which is the task we worked on, top recall scores were around 72% [1].

B. Topic Models

Topic models are statistical models that aim to represent textual data as vectors in a topic-based vector space. Topic models have been applied to many different information retrieval and text clustering problems, but also in domains related to bioinformatics and other fields [2]. In essence, topic models learn semantic patterns and relations from unstructured data, in order to perform dimensionality reduction.

A topic in this context is essentially a probability distribution over all words in the dictionary, with words that are relevant to the topic carrying positive weights, and the other words being assigned a weight very close or equal to zero. Each dimension of the output vector then represents a weight of how well the document fits the particular topic. To train a topic model, the respective set of model parameters is estimated through methods such as maximum likelihood or bayesian inference (in the form of maximum a posteriori (MAP) estimation), by drawing samples from a large corpus. Most topic models are unsupervised algorithms.

As a result of the dimensional reduction performed by converting documents into topic space, topic models enable better computation of semantic document similarity. Even documents that share no words directly but discuss a similar issue are likely to have similar vectors in topic space, even though their cosine similarity would be zero when considered as bag-of-words or TF-IDF vectors. In our experiments,

we dealt with three specific topic models: Latent Semantic Analysis (LSA), Latent Dirichlet Allocation (LDA), and Hierarchical Dirichlet Process (HDP) [3] [4] [5].

C. Applications of Topic Models in WSD

WSD algorithms usually involve either training classifiers, or computing similarities between contexts and glossaries (or other contexts). In both cases, topic models have strong potential applications. When training a classifier, the topic vectors themselves can be seen as features, and for computing simiarities topic models can provide much higher accuracy than more primitive methods like word overlap. Topic models have been applied to WSD in a number of different ways [6]. In fact, the LDA with WORDNET (LDAWN) model was developed specifically for the problem of unsupervised WSD [7].
Our approach to supervised WSD employs topic models to improve the comparison of contexts between training and evaluation data. Contexts are converted into topic space, and on evaluation the context similarities are used to establish the most likely sense for the instance.

II. TOPIC MODELS

Our experiments dealt with three different topic models: Latent Semantic Analysis, Latent Dirichlet Allocation and the Hierarchical Dirichlet Process. For more detailed description of each of these models see [3] [4] [5].

A. Latent Semantic Analysis

Latent Semantic Analysis (LSA) is a method that allows to identify topcis in unstructured text. It uses the bag of words model and the notion that words with similar meaning will occur in similar pieces of text.
When using LSA, one would typically construct a matrix C with each row representing a unique word and each column representing a document or paragraph. The entry $C_{i,j}$ would then contain the number of occurrences of the (unique) word i in the document j. In our particular implementation we chose to use the TF-IDF frequency instead of plain word counts, in an attempt to boost the relevance of rare words. The matrix C will invariably be very sparse as each document only contains a tiny fraction of the total number of words. Furthermore, with large numbers of documents as input the matrix will also be of very high rank and simply be too massive to process or even fit in memory. LSA thus uses a mathematical technique called Singular Value Decomposition (SVD) to reduce the rank of the matrix to k, with k usually being in the low hundreds. In short, the SVD of a matrix C leads to three matrices so that $C = U\Sigma V^T$ is true, with U and V being orthogonal matrices and Σ being a diagonal matrix. The entries $\Sigma_{i,i}$ are called the singular values with U_i and V_i being the left and right singular vectors respectively. The dimensionality reduction to rank k then works by selecting the k biggest singular values and the corresponding singular vectors $C_k = U_k\Sigma_k V_k$. This leads to a mathematical approximation of rank k that has minimal error.

The resulting low dimensionality matrix C_k is now referred to as *semantic space* and finding similar words/topics boils down to computing the *cosine distance* between two word vectors. A cosine value close to one means that the values are very similar (e.g. synonyms) while a value close to zero hints at words with different meaning.

B. Latent Dirichlet Allocation

Latent Dirichlet Allocation is a more sophisticated way of modelling and learning to discover topics in text. In contrast to LSA, LDA is a generative model that represents each document as a mixture of a small number of topics where each word in a document exists because it was generated by one of the document's topics. More specifically, LDA assumes that each document is generated in the following fashion:

1) Determine the number of words N in the document according to some probability distribution (e.g. Poisson).
2) Select a mixture of k topics for the document according to a Dirichlet distribution. Due to the lack of other evidence all k parameters α_i of the distribution are set to the same α.
3) For each topic choose another parameter β for the Dirichlet prior of the word distribution.
4) Finally, generate each word by first picking a topic according to the multinomial distribution (as given by the chosen Dirichlet distribution) and second by using the topic's multinomial distribution to generate the word itself.

Assuming this generative model, LDA then tries to backtrack from the documents to find the k topics that have generated the document. More specifically, it tries to find the before mentioned multinomial distributions using Bayesian inference techniques such as Gibbs sampling and expectation propagation.

C. Hierarchical Dirichlet Process

The Hierarchical Dirichlet Process is a non-parametric approach to topic models that utilizes similar underlying models to LDA. In fact, HDP can be considered a natural generalization of LDA with an unbound number of topics. The most important distinction of HDP to LDA (or LSA) is that the number of topics is not specified beforehand, but is instead learned from the data. This is achieved through utilizing nested Dirichlet processes - one for each document, with all processes sharing a common base distribution H, which is also drawn from a Dirichlet process [5].

While HDP is referred to as a non-parametric method, there are in fact a few parameters that must be manually specified. These are related to the different Dirichlet processes

Algorithm 1 Word-Sense Disambiguation

initialize TM
sense_vectors := *Map*()
procedure TRAIN
    for all $instance \in trainset$ **do**
        $vector := TM.convert(instance.bow)$
        $l := sense\_vectors[instance.word]$
        $l.append((vector, sense\_id))$
procedure EVALUATE
    results := *Map*()
    for all $instance \in evaluationset$ **do**
        $vector := TM.convert(instance.bow)$
        $candidates := sense\_vectors[instance.word]$
        sort candidates by cosine similarity to vector
        cut off candidates at index k
        $best\_sense := most\_freq\_sense(candidates)$
        $results[instance.id] := best\_sense$
        return $results$

and distributions utilized by the model. We will briefly discuss our settings of these parameters in section IV.

III. METHOD

We created LSA, LDA and HDP topic models on the english wikipedia corpus (see III-A). To investigate the influence of topic count on the results, we computed multiple instance of LDA and LSA models with different amounts of topics. Note that we do not include this step in our algorithm description - it is simply assumed that a trained topic model is already available.

Our algorithm consists of a training phase and an evaluation phase, however the training phase does not actually train a classifier. Instead, the contexts of all training instances are converted to the vector space of the topic model, and supsequently they are saved associated with the resepctive sense id.
The evaluation of a test instance is then done as follows. The test instance is also converted to the vector space of the topic model. Then, the top k training contexts for the word in question are retrieved using the cosine similarity of the contexts. Finally, the sense id that occurs most frequently in the top k results is chosen as the result for the test instance. Note that setting k very high will simply result in always choosing the most frequent sense, which is in fact the baseline to which we measure the system's performance.

A. Training of Topic Models

For the training of a topic model large amounts of text are required. For providing that data we looked at the English Wikipedia. Wikipedia does frequent backups of its articles and provides them for public use. We used the latest dump (June 2014) containing about 10GiB of compressed XML data, or 3.6 million documents [8]. The topic models were all trained using this data in an unsupervised manner.

The implementations of LSA, LDA and HDP topic models that we used were all provided in the python library gensim. Gensim is a free, open-source library for topic modelling. For using the wikipedia dump with gensim, it first had to be preprocessed using TF-IDF. Gensim was then used to create LSA, LDA and HDP models with different levels of dimensionalities from that corpus. We did not stem our training data, as this would have been very computationally expensive. In evaluation, the context data was therefore also not stemmed.

There are other models that are also supported by gensim that were not used for different reasons. Random projections (RP) [9] and log entropy (LE) [10] models were not used due to time contraints, but also because we did not expect them to outperform our other models. LE does not perform any dimensionality reduction, and RP does not usually outperform LSA.

IV. EXPERIMENTS

A. Experimental Setup

We used the SensEval-3 English lexical sample data as our training and evaluation set, so that we would be able to compare the performance of our approach to existing systems. The baseline performance for this task is to always choose the most frequent sense for each word.

We ran a large number of experiments at different parameters in order to find the optimal model and configuration. LSA was evaluated at n =150, 200, 250, 300, 350 and 400 topics, each with a single pass over the corpus, two power iterations and no decay.

LDA was only evaluated at 200, 300 and 400 topics, since it is more computationally expensive to create those models. Here we used a symmetric prior of $\alpha = \frac{1}{n}$ for the per-document topic distribution, with the same value for β, the per-topic word distribution.

HDP was evaluated using the default parameters of Gensim: $\kappa = 1, \tau = 64, \alpha = 1, \gamma = 1, \eta = 0.01$. Unfortunately, we were not able to properly use the HDP model, as the topics it generated did not appear make sense, and were all nearly identical. Due to time constraints we were not able to fix these issues and retrain the model. The results are nevertheless included in this paper.

B. Results

Figure 1, Figure 2 and Figure 3 visualize the results we obtained for different dimensionalities of LSA and LDA. The failed HDP attempt is not included (it was in fact lower than the baseline performance).

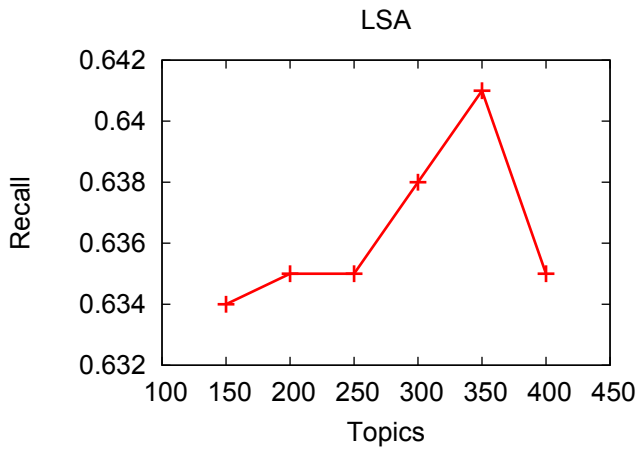

Fig. 1. LSA results

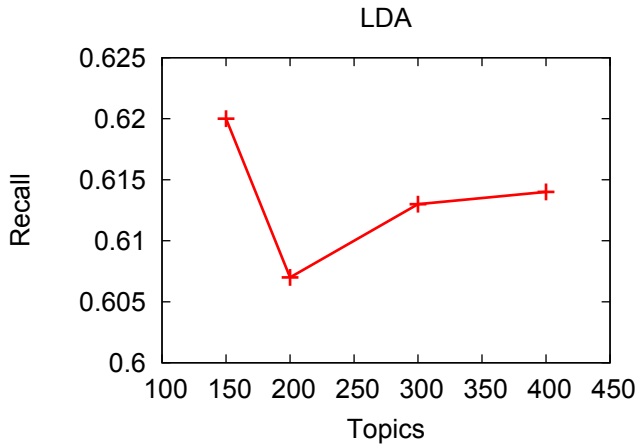

Fig. 2. LDA results

As you can see, LSA outperforms LDA, and achieves its highest result of 64.1% at 350 topics with $k = 10$. The best performance for LDA was a recall of 62% at 150 topics. The HDP model only achieved a recall of 47.4%, which is below baseline performance (which is possible as only values of k below 50 were considered). Other systems in SensEval-3 ranged from approximately 60% to 73% recall [1], so while our system is not on par with the state of the art in its current form, it performs reasonably well and presents a decent approach to the presented task.

C. Outlook

The algorithm presented here was fairly primitive, and serves mainly as a benchmark to compare the performance of different topic models for WSD. To optimize the systems performance, additional features may be considered and used to train a classifier of some sort. The entry *IRST-Kernels* by the Team ITC-IRST at SensEval-3 was one of the highest performing systems, and used LSA in conjunction with paradigmatic and syntagmatic information and other

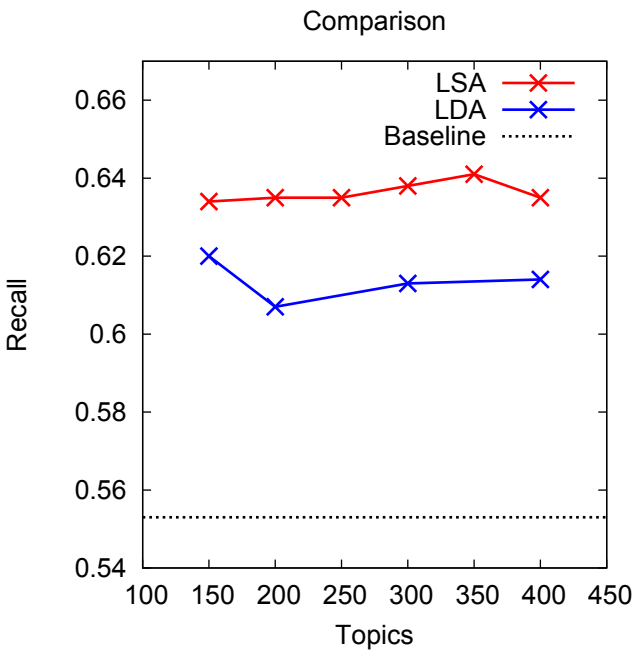

Fig. 3. Comparison of results and baseline

features in an SVM classifier [1]. It is conceivable to achieve significantly higher performance by following similar methods, especially by considering local features in conjunction with LSA (which ignores local features).

There are also many more topic models that can be tried, as well as parameters in the training that can be tweaked. Going beyond topic models, any dimensionality reduction algorithm could be applied to this approach of WSD. For example, an interesting method to consider might be marginalized Stacked Denoising Autoencoders (mSDA), which actually come from the field of deep learning but have in some benchmarks outperformed LDA in both accuracy and computational complexity [11].

V. Conclusion

We found LSA to outperform LDA in our experiments, which we did not initially expect, as LDA supposedly learns a more sophisticated model of the language it is trained on. This discrepancy is likely due to insufficient tuning of parameters, or the broadness of topics in the Wikipedia corpus. In any case, we have shown that topic models present a viable approach to WSD, as all of our experiments significantly outperformed the baseline and matched performance of existing systems. Topic models also have the potential to be used achieve even better results than we were able to attain during our experiments.

We were unable to train a HDP model on our corpus that learned meaningful topics, although we were not able to determine why we achieved such poor results.

In future work, it may be interesting to consider more sophisticated algorithms for performing WSD than we present here, such as employing classifiers with global features provided by topic models in conjunction with local or other features.

REFERENCES

[1] R. Mihalcea, T. Chklovski, M. Rey, and A. Kilgarriff, "The SENSEVAL 3 English Lexical Sample Task," no. July, 2004.

[2] B. Zheng, D. C. McLean, and X. Lu, "Identifying biological concepts from a protein-related corpus with a probabilistic topic model," *Bmc Bioinformatics*, vol. 7, no. 1, p. 58, 2006.

[3] S. Deerwester, S. T. Dumais, G. W. Furnas, and T. K. Landauer, "Indexing by Latent Semantic Analysis," 1998.

[4] D. Blei, A. Ng, and M. Jordan, "Latent dirichlet allocation," *The Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003. [Online]. Available: http://dl.acm.org/citation.cfm?id=944937

[5] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. Blei, "Hierarchical dirichlet processes," *Journal of the American Statistical Association*, pp. 1–41, 2006. [Online]. Available: http://amstat.tandfonline.com/doi/abs/10.1198/016214506000000302

[6] L. Li, B. Roth, and C. Sporleder, "Topic Models for Word Sense Disambiguation and Token-based Idiom Detection," no. July, pp. 1138–1147, 2010.

[7] J. Boyd-Graber and D. Blei, "A Topic Model for Word Sense Disambiguation," no. June, pp. 1024–1033, 2007.

[8] Various, "Wikipedia:Database download." [Online]. Available: http://en.wikipedia.org/wiki/Wikipedia:Database_download

[9] R. Rehurek, "models.rpmodel Random Projections." [Online]. Available: http://radimrehurek.com/gensim/models/logentropy_model.html

[10] ——, "models.logentropymodel LogEntropy model." [Online]. Available: http://radimrehurek.com/gensim/models/rpmodel.html

[11] M. Chen, Z. Xu, K. Weinberger, and F. Sha, "Marginalized denoising autoencoders for domain adaptation," *arXiv preprint arXiv:1206.4683*, 2012. [Online]. Available: http://arxiv.org/abs/1206.4683