INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

Convolutional Neural Networks with Data Augmentation for Classifying Speakers' Native Language

Gil Keren[1], *Jun Deng*[1], *Jouni Pohjalainen*[1], *Björn Schuller*[1,2]

[1]Chair of Complex & Intelligent Systems, University of Passau, Germany
[2]Department of Computing, Imperial College London, UK
gil.keren@uni-passau.de

Abstract

We use a feedforward Convolutional Neural Network to classify speakers' native language for the INTERSPEECH 2016 Computational Paralinguistic Challenge Native Language Sub-Challenge, using no specialized features for computational paralinguistics tasks, but only MFCCs with their first and second order deltas. In addition, we augment the training data by replacing the original examples with shorter overlapping samples extracted from them, thus multiplying the number of training examples by almost 40. With the augmented training dataset and enhancements to neural network models such as Batch Normalization, Dropout, and Maxout activation function, we managed to improve upon the challenge baseline by a large margin, both for the development and the test set.

Index Terms: Computational Paralinguistics, Deep Learning, Convolutional Neural Networks.

1. Introduction

In recent years, neural networks have become empirically successful in a wide range of supervised learning applications, such as computer vision [1, 2], speech recognition [3, 4] and natural language processing [5]. In addition, in many applications, the traditionally used models that are based on hand-crafted features to represent the data, have been replaced by neural network models with convolutional layers [6] that are learned directly from raw data or from simpler general-purpose features in an end-to-end manner. In these models, the convolutional layers are typically the lower layers in the network (closer to the input) and can be seen as extracting features from small patches of data, to create a new and possibly more useful representation of the data. Prominent examples of the end-to-end approach can be seen in [1], in which an object recognition model is learned from raw images without any prior knowledge, and [4], where a state-of-the-art speech recognition model is learned directly from the speech spectrogram.

In many cases in the field of computational paralinguistics, including the ComParE Challenge 2016 [7], the pipeline of a machine learning model begins by extracting features to represent the data at hand: first, low-level descriptors (LLDs) are extracted from short time windows and include short term characteristics of the audio signal such as voicing probability, HNR, F0 and zero-crossing rate [8]. This first step is possibly followed by a number of functionals (e.g., mean, max, percentiles), computed over time on these LLDs. Recent studies have shown ([9, 10]), that the end-to-end approach can be adopted to applications in computational paralinguistics, by training a convolutional neural network model based on simpler and non-specialized features.

A necessary condition for the success of neural network models for classification is in many cases the presence of large amounts of labeled training data [11]. In many classification datasets in the field of computational paralinguistics, the number of labeled training examples is at most a few thousands [12, 13, 14], which is rather small when compared to object detection or speech recognition datasets that typically contain at least a few tens of thousands of labeled examples, and in many cases much more than that [15, 16, 4]. On the other hand, in audio datasets, this problem might be alleviated if every labeled example is long enough.

In an audio classification task, there might be some redundancy in long examples, when smaller parts of the whole example contain enough information each for correctly classifying the whole example. This is indeed the case, when very long time dependencies (relations between two distant time frames) in the data are not needed for correctly classifying the whole example. In that situation, we can safely introduce data augmentation of the training set by replacing each labeled example with many small patches from it, retaining the same class label. This approach is different to other data augmentation approaches, such as changing the speed of an audio signal ([17]).

In this year's ComParE challenge, the *Native Language* dataset is comprised of 5,132 labeled examples, where the length of each example is approximately 45 seconds. The task in the nativeness subchallenge is to classify the mother tongue of the speaker in each example. We assume that less than the whole 45 seconds are required for correctly classifying an example, therefore posing the possibility for the necessary data augmentation that will allow successfully training a large neural network model with convolutional layers, using relatively simple and general-purpose input features.

As a step towards the end-to-end approach, we extract only MFCC features with first and second order delta regression coefficients, and we train feedforward neural network models with convolutional layers to classify the data for the nativeness subchallenge of the ComParE 2016 competition. In addition, we introduce data augmentation as described above, and we use two well-known enhancements for the neural network models, namely *Dropout* [18] and *Batch Normalization* [19]. For comparison, we train a few additional neural networks and Support Vector Machine (SVM) classifiers, that vary in the input features used and training data augmentation.

Therefore, the purpose of this work is two-fold: First, to explore the possibility that general-purpose features of relatively lower complexity might yield good classification results when coupled with a neural network model, compared to models using the provided challenge features. Second, to explore the benefits of the training data augmentation resulting from training

http://dx.doi.org/10.21437/Interspeech.2016-261

models on smaller patches from the original training examples.

In the rest of the paper, we present the models we use and the specifications of the training data augmentation we apply (Section 2), our experiments and results (Section 3), and conclude our work in Section 4.

2. Model and Data Augmentation

For classifying audio signals with a neural network, we use a feedforward neural network comprised of a few convolutional hidden layers followed by a few fully connected hidden layers (dense layers), with a softmax layer [20] on top for classification. For this task, a more natural choice of neural networks might be Recurrent Neural Networks (RNNs), which are neural networks that process the input sequentially and are able to model dependencies over time. However, sequential processing of even a few tens of time steps can introduce a significant computational overhead, as the network then corresponds to a very deep neural network, with many more steps of computation. This becomes even more significant, when using variants of RNNs that require more computation when processing each time step, such as Long Short Term Memory networks (LSTMs) [21].

We elaborate on the key components of our feedforward neural networks.

2.1. Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are neural networks that contain one or more convolutional layers. A convolutional layer processes a two-dimensional input of size $m \times n$ with k channels $x \in \mathbb{R}^{m \times n \times k}$ in the following manner: The input x is split spatially across its first two dimensions into (potentially overlapping) patches of the same size $\{x_{ij}\}_{i \in I, j \in J}$, where $x_{ij} \in \mathbb{R}^{m_1 \times n_1 \times k}$. Then, the n-dimensional output vector of a patch x_{ij} is:

$$c_{ij} = \sigma(x_{ij} \odot W_1 + b_1, \ldots, x_{ij} \odot W_n + b_n), \qquad (1)$$

where $W_1, \ldots, W_n$ are weight matrices of size $m_1 \times n_1 \times k$, the bias terms are $b_1, \ldots, b_n \in \mathbb{R}$, $\odot$ is an element-wise matrix multiplication and σ is a non-linear activation function operating element-wise. The output of the convolutional layer is the spatial map $c = \{c_{ij}\}_{i \in I, j \in J}$ of dimension $|I| \times |J| \times n$. The number n is called the number of *feature maps* in the convolutional layer, and c can be viewed as a spatial two dimensional map with n channels in each spatial location.

Potentially, a convolutional layer can include a max-pooling mechanism after the above described process, that operates as follows: The output of the above procedure c is again split spatially across its first two dimensions into (potentially overlapping) patches $\{p_{ij}\}_{i \in I', j \in J'}$. Then, a *max* operation is performed across the first two dimensions of a given patch, for each patch separately, resulting in an n-dimensional representation for each path. The output of the max-pooling mechanism (and the whole convolutional layer) p is the spatial map of all the n-dimensional representations, $p \in \mathbb{R}^{|I'| \times |J'| \times n}$.

Other types of pooling exist, such as mean-pooling [22], but models in this work use only the max-pooling mechanism.

In the case of an audio signal, the dimension of x can be seen as $1 \times t \times k$, where t is the number of time steps in the audio signal and k in the number of features used to represent each time step. In this case, Figure 1 depicts the procedure described above (without pooling).

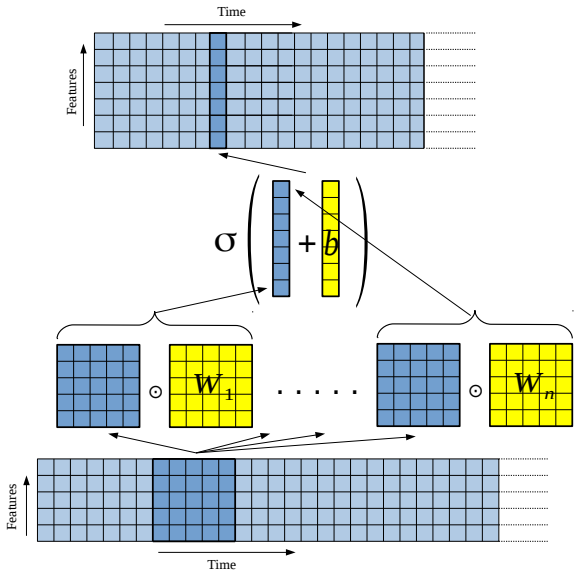

Figure 1: *A convolutional layer operating on an audio signal (without pooling)*

2.2. Batch Normalization

When training neural networks, the distribution of each layer's inputs changes during training, as the parameters of the previous layers change. The authors of [19] name this phenomenon *internal covariance shift*, and state that it slows down training by requiring lower learning rates and careful parameter initialization. To alleviate this issue, they introduce *Batch Normalization*, which applies a linear transformation on the output of a layer (just before applying the activation function), to enforce values for its mean and standard deviation. The mean and standard deviation are calculated per mini-batch of examples, and for each element of the output vector separately, in the case of a dense layer, or for each feature map separately and across all spatial locations, in the case of a convolutional layer. The target mean and standard deviation are again decided for each element of the output vector / feature map separately in the case of a dense layer / convolutional layer, and are additional parameters of the model that are learned during training. At inference time, mini-batch mean and standard deviation are replaced by their equivalents calculated on the whole training set.

All neural network models in this work use Batch Normalization for all hidden layers, not including the softmax layer.

2.3. Dropout

Dropout [18] is a technique for preventing overfitting in neural network models. When dropout is applied on some layer of a neural network, each element in the output (possibly multidimensional output, as in convolutional layers) is set to zero with probability p. This process is being done for each training example separately. At inference time, dropout is not used, therefore on average test examples might induce bigger outputs in absolute values, compared to training examples. To compensate for this effect, at inference time the output of the layer with dropout is multiplied by $\frac{1}{1-p}$. By using dropout, we might prevent different units in the neural network from co-adapting too much.

Table 1: *Architectures of the three main settings using neural networks.*

Data Augmentation	Features	Convolutional Layers	Dense Layers	Batch Normalization	Dropout
Yes	MFCC + deltas + delta-deltas	3	2	Yes	0.5
Yes	ComParE 2013	0	1	Yes	0.5
No	ComParE 2013	0	4	Yes	0.5

2.4. Data Augmentation

As was mentioned before, in some of our models we augment the training dataset by replacing each training example x with a few samples $x_1, \ldots, x_r$ from it. In this work, each training example is an audio signal, and each sample is a continuous patch of the original audio signal. It is important to note that, two samples that are almost completely overlapping, are still very different in values when compared element-wise for each element in the input.

For classification at inference time, we need to predict one class label for each example x in a test/development set. Note that, in the case of a neural network classifier with a softmax layer, for each sample x_i the network outputs the selected class label y_i, as well as a probability p_{ij} for each class j. We examine three different ways to combine the class predictions / probability distributions from the samples $x_1, \ldots, x_r$ into one class prediction y for the example x:

- Using the prediction of the sample the classifier is most sure about: $y = argmax_j \, max(\{p_{ij}\}_{1 \leq i \leq r})$
- Averaging all predictions: $y = argmax_j \, \frac{1}{r} \sum_{i=1}^{r} p_{ij}$
- Majority vote: y is the the most common value in $(y_1, \ldots, y_r)$. In case two classes' labels are equally common, the one with the lower index is chosen.

Except for enlarging the training dataset, replacing examples with shorter samples extracted from them has two additional properties, that are crucial in order to use a feedforward neural network model on this data. First, using fixed size samples from the training data forces all inputs to be of the same size, allowing us to use feedforward neural networks for this problem. Second, since the original audio clips from the Native Language dataset are approximately 45 seconds long, extracting LLDs from short time windows can result in an infeasible size of input, and using shorter samples may alleviate this problem.

3. Experiments

For the main model used in this paper, we augment the training set by replacing each example with 10 seconds samples from it, with the starting point of samples shifted by one second. For example, for a 45 seconds audio signal, 36 samples of 10 seconds will replace the original audio signal in the training set. The augmented training set contains 122,980 examples, compared to 3,300 in the original set. For each example in the augmented dataset, we extract 12 MFCC features and the logarithmic energy for windows of 25 ms shifted by 10 ms. In addition, we extract first and second order delta regression coefficients, to result in a total 39 features to represent each 25 ms time window. We apply mean and standard deviation normalization for each feature separately.

We experiment with different network architectures, learning rate, momentum, dropout rate and training algorithm, and we report the setting that was used in our best performing model on the development set. The network contains three convolutional layers with 350 feature maps each, with a window size of five time steps, shifted by two time steps. In the first two layers, max-pooling is used, to pool over non-overlapping groups of two time steps. After each convolutional layer, a maxout activation function [23] is applied, grouping sets of two feature maps with the *max* operator, to result in 175 output feature maps for each convolutional layer. The output of the third convolutional layer is flattened to be one dimensional, and is fed into two consecutive dense layers with 1000 hidden units each, and a maxout activation function applied over groups of two units. The output of the second dense layer is fed into a softmax layer to output a probability distribution over the possible classes.

Batch Normalization is applied for each of the hidden layers (convolutional and dense) just before applying the activation function. Dropout is as well applied for all hidden layers, with dropout probability of 0.5. Initial values for all weights in the network were sampled from a Gaussian distribution with a standard deviation of 0.1. Optimization is performed using stochastic gradient descent with momentum [24], using a learning rate of 0.1, momentum value of 0.9 and a mini-batch size of 128 examples. Gradient-Clipping [25] is applied as well, to limit the L^2 norm of the gradient to 100.0. The selected model is the one performing best on the development set, and training is stopped after 25 training epochs with no improvement in Unweighted Average Recall (UAR) on the development set, using all three ways to combine predictions on samples to predictions on full-length examples, described in Section 2.4. The experiments were performed using the deep learning framework Blocks [26] based on Theano [27, 28].

Table 2: *Unweighted Average Recall on the development set for different values of the complexity parameter C, for an SVM classifier with a linear kernel.*

C	UAR [%]
5	50.40
2	50.60
1	50.61
10^{-1}	50.33
10^{-2}	52.18
10^{-3}	53.64
10^{-4}	55.11
10^{-5}	53.66

For comparison, two additional settings using neural networks are evaluated. First, a setting where we use the same data augmentation technique as the main model, but use the challenge's provided feature set (the ComParE 2013 feature set, with 6373 features), which includes functionals computed over LLDs. Second, a setting in which we use challenge's provided feature set and do not use data augmentation. We extract the ComParE 2013 features from each 10 seconds example, using openSMILE [29]. Note, that these features are not ordered by

Table 3: *Unweighted Average Recall on the development and test sets for best models of each of the four experiment settings and the challenge baseline. When data augmentations was used, we also report the UAR of samples from the development set, before combining the predictions on samples to predictions on full-length examples.*

Classifier	Features	Data Augment'	Dev' UAR [%]	Best Predictor	Samples UAR [%]	Test UAR [%]
SVM (baseline)	ComParE 2013	No	45.10	—	—	47.50
SVM	ComParE 2013	Yes	55.11	Majority Voting	41.94	—
NN	ComParE 2013	No	50.03	—	—	—
NN	ComParE 2013	Yes	55.78	Majority Voting	42.98	—
NN	MFCC + d + dd	Yes	59.67	Mean	50.99	55.95
NN (ensemble)	MFCC + d + dd	Yes	61.47	—	—	58.33

time or any other order, therefore it does not make sense to use convolutional layers in this case. Since all settings are different one from the other by the input features or the size of the training data, we had to configure each setting separately. We found that the same learning rate, momentum and dropout values, as well as the number of hidden units in the dense layers and the choice of activation function that yield best results on the development set, are the same in our main setting and in the two additional settings described above, leaving only the number of hidden layers to vary. An overview of the architectures used for the three settings we discussed so far appears in Table 1.

In the last additional setting we use for comparison, we still use the same training data augmentation method and the ComParE 2013 features, but we use a Support Vector Machine (SVM) with a linear kernel as the classifier, instead of a neural network. Since an SVM in its basic version does not output a probability distribution over the possible classes, at inference time we use only majority voting in order to combine the predictions on the 10 seconds samples to one prediction on the longer example from the development / test set. We use the implementation of an SVM from the Python package scikit-learn [30]. The complexity parameter C is optimized using the development set. The Unweighted Average Recall on the development set for the different values of C is found in Table 2.

Table 3 contains the UAR for the development and test sets for each of the total four experiment setting (three with Neural Networks, one with an SVM) and the challenge baseline. From the results in the table, it is evident that, the best performing model on the development set was from the setting of a convolutional neural network with MFCC + deltas + delta-deltas input features, yielding a UAR of 59.67%, outperforming the challenge baseline (45.10%) by a large margin. A great improvement over the challenge baseline was observed also for the test set results, where our best model yielded a UAR of 55.95%, compared to 47.50% of the challenge baseline model. In particular, our best setting yielded superior results over a similar setting that differs only by the input features used. These results are positive evidence that simpler and more general-purpose features, paired with a convolutional neural network and a large enough dataset, can outperform a standard approach relying on a large number of specialized features, as we hypothesized in the introduction.

In addition, when data augmentation was used, Table 3 also reports the UAR on the samples themselves from the development set, and the best predictor, that is, the best method to combine the predictions on samples to one prediction on a full-length example. The results in the table allow us to evaluate the contribution of the training data augmentation, which seems to improve the results for both SVM and neural network classifiers, using the ComParE features (a setting with a neural network classifier, no data augmentation and MFCC + deltas + delta-deltas input features was not evaluated for comparison, due to infeasible input size). For the SVM classifiers, combing the predictions on samples that were classified with a UAR of only 41.94%, was enough to yield a UAR of 55.11% for full-length examples. A similar phenomenon regarding the gap between the UAR on samples and full-length examples is present in all experiment settings that include data augmentation.

To further improve our results, we evaluated the majority voting of an ensemble, comprised of our best model together with six other models from the same setting, that differ from the best model by the number of feature maps in the convolutional layers, dropout probability, learning rate and momentum value. The ensemble of models yielded a UAR of 61.47% on the development set, and 58.33% on the test set.

4. Conclusion

We compared models from a few settings of neural networks and Support Vector Machines (SVM), differing in data augmentation and input features used, for the classification task of the INTERSPEECH 2016 Computational Paralinguistics Native Language subchallenge. For training data augmentation, we replaced the original examples with shorter overlapping samples extracted from them, multiplying the number of training examples by more than 37. We found that the best performing model on the development set is a Convolutional Neural Network model, using the augmented data set and trained in an end-to-end manner from MFCCs and their first and second order delta regression coefficients as input features. In particular, our best model obtained an Unweighted Average Recall (UAR) of 55.95% on the challenge test set (and 58.33% with an ensemble of similar models), improving over the challenge baseline of 47.50% by a large margin. The results are a positive evidence that specialized features for tasks in computation paralinguistics can be replaced by general-purpose and simpler features like MFCCs. In addition, we evaluated the isolated effect of training data augmentation, that yielded a 10.01% improvement in UAR using an SVM classifier with the provided challenge features, and a 5.75% improvement in UAR using a neural network model with the same features.

5. Acknowledgements

This work has been partially supported by the European Community's Seventh Framework Programme through the ERC Starting Grant No. 338164 (iHEARu) and the BMBF IKT2020-Grant under grant agreement No. 16SV7213 (EmotAsS). We further thank the NVIDIA Corporation for their support of this research by Tesla K40-type GPU donation.

6. References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. of Advances in neural information processing systems (NIPS)*, Lake Tahoe, NV, 2012, pp. 1097–1105.

[2] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. of The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, 2015, pp. 1–9.

[3] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.

[4] D. Amodei, R. Anubhai, E. Battenberg, C. Case, J. Casper, B. C. Catanzaro, J. Chen, M. Chrzanowski, A. Coates, G. Diamos, E. Elsen, J. Engel, L. Fan, C. Fougner, T. Han, A. Y. Hannun, B. Jun, P. LeGresley, L. Lin, S. Narang, A. Y. Ng, S. Ozair, R. Prenger, J. Raiman, S. Satheesh, D. Seetapun, S. Sengupta, Y. Wang, Z. Wang, C. Wang, B. Xiao, D. Yogatama, J. Zhan, and Z. Zhu, "Deep speech 2: end-to-end speech recognition in English and Mandarin," *arXiv preprint arXiv:1512.02595*, 2015.

[5] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. of Advances in neural information processing systems (NIPS)*, Montreal, Canada, 2014, pp. 3104–3112.

[6] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[7] B. Schuller, S. Steidl, A. Batliner, J. Hirschberg, J. K. Burgoon, A. Baird, A. Elkins, Y. Zhang, E. Coutinho, and K. Evanini, "The INTERSPEECH 2016 Computational Paralinguistics Challenge: Deception & Sincerity," in *Proc. of INTERSPEECH 2016, 17th Annual Conference of the International Speech Communication Association*, San Francsico, CA, 2016.

[8] B. Schuller, S. Steidl, A. Batliner, A. Vinciarelli, K. Scherer, F. Ringeval, M. Chetouani, F. Weninger, F. Eyben, E. Marchi, M. Mortillaro, H. Salamin, A. Polychroniou, F. Valente, and S. Kim, "The INTERSPEECH 2013 Computational Paralinguistics Challenge: Social Signals, Conflict, Emotion, Autism," in *Proc. of INTERSPEECH 2013, 14th Annual Conference of the International Speech Communication Association*, Lyon, France, 2013, pp. 148–152.

[9] G. Keren and B. Schuller, "Convolutional RNN: an enhanced model for extracting features from sequential data," in *Proc. of 2016 International Joint Conference on Neural Networks (IJCNN) as part of the IEEE World Congress on Computational Intelligence (IEEE WCCI)*, Vancouver, Canada, 2016.

[10] G. Trigeorgis, F. Ringeval, R. Brückner, E. Marchi, M. Nicolaou, B. Schuller, and S. Zafeiriou, "Adieu features? End-to-end speech emotion recognition using a deep convolutional recurrent network," in *Proc. of the 41st IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Shanghai, P. R. China, 2016, pp. 5200–5204.

[11] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[12] B. Schuller, S. Steidl, A. Batliner, E. Nöth, A. Vinciarelli, F. Burkhardt, R. van Son, F. Weninger, F. Eyben, T. Bocklet, G. Mohammadi, and B. Weiss, "The INTERSPEECH 2012 Speaker Trait Challenge," in *Proc of. INTERSPEECH 2012, 13th Annual Conference of the International Speech Communication Association*, Portland, OR, 2012, pp. 254–257.

[13] B. Schuller, S. Steidl, A. Batliner, J. Epps, F. Eyben, F. Ringeval, E. Marchi, and Y. Zhang, "The INTERSPEECH 2014 Computational Paralinguistics Challenge: Cognitive & Physical Load," in *Proc. of INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association*, Singapore, Singapore, 2014, pp. 427–431.

[14] B. Schuller, S. Steidl, A. Batliner, S. Hantke, F. Hönig, J. R. Orozco-Arroyave, E. Nöth, Y. Zhang, and F. Weninger, "The INTERSPEECH 2015 Computational Paralinguistics Challenge: Degree of Nativeness, Parkinson's & Eating Condition," in *Proc. of INTERSPEECH 2015, 16th Annual Conference of the International Speech Communication Association*, Dresden, Germany, 2015, pp. 478–482.

[15] Y. LeCun, C. Cortes, and C. J. Burges, "The mnist database of handwritten digits," 1998.

[16] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[17] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Proc. of INTERSPEECH 2015, 16th Annual Conference of the International Speech Communication Association*, Dresden, Germany, 2015, pp. 3586–3589.

[18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[19] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. of of the 32nd International Conference on Machine Learning (ICML)*, Lille, France, 2015, pp. 448–456.

[20] J. S. Bridle, "Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition," in *Neurocomputing*. Springer, 1990, pp. 227–236.

[21] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[22] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[23] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. C. Courville, and Y. Bengio, "Maxout networks," in *Proc. of the 30th International Conference on Machine Learning (ICML)*, Atlanta, GA, 2013, pp. 1319–1327.

[24] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proc. of the 30th international conference on machine learning (ICML)*, Atlanta, GA, 2013, pp. 1139–1147.

[25] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *Proc. of the 30th International Conference on Machine Learning (ICML)*, Atlanta, GA, 2013, pp. 1310–1318.

[26] B. van Merriënboer, D. Bahdanau, V. Dumoulin, D. Serdyuk, D. Warde-Farley, J. Chorowski, and Y. Bengio, "Blocks and fuel: Frameworks for deep learning," *arXiv preprint arXiv:1506.00619*, 2015.

[27] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. J. Goodfellow, A. Bergeron, N. Bouchard, and Y. Bengio, "Theano: new features and speed improvements," Deep Learning and Unsupervised Feature Learning Workshop, Advances in neural information processing systems (NIPS), Lake Tahoe, NV, 2012.

[28] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: a CPU and GPU math expression compiler," in *Proc. of the Python for Scientific Computing Conference (SciPy)*, vol. 4, 2010, p. 3.

[29] F. Eyben, F. Weninger, F. Groß, and B. Schuller, "Recent developments in openSMILE, the Munich open-source multimedia feature extractor," in *Proc. of the 21st ACM International Conference on Multimedia*, Barcelona, Spain, 2013, pp. 835–838.

[30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

Determining Native Language and Deception Using Phonetic Features and Classifier Combination

Gábor Gosztolya[1,2], *Tamás Grósz*[1], *Róbert Busa-Fekete*[3], *László Tóth*[2]

[1]Institute of Informatics, University of Szeged, Hungary
[2]MTA-SZTE Research Group on Artificial Intelligence, Szeged, Hungary
[3]Department of Computer Science, University of Paderborn, Germany
{ ggabor, groszt, tothl } @ inf.u-szeged.hu, busarobi@upb.de

Abstract

For several years, the Interspeech ComParE Challenge has focused on paralinguistic tasks of various kinds. In this paper we focus on the Native Language and the Deception sub-challenges of ComParE 2016, where the goal is to identify the native language of the speaker, and to recognize deceptive speech. As both tasks can be treated as classification ones, we experiment with several state-of-the-art machine learning methods (Support-Vector Machines, AdaBoost.MH and Deep Neural Networks), and also test a simple-yet-robust combination method. Furthermore, we will assume that the native language of the speaker affects the pronunciation of specific phonemes in the language he is currently using. To exploit this, we extract phonetic features for the Native Language task. Moreover, for the Deception Sub-Challenge we compensate for the highly unbalanced class distribution by instance re-sampling. With these techniques we are able to significantly outperform the baseline SVM on the unpublished test set.

Index Terms: accent recognition, SVM, deep neural networks, AdaBoost.MH, classifier combination, instance sampling

1. Introduction

Computational paralinguistics, a subfield of speech technology, is concerned with the non-linguistic information content of the speech signal. A large number of different paralinguistic tasks exist like detecting laughter [1, 2, 3], emotions [4, 5], estimating the intensity of conflicts [6, 7, 8], and so on. The importance of this area is reflected in the fact that for several years now the Interspeech Computational Paralinguistic Challenge (ComParE) has been held regularly (e.g. [9, 10, 11]).

Here, we describe our approach for the Deception and the Native Language sub-challenges of ComParE 2016 [12]. In the first sub-challenge, deceptive speech has to be identified, while in the Native Language Sub-Challenge, the task is to determine the native language (referred to as L1) of the speaker while he is speaking in another language (L2), this time in English. Following the Challenge guidelines (see [12]), we will omit the description of the tasks, datasets and the method of evaluation, and concentrate on the techniques we applied. We should also note that, unlike in a standard conference study, in this case it makes sense to experiment with several techniques at the same time, which we will indeed do.

Determining the L1 language of the speaker (or *accent recognition*) is a well-studied task within speech technology (see e.g. [13, 14, 15]). Similarly to the standard techniques found in the literature, our approach for the Native Language Sub-Challenge is based on the fact that L1 affects the pronunciation of L2 phonemes. However, to avoid the complexity introduced by i-Vectors, which are the standard solution for this task, we first performed frame-level phoneme identification. Then we examined the frame-level DNN outputs, and extracted different features based on them, as we assumed that they encode valuable information on the pronunciation of the L2 phonemes.

After feature extraction, the next important step is that of classification. However, there are several machine learning algorithms available, which can be considered as state-of-the-art. In our study we examined three such methods, namely Support-Vector Machines (SVM, [16]), DNN and AdaBoost.MH [17]. We also applied a robust classifier output combination method, and, for the Deception Sub-Challenge, instance re-sampling.

The structure of this paper is as follows. First, in Section 2 we describe the way we extracted the phonetic features of the utterances of the Native Language Sub-Challenge, and we also present the results got with these features on the development set. Next, in Section 3 we describe the classifier methods used, how we set their hyper-parameters, the way we combined their outputs, and we then describe the sampling techniques applied. Lastly, in Section 4.2 we present and analyze our test results.

2. Phonetic Feature Extraction from DNN Output for Native Language Determination

Our approach for identifying the native language of the speaker (i.e. Native Language Sub-Challenge) was based on the observation that native language significantly affects the pronunciation of certain L2 phonemes. In speech recognition terms it means that the L1 languages of the speakers may lead to further errors in the ASR output. However, as the reason for these mispronunciations is partly that the speaker has a different native language, they supposedly appear as some tendencies in the phoneme confusion matrix of the phoneme-level ASR output.

Of course, to construct a phoneme confusion matrix we would need a ground truth transcription of the utterances, which was not available (and in an application situation it is expected to be unknown anyway). However, assuming that, in the long term, phonemes in a given (L2) language follow some specific distribution, we can expect that the phonemes in the ASR output will differ from this ideal distribution (that can be obtained from native speakers), and this tendency is L1-dependent.

To this end, first we performed speech recognition on the utterances, using a DNN/HMM hybrid model. This resulted in a phoneme-level posterior probability vector for each frame, and a time-aligned phoneme sequence for each utterance, which served as a basis for the next feature extraction step.

http://dx.doi.org/10.21437/Interspeech.2016-962

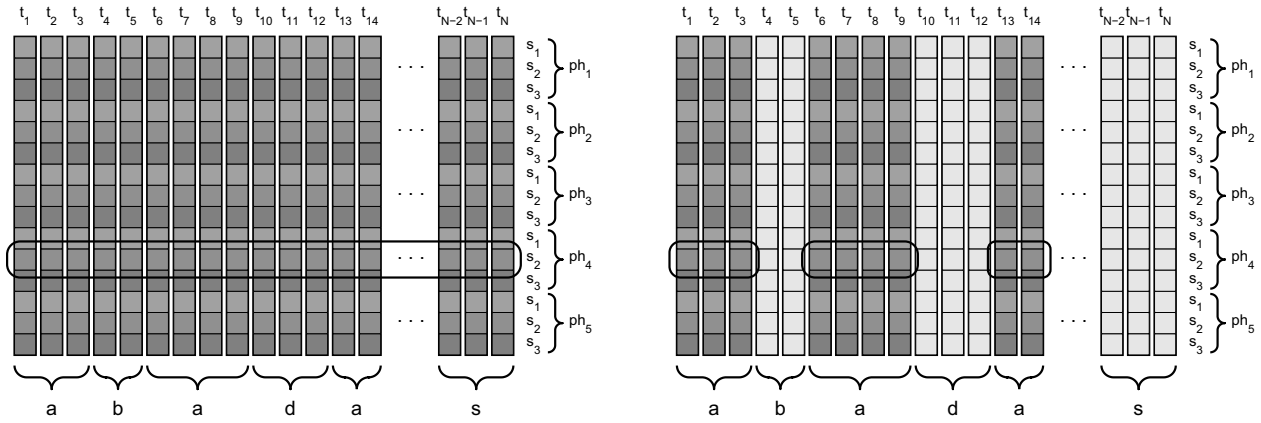

Figure 1: *Feature extraction from the frame-level output of the CI DNN. The vertical bars represent the CI DNN output vectors for frame t_i, belonging to the states $s_1 \dots s_3$ of each possible phoneme ph_j. The circled region displays the region for averaging: in the "DNN stats #1" strategy (left hand side) we take the mean of the frames of the whole utterance, while in the "DNN stats #2" strategy (right hand side) we average out the frames belonging to the same phoneme, according to the time-aligned ASR output.*

2.1. Obtaining a Time-Aligned Phoneme Sequence by ASR

As the acoustic model we utilized a Deep Rectifier Neural Network [18] with 5 hidden layers and 1000 neurons in each layer. The DNN was trained on the fairly large, freely available "TEDLIUM" English speech corpora [19], following the Kaldi recipe [20]. We applied our custom DNN implementation for GPU, which achieved outstanding results on several datasets (e.g. [21, 22]). As acoustic features 12 mel-frequency spectral coefficients ("MFCC", [23]), along with energy, and their first and second order derivatives were used. We used context-independent (CI) phoneme models, based on the results of our preliminary tests. We had 47 phonemes, which, since we used a tri-state setup, led to 141 states overall.

2.2. Feature Extraction

Next, we extracted numerous feature sets from the time-aligned phonetic ASR output. Note that we implicitly made use of the fact that the utterances are of roughly the same length, hence fairly reliable statistics can be calculated from them. Furthermore, besides our feature sets extracted, we also utilized the 6373-item feature set provided by the Challenge organizers (see [12]), which will be referred to as the standard feature set.

The simplest feature set that we constructed consisted of the total number of occurrences and the total duration of occurrences for each phoneme (feature set of "phoneme stats"). With the 47-item phoneme set, this resulted in a quite compact vector with only 94 attributes for each utterance.

We based the following two feature sets on the frame-level output of our acoustic CI tri-state DNN. In the first feature set (referred to as "DNN stats #1"), we calculated the mean and standard deviation of each DNN output utterance-wise, resulting in 282 features overall. This is also a quite compact feature set compared to the 6373-long standard feature vector. Lastly, we calculated the mean and the standard deviation of the DNN outputs, but this time we also used the time-alignment we got previously: we calculated these scores for the frames belonging to the same phoneme in the phoneme-level ASR output ("DNN stats #2"). This led to 13254 attributes overall. The scheme of these feature extraction steps can be seen in Figure 1 above.

Feature set	Size	Acc.	UAR
Standard	6 373	46.9%	47.1%
Phoneme stats	94	45.2%	45.5%
DNN output stats #1	282	54.7%	54.8%
DNN output stats #2	13 254	56.7%	56.9%
Standard + phoneme stats	6 467	50.0%	50.2%
Standard + DNN stats #1	6 655	53.4%	53.5%
Standard + DNN stats #2	19 627	61.1%	61.3%
All features	20 003	62.9%	63.1%
ComParE baseline [12]		44.9%	45.1%

Table 1: *Accuracy and UAR scores got by using the different feature sets with SVM applied on the development set of the Native Language Sub-Challenge.*

2.3. Results

We evaluated the different feature sets by applying Support-Vector Machines (SVM, [16]) with a linear kernel, using the libSVM library [24]. The value of C was tested in the range $10^{\{-5,\dots,1\}}$. The results for this can be seen in Table 1 above. Our scores for the standard, 6373-item feature set are slightly above the baseline scores provided in the Challenge paper for two reasons: we used libSVM instead of Weka, and we performed an energy-based volume normalization on the input utterances.

What was quite surprising is that by just using the total number and duration of the phonemes (only 94 attributes overall), we managed to slightly outperform the baseline accuracy scores, and almost match our scores got by using the standard, 6373-long feature set. This is really unexpected for such a compact and simple feature set, and in our opinion it indicates that attempting to identify the native language of the speaker based on the phonetic-level ASR output is a viable approach. By using the features extracted from the frame-level DNN outputs, we even managed to significantly outperform the baseline scores; and by combining them with the standard feature set we achieved accuracy scores significantly above the baseline score. In fact, combining all the above-mentioned features led to a UAR score of 63.1% on the development set, which means a 32% relative error reduction compared to the baseline.

3. Classification Methods and Refinements

After describing the feature extraction step, we turn to the next phase: that of classification. Besides describing the classification algorithms utilized for both sub-challenges, we will explain the other techniques applied: DNN model output aggregation, the sampling methods and a classifier combination approach.

3.1. Classifier Methods

We utilized three methods for classification. First, we applied Support-Vector Machines [16] with a linear kernel. The value of C was again tested in the range $10^{\{-5,\ldots,1\}}$. Then for both tasks we trained 10 randomly initialized Deep Rectifier Neural Networks (DNN, [18]) having three fully connected hidden layers, each containing 100 and 500 neurons for the Deception and the Native Language sub-challenges, respectively. We used our custom implementation, originally developed for phoneme classification, by which we achieved a good performance in the previous ComParE Challenges [22, 25]. Finally, we applied AdaBoost.MH [17, 26] using stumps as base learners, because we also got good results previously with it (e.g. [5, 25, 27]). The meta-parameters of each method (C for SVM, number of iteration for AdaBoost.MH) were set using the development set for both sub-challenges.

3.2. DNN Model Output Aggregation

Training a neural network is a non-deterministic procedure due to the random initial weight values. To reduce this effect of uncertainty, it is common to train several models with the same parameters, and aggregate their outputs in some way. Perhaps the most commonly used way of aggregating outputs is via simple majority voting: we choose the class label which was supported by the largest number of models. It can be readily applied to neural networks, but it is well known that DNNs are able to produce accurate posterior scores; unfortunately, this information is lost during simple majority voting. So we used the technique called *probabilistic voting* [25]: for each example and each class we averaged out the output posterior values of all models, and chose the class where this value was the highest.

3.3. Instance Re-Sampling

Most machine learning algorithms are sensitive to class imbalances, and tend to behave inaccurately on classes having only a few examples. Since in the Deception Sub-Challenge the distribution of the D and ND classes was unbalanced to the extent that even the baseline included the upsampling of the D class (i.e. using the same training examples several times), we decided to experiment with sampling methods for this sub-challenge. For DNN, we applied the sampling method called probabilistic sampling [28, 29]. It is a simple two-step sampling scheme: first we select a class, then randomly pick a training sample from the samples of this class. Selecting a class can be viewed as sampling from a multinomial distribution after we assign a probability to each class:

$$P(c_k) = \lambda \frac{1}{K} + (1 - \lambda) P_0(c_k), \qquad (1)$$

where $P_0(c_k)$ is the prior possibility of class c_k, K is the number of classes and $\lambda \in [0, 1]$ is a parameter. If λ is 1, then we get a uniform distribution over the classes, and with $\lambda = 0$ we get the original class distribution. Choosing a value between 0 and 1 for λ allows us to linearly interpolate between these two distributions.

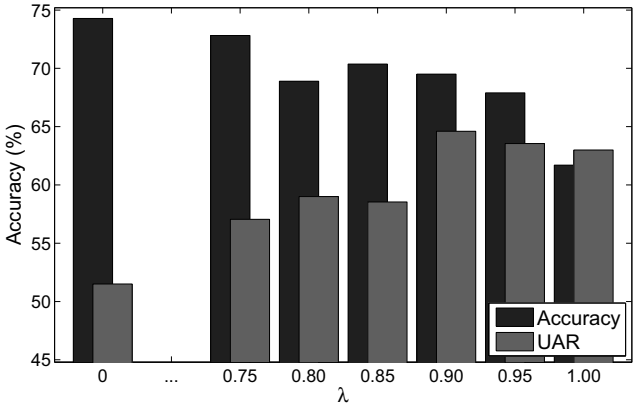

Figure 2: *Accuracy and UAR scores got on the development set of the Deception Sub-Challenge for DNN probabilistic sampling.*

The value of the λ parameter was set on the development set. The accuracy and UAR scores we got can be seen in Fig. 2. It is clear that DNNs optimize for accuracy by default ($\lambda = 0$); by increasing the value of λ, the UAR scores tend to rise, while accuracy tends to drop. In the end we used the optimal value of $\lambda = 0.9$ in our later experiments.

For SVM and AdaBoost.MH, we did not have the option of using probabilistic sampling, as we utilized standard libraries (libSVM [24] and multiboost [26]). Therefore we experimented both with upsampling and downsampling (i.e. not using all the training instances) to balance the class distribution. We trained several models for AdaBoost.MH, so we selected the training set randomly for each training.

M.L. Method	Sampling	Acc.	UAR
SVM	—	69.8%	58.1%
	Upsampling	67.9%	62.6%
	Downsampling	62.8%	62.8%
DNN	—	74.3%	51.5%
	Probabilistic	69.5%	64.7%
AdaBoost.MH	—	72.6%	57.1%
	Upsampling	72.2%	57.3%
	Downsampling	66.7%	62.5%
ComParE baseline [12]		70.2%	61.1%

Table 2: *Accuracy and UAR scores got by using the different classifier and sampling methods on the development set of the Deception Sub-Challenge.*

Table 3 shows the accuracy and UAR scores achieved by using the different re-sampling strategies. It is cleat that re-sampling greatly improved the performance of each method; it was a bit surprising, though, that for AdaBoost.MH, downsampling (i.e. discarding training examples) worked better. Although for SVM, downsampling was slightly better that upsampling UAR-wise, due to the much higher accuracy score we decided that we should use upsampling later. In the following experiments carried out on the Deception Sub-Challenge, we always applied instance re-sampling during classifier model training.

Method	Dev.		Test	
	Acc.	UAR	Acc.	UAR
SVM (upsampling)	67.9%	62.6%	—	—
DNN (prob. sampling)	69.5%	**64.7%**	69.8%	**68.6%**
AdaBoost.MH (downs.)	66.7%	62.5%	—	—
SVM + DNN	69.3%	64.0%	**70.6%**	67.7%
SVM + AdaBoost.MH	67.5%	62.8%	—	—
DNN + AdaBoost.MH	66.9%	61.9%	—	—
All three methods	67.7%	62.9%	—	—
ComParE baseline [12]	**70.2%**	61.1%	—	68.3%

Table 3: *Accuracy and UAR scores got by using the different classifier methods on the Deception Sub-Challenge.*

Method	Dev.		Test	
	Acc.	UAR	Acc.	UAR
SVM	62.9%	63.1%	—	—
DNN	64.5%	64.5%	62.9%	62.9%
AdaBoost.MH	69.3%	69.3%	69.2%	69.2%
SVM + DNN	64.5%	64.7%	—	—
SVM + AdaBoost.MH	70.1%	70.1%	—	—
DNN + AdaBoost.MH	**70.7%**	**70.7%**	**70.0%**	**70.1%**
All three methods	69.0%	69.1%	—	—
ComParE baseline [12]	44.9%	45.1%	—	47.5%

Table 4: *Accuracy and UAR scores got by using the different classifier methods on the Native Language Sub-Challenge.*

3.4. Classifier Combination

It is well known that a good combination of the original classifiers may reinforce their advantages. This may explain the interest in classifier combination techniques in several areas of Artificial Intelligence (e.g. [30, 31]), and also in speech recognition [32, 33]. In our previous experiments (such as our contribution submitted for the Eating Sub-Challenge for ComParE 2015 [11]), we found that a quite robust way of combining different kinds of classifiers is to aggregate their class-wise posterior scores. We will now also apply this strategy here.

We combined our three classifier methods by taking the mean of their posteriors for each example and class, and chose the class for each example which had the highest posterior score. Getting posteriors is quite easy for SVM; for DNNs we used the aggregated likelihood estimates of several trained models (see Section 3.2). For AdaBoost.MH, we trained 1000 models independently, and for each example we counted how many models voted for each class; these values were then used as posterior estimates after normalizing their sum to one. To calibrate the different distribution of likelihoods for the three methods, we first normalized the vote vectors of the different classification methods so as to have the same standard deviation.

4. Results

4.1. Deception Sub-Challenge

Table 3 shows the values achieved by using the different classifier methods on the development and on the test sets of the Deception Sub-Challenge. We can see that, although we were able to outperform baseline SVM on the development set, this was true for the test set only to a limited extent: with DNN alone, we managed to get a higher UAR value than the baseline by a mere 0.3%, but this is quite different from the 3.6% improvement measured on the development set. And although combining DNN with SVM led to a slight gain in accuracy, it led to a slight drop in the UAR value. In our opinion this can be attributed to the small size and the unbalanced nature of the dataset. Of course, the fact that we used a different SVM implementation and a slightly different instance re-sampling method might also affect the scores we obtained.

4.2. Native Language Sub-Challenge

Table 4 shows the results we got by applying the classifier methods on the development and on the test sets of the Native Language Sub-Challenge. As input we used the extended feature set described in Section 2. Although the construction of this 20003-long feature set was achieved by utilizing SVM, we got better scores on it with DNNs. This is probably because this task is fairly large by computational paralinguistics standards: the training set consists of 3300 utterances, and the development and test sets are roughly 1000 recordings long each. With such a high number of samples a DNN can be trained quite reliably. However, AdaBoost.MH performed even better, and by a large amount. In our opinion this is probably due to the diversity of this feature set. Recall that the 20003 attributes consist of the standard paralinguistic features, phoneme occurrence counts, and means and standard deviations of DNN outputs. Our hypothesis is that different normalization techniques are optimal for these different kinds of attributes, while we always used standardization both for SVM and DNN. AdaBoost.MH, however, does not require any kind of normalization, hence it is not affected by a suboptimal normalization procedure.

As regards classifier combination, we can see that if we combine AdaBoost.MH with either SVM or DNN, the accuracy and UAR scores improve. However, by combining all three methods, these values drop, which is probably due to the fact that, in this case, AdaBoost.MH has only a weight of $1/3$.

Overall, on the test set we observed pretty similar tendencies to those seen on the development set. Even combining DNN with AdaBoost.MH yielded a 1% improvement in both accuracy values. In the end we achieved a UAR value of 70.1%, which is way above the 47.5% score of baseline SVM.

5. Conclusions

In this study, submitted for the Computational Paralinguistic Challenge (ComParE) of Interspeech 2016, we focused on to two classification tasks: the Deception and the Native Language sub-challenges. We utilized three different classifier methods (SVM, DNN and AdaBoost.MH) and also tried out a robust classifier output combination approach. Furthermore, we utilized instance re-sampling techniques for the Deception Sub-Challenge, while we extracted various acoustic features for the Native Language Sub-Challenge. The accuracy scores of our methods revealed that instance re-sampling is essential for achieving competitive scores in the Deception Sub-Challenge. For the Native Language Sub-Challenge, however, we found that our feature extraction approach is a viable one for L1 language determination: with quite basic features such as the number of the different phonemes found in the utterances, we were able to practically match the baseline score got using the standard 6373-long feature vector. By extracting further features we outperformed the baseline method by a large amount even on the unpublished test set, although it seems that the way of normalization significantly affects the accuracy values.

6. References

[1] T. Neuberger and A. Beke, "Automatic laughter detection in spontaneous speech using GMM–SVM method," in *Proceedings of TSD*, 2013, pp. 113–120.

[2] G. Gosztolya, "On evaluation metrics for social signal detection," in *Proceedings of Interspeech*, Dresden, Germany, Sep 2015, pp. 2504–2508.

[3] R. Gupta, K. Audhkhasi, S. Lee, and S. S. Narayanan, "Speech paralinguistic event detection using probabilistic time-series smoothing and masking," in *Proceedings of Interspeech*, 2013, pp. 173–177.

[4] S. L. Tóth, D. Sztahó, and K. Vicsi, "Speech emotion perception by human and machine," in *Proceedings of COST Action*, Patras, Greece, 2012, pp. 213–224.

[5] G. Gosztolya, R. Busa-Fekete, and L. Tóth, "Detecting autism, emotions and social signals using AdaBoost," in *Proceedings of Interspeech*, Lyon, France, Aug 2013, pp. 220–224.

[6] H. Kaya, T. Özkaptan, A. A. Salah, and F. Gürgen, "Random discriminative projection based feature selection with application to conflict recognition," *IEEE Signal Processing Letters*, vol. 22, no. 6, pp. 671–675, 2015.

[7] O. Räsänen and J. Pohjalainen, "Random subset feature selection in automatic recognition of developmental disorders, affective states, and level of conflict from speech," in *Proceedings of Interspeech*, Lyon, France, Sep 2013, pp. 210–214.

[8] G. Gosztolya, "Conflict intensity estimation from speech using greedy forward-backward feature selection," in *Proceedings of Interspeech*, Dresden, Germany, Sep 2015, pp. 1339–1343.

[9] B. Schuller, S. Steidl, A. Batliner, A. Vinciarelli, K. Scherer, F. Ringeval, M. Chetouani, F. Weninger, F. Eyben, E. Marchi, H. Salamin, A. Polychroniou, F. Valente, and S. Kim, "The Interspeech 2013 Computational Paralinguistics Challenge: Social signals, Conflict, Emotion, Autism," in *Proceedings of Interspeech*, 2013.

[10] B. Schuller, S. Steidl, A. Batliner, J. Epps, F. Eyben, F. Ringeval, E. Marchi, and Y. Zhang, "The INTERSPEECH 2014 computational paralinguistics challenge: Cognitive & physical load," in *Proceedings of Interspeech*, 2014, pp. 427–431.

[11] B. Schuller, S. Steidl, A. Batliner, S. Hantke, F. Hönig, J. R. Orozco-Arroyave, E. Nöth, Y. Zhang, and F. Weninger, "The INTERSPEECH 2015 computational paralinguistics challenge: Nativeness, Parkinson's & eating condition," in *Proceedings of Interspeech*, 2015.

[12] B. Schuller, S. Steidl, A. Batliner, J. Hirschberg, J. K. Burgoon, A. Baird, A. Elkins, Y. Zhang, E. Coutinho, and K. Evanini, "The Interspeech 2016 computational paralinguistics challenge: Deception, sincerity & native language," in *Proceedings of Interspeech*, San Francisco, USA, 2016.

[13] A. DeMarco and S. Cox, "Native accent classification via i-vectors and speaker compensation fusion," in *Proceedings of Interspeech*, Lyon, France, Sep 2013, pp. 1472–1476.

[14] V. Hautamäki, S. M. Siniscalchi, H. Behravan, V. M. Salerno, and I. Kukanov, "Boosting universal speech attributes classification with deep neural network for foreign accent characterization," in *Proceedings of Interspeech*, Dresden, Germany, Sep 2015, pp. 408–412.

[15] H. Behravan, V. Hautamäki, and T. Kinnunen, "Factors affecting i-vector based foreign accent recognition: a case study in spoken Finnish," *Speech Communication*, vol. 66, pp. 118–129, 2015.

[16] B. Schölkopf, J. Platt, J. Shawe-Taylor, A. Smola, and R. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[17] R. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated predictions," *Machine Learning*, vol. 37, no. 3, pp. 297–336, 1999.

[18] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier networks," in *Proceedings of AISTATS*, 2011, pp. 315–323.

[19] A. Rousseau, P. Deléglise, and Y. Esteve, "TED-LIUM: an Automatic Speech Recognition dedicated corpus," in *Proceedings of LREC*, 2012, pp. 125–129.

[20] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlíček, Y. Qian, P. Schwarz, J. Silovský, G. Stemmer, and K. Veselý, "The Kaldi speech recognition toolkit," in *Proceedings of ASRU*, Hilton Waikoloa Village, Big Island, Hawaii, US, Nov 2011, pp. 1339–1343.

[21] L. Tóth, "Phone recognition with hierarchical Convolutional Deep Maxout Networks," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2015, no. 25, pp. 1–13, 2015.

[22] T. Grósz, R. Busa-Fekete, G. Gosztolya, and L. Tóth, "Assessing the degree of nativeness and Parkinson's condition using Gaussian Processes and Deep Rectifier Neural Networks," in *Proceedings of Interspeech*, Dresden, Germany, Sep 2015, pp. 1339–1343.

[23] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*. Prentice Hall, 1993.

[24] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 1–27, 2011.

[25] G. Gosztolya, T. Grósz, R. Busa-Fekete, and L. Tóth, "Detecting the intensity of cognitive and physical load using AdaBoost and Deep Rectifier Neural Networks," in *Proceedings of Interspeech*, Singapore, Sep 2014, pp. 452–456.

[26] D. Benbouzid, R. Busa-Fekete, N. Casagrande, F.-D. Collin, and B. Kégl, "MultiBoost: a multi-purpose boosting package," *Journal of Machine Learning Research*, vol. 13, pp. 549–553, 2012.

[27] G. Gosztolya, "Is AdaBoost competitive for phoneme classification?" in *Proceedings of CINTI (IEEE)*, Budapest, Hungary, Nov 2014, pp. 61–66.

[28] L. Tóth and A. Kocsor, "Training HMM/ANN hybrid speech recognizers by probabilistic sampling," in *Proceedings of ICANN*, 2005, pp. 597–603.

[29] S. Lawrence, I. Burns, A. Back, A. Tsoi, and C. Giles, "Chapter 14: Neural network classification and prior class probabilities," in *Neural Networks: Tricks of the Trade*. Springer, 1998, pp. 299–313.

[30] B. Plessis, A. Sicsu, L. Heutte, E. Menu, E. Lecolinet, O. Debon, and J.-V. Moreau, "A multi-classifier combination strategy for the recognition of handwritten cursive words," in *Proceedings of ICDAR*, 1993, pp. 642–645.

[31] K. Yu, X. Jiang, and H. Bunke, "Lipreading: A classifier combination approach," *Pattern Recognition Letters*, vol. 18, no. 11–13, pp. 1421–1426, 1997.

[32] L. Felföldi, A. Kocsor, and L. Tóth, "Classifier combination in speech recognition," *Periodica Polytechnica, Electrical Engineering*, vol. 47, no. 1, pp. 125–140, 2003.

[33] G. Gosztolya and J. Dombi, "Applying representative uninorms for phonetic classifier combination," in *Proceedings of MDAI*, Tokyo, Japan, Oct 2014, pp. 182–191.

INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

Native Language Identification Using Spectral and Source-Based Features

Avni Rajpal[1], Tanvina B. Patel[1], Hardik B. Sailor[1], Maulik C. Madhavi[1], Hemant A. Patil[1] and Hiroya Fujisaki[2]

[1] Dhirubhai Ambani Institute of Information and Communication Technology (DA-IICT), India
[2]University of Tokyo, Japan

{avni_rajpal, tanvina_bhupendrabhai_patel, sailor_hardik, maulik_madhavi, hemant_patil}@daiict.ac.in fujisaki@alum.mit.edu

Abstract

The task of native language (L1) identification from non-native language (L2) can be thought of as the task of identifying the common traits that each group of L1 speakers maintains while speaking L2 irrespective of the dialect or region. Under the assumption that speakers are L1 proficient, non-native cues in terms of segmental and prosodic aspects are investigated in our work. In this paper, we propose the use of longer duration cepstral features, namely, Mel frequency cepstral coefficients (MFCC) and auditory filterbank features learnt from the database using Convolutional Restricted Boltzmann Machine (ConvRBM) along with their delta and shifted delta features. MFCC and ConvRBM gave accuracy of *38.2%* and *36.8%*, respectively, on the development set provided for the ComParE 2016 Nativeness Task using Gaussian Mixture Model (GMM) classifier. To add complementary information about the prosodic and excitation source features, phrase information and its dynamics extracted from the log(F_0) contour of the speech was explored. The accuracy obtained using score-level fusion between system features (MFCC and ConvRBM) and phrase features were *39.6%* and *38.3%*, respectively, indicating that phrase information and MFCC capture complementary information than ConvRBM alone. Furthermore, score-level fusion of MFCC, ConvRBM and phrase improves the accuracy to *40.2*%.

Index Terms: Shifted delta cepstrum, Convolutional Restricted Boltzmann Machine, F_0, Accent, Phrase.

1. Introduction

In general, multilingual speakers lack thorough acquisition of second language (L2) and speech from a particular group of non-native speakers show common traits such as distinct 'foreign accent' and typical pronunciation errors [1], [2]. The task of native language (L1) identification aims at identifying such commonalities from spontaneous speech that can be used to identify the mother tongue of English (L2) speakers. The feasibility of proposed Native Language Identification (NLID) task in this challenge lies in phenomenon of *prosodic transfer* from L1 to L2. Moreover, the dialectal differences of learner's L1 in aspects of prosodic transfer from L1 to L2 should be known as observed by Fujisaki and others [3], [4]. NLID system can be used in parallel with computer-aided language learning systems to provide L1- specific training program, also for automated speech assessment systems, reading tutors, adaptation in ASR, speaker forensics etc. [5], [6].

Non-native speakers frequently maintain a foreign accent and inadvertently carry phonemic details from L1 to L2 [7]. In addition, non-native speech typically includes more disfluencies than native speech and is characterized by a lower speech rate [2], [8]. This indicates the influence of L1 over L2, both in terms of prosody as well as segmental aspects [1]. However, the degree of influence, depends on the amount of L1 used and the proficiency of L2 [2], [9]. The above difficulties are very prominent in Native Language Speech Corpus (NLSC), since the recordings are from the TOEFL iBT® assessment given by non-native speakers under examination conditions [10]. Thus, nativeness of speaker can be identified by studying the acoustic and prosodic aspects that remain native-like or become prominent while speaking L2.

In this paper, we intend to explore both acoustic and prosodic features for L1 identification. We propose to use acoustic features obtained from the auditory filterbank learnt from the speech signals using Convolutional Restricted Boltzmann Machine (ConvRBM). In majority of the experiments related to evaluation of degree of nativeness of the speaker and L2 (particularly English) acquisition, native English speaking listeners are used for human scoring [9], [11]. This is because, it is assumed that non-native cues can be easily identified by the native English speaking listeners that are having well-established linguistic knowledge of speech. This idea to imitate the hearing mechanism of native English speakers motivates us to learn features not only from NLSC but also from other databases, which contains speech recordings from native English speakers. In this work, we have used WSJ0 database having recording from clean environment and AURORA 4 multi condition database [12], [13].

The time interval or frame size for feature extraction is known to capture different segmental information as per the window chosen for analysis. For longer analysis window, the cepstral features contain information about formants structure and its dynamics that can reflect the movement and position of vocal and nasal articulators [14]. Moreover, a time interval of 90 ms is suggested by Furui [15] to preserve the transitional information associated with changes from one phoneme to another. Moreover, Soong and Rosenberg [16] proposed 100 to 160 ms time interval, to obtain good estimates of the trend of spectral transitions between the syllables. Thus, in order to extract segmental information, the proposed features for L1 identification task are extracted over longer window durations. Furthermore, the dynamic features (i.e., delta (Δ), delta-delta ($\Delta\Delta$)) and Shifted Delta Cepstral (SDC) features were explored to capture the spectral dynamics. To gain the advantage of prosodic cues for L1 identification task, prosodic features such as fundamental frequency (F0), phrase and accent along with their dynamics are considered. The results indicate that features from WSJ0 and AURORA 4 can better identify L1 than features from NLSC. Moreover, phrase with its dynamics was found to capture complementary information with respect to spectral features and was used to further improve the accuracy of the classification system.

http://dx.doi.org/10.21437/Interspeech.2016-1100

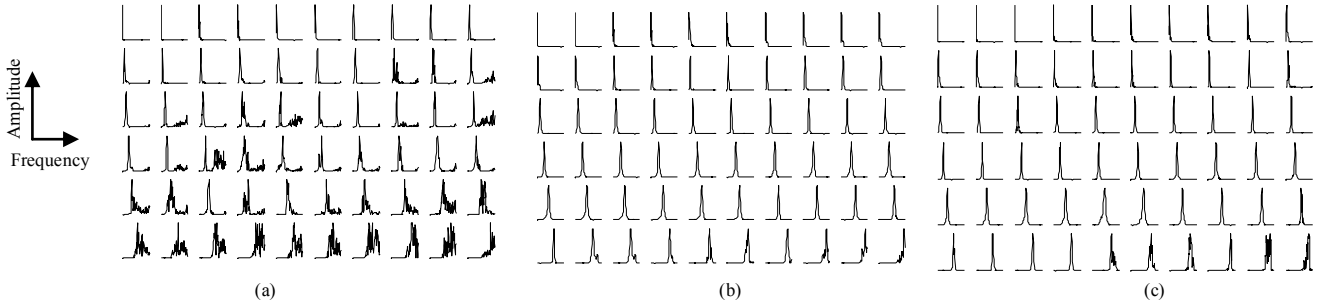

Figure 1: *Subband filters in frequency-domain trained on (a) NLSC, (b) WSJ0 and (c) AURORA 4.*

2. Acoustic Features

The NLID task requires features that capture idiosyncrasies of a particular L1 that remain while speaking L2. Thus, for this task, both spectral and excitation source features are explored.

2.1. Convolutional Restricted Boltzmann Machine

ConvRBM is an unsupervised probabilistic model used to learn auditory filterbanks directly from speech signals. The inference is based on sampling of hidden units from noisy rectified linear units (NReLU) [16]. Training is based on contrastive divergence and parameters are updated using gradient descent. The training of the model and feature extraction is similar to our very recent work reported in [16]. The model is trained using three databases, namely, NLSC, WSJ0 and AURORA 4. The subband filters learnt on these three databases are shown in Figure 1. To analyze the subband filters, they were arranged according to their center frequency. As shown in Figure 1, we can see that many subband filters trained on NLSC are different than the subband filters trained on native English speakers' databases (WSJ0 and AURORA 4). As seen from Figure 1(a) many of the filters are not localized specifically in mid-center frequency range compared to subband filters in Figure 1(b) and Figure 1(c). This indicates that speaker-specific traits are learnt by NLSC, which may be due to interference of L1 over L2 [18].

2.2. Shifted Delta Cepstral (SDC) Features

SDC are long-term temporal features that capture the spectral dynamics of speech via cepstral trajectory in *N*-dimensional (dim) feature space. SDC are known to improve the performance of speaker recognition and language recognition systems [14], [19], [20]. SDC has pseudo-prosodic behavior, since in each frame; it captures the temporal dynamics of the articulators present in the next frames [14]. For a given *N*-dim cepstral feature vector, SDC vector at n^{th} frame is obtained by concatenating *k* blocks of delta coefficients and is given by

$$\Delta c(n+iP) = \sum_{d=-D}^{D} dc(n+iP+d) \Big/ \sum_{d=-D}^{D} d^2 \quad , \qquad (1)$$

where *D* is time advance and delay for delta computation, *P* is time shift between consecutive blocks, *i*=0 to *k-1* blocks to be concatenated [14].

2.3. Excitation Source-Based Parameters

The F0 contour of speech is known to capture both linguistic and non-linguistic information (speech prosody). This information is embedded in the low frequency variations (LFV) and high frequency variations (HFV) of the F0 contour [21]. The state-of-the-art Fujisaki model decomposes F0 in log- domain into phrase and accent components extracted from LFV and HFV, respectively [21], [22]. In this work, instead of extracting components from their respective commands, we use the LFV and HFV directly along with their dynamics for NLID task. We have used the Zero Frequency Filtering (ZFF) method to estimate the F0 contour [23]. The negative-to-positive zero-crossings of the zero frequency filtered signal gives an estimate of the Glottal Closure Instant (GCIs) Thereafter, the F0 contour is obtained from the GCI locations. In order to estimate HFV and LFV, log (F0) is processed, i.e., the intermediate values of log (F0) for unvoiced speech regions and short pauses are interpolated and microprosodic variations due to individual speech sounds (such as plosive, fricatives, etc.) are smoothed out. The interpolated and spline fitted log (F0) contour is then passed through a highpass filter with a stop frequency at 0.5 Hz to obtain HFV. The HFV is then subtracted from the processed log (F0) contour yielding LFV. LFV consists of phrase component and Fb (speaker-specific constant) [21], [24]. Fb is set to the overall minimum of the LFV and is subtracted from it to give phrase component.

3. Experimental Results

3.1. Speech Corpus

NLSC is a corpus of non-native English speech consisting of spoken response provided during a high-stakes global assessment of English language proficiency, the Test of English as a Foreign Language (TOEFL iBT®). It contains *5,132* spoken responses. For the ComParE 2016 Nativeness Task, the NLSC corpus is partitioned as follows: *3,300* responses (*64 %*, approximately *41.3* hours) will be used as training data, *965* responses (*19 %*, approximately *12.1* hours) will be used as development data, and *867* responses (*17 %*, approximately *10.8* hours) will be used as the test data. The complete details of the corpus are mentioned in [10].

3.2. System Building

In this paper, we use Gaussian Mixture Model (GMM) with *128* mixtures for performing the classification of input speech into given L1 classes. At the training stage, *11* GMM models each corresponding to separate L1 class were built. The models were trained using *300* instances of each L1 class available from the training set. Final scores are represented in terms of log-likelihood (LLK). The decision for the predicted class for the test speech is taken by calculating LLK for each GMM class. The class associated with the GMM that has maximum LLK is the predicted class.

3.3. Performance Measure

In this paper, Unweighted Average Recall (UAR) and weighted average recall (WA) ('conventional accuracy') is used as the evaluation measure [10]. UAR is given by [25]:

$$UAR = \frac{1}{N}\sum_{i=1}^{N}(C_{ii}/\sum_{j=1}^{N}C_{ij}), \quad (2)$$

where C_{ij} is the number of instances of class i in contingency matrix C that are classified as class j with N as the total number of the classes. To utilize complementary information, score-level fusion of features is obtained using (3).

$$LLK_{fused} = \sum_{i=1}^{N}\alpha_i LLK_{feati}, \quad (3)$$

where LLK_{feati} is the LLK score of i^{th} feature, α_i decides the weight of the scores such that $\sum_{i=1}^{N}\alpha_i = 1$. Before score-level fusion, the scores were normalized using *tanh-estimators* [26]. This technique was chosen for normalization since it is robust and is not sensitive to the outliers [26]. The normalization is given by:

$$LLK'_k = \frac{1}{2}\left\{\tanh\left(0.01\left(\frac{LLK_k - \mu}{\sigma}\right)\right) + 1\right\}, \quad (4)$$

where LLK_k and LLK'_k are the original and normalized score of the k^{th} class, μ and σ are the global mean and standard deviation of the scores, respectively.

3.4. Effect of Window Length of Spectral Features

In this work, we consider *13*-dim NLSC, WSJ and AURORA features. In addition, we also used *13-dim* MFCC as spectral features. To capture the short-time dynamic (DYN) information, these static spectral features are concatenated with Δ and ΔΔ to form *39-dim* MFCC+DYN, NLSC+DYN, WSJ+DYN and AURORA+DYN. Furthermore, to capture long-term temporal spectral dynamics, the *13 dim* spectral features were also combined with SDC resulting in *39-dim,* MFCC+SDC, NLSC+SDC, WSJ+SDC and AURORA+SDC features. The parameters for SDC are set *to N=13, D=2, P=2, k=2* in (1). The *13-dim* spectral features are extracted over various window durations (*25 ms, 100 ms, 150 ms* and *200 ms*). This is to quantify the fact that in spectral features, long duration window is useful in capturing the prosodic trends. Therefore, the window that is suitable for NLID task is investigated. As shown in Figure 2, WA on the development set increases with increase in window length. Moreover, *150 ms* and *100 ms* are ideal window lengths for both DYN and SDC features.

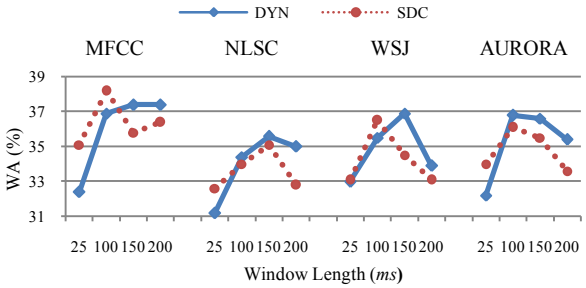

Figure 2: *Plot of window length (ms) vs. WA (in %).*

3.5. Effect of Dynamics of Source Features

In order to capture the temporal dynamic information contained in log(F_0), Phrase and HFV (denoted as Accent) features, their, Δ, ΔΔ and ΔΔΔ (3Δ) components were extracted. Figure 3 shows the effect of dynamics of source-based features on the classification accuracy. It can be observed from Figure 3 that dynamic features of log(F_0) and phrase contain significant information about L1 of the speakers. However, the dynamic information undermines the performance of accent features.

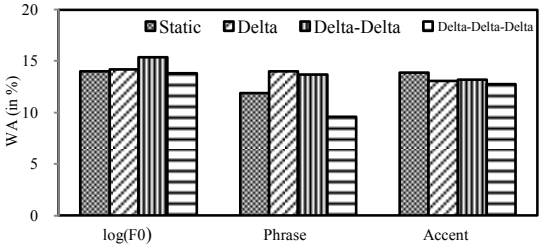

Figure 3: *Plot of dynamic features vs. classification accuracy for source features.*

3.6. Results on the Development Set

The best results from Figure 2 are summarized in Table 1 and it is observed that among data-driven approaches, WSJ and AURORA perform equally well with accuracy of *36.9 % and 36.8 %,* respectively, compared to the NLSC features. The possible reason of underperformance of NLSC could be that the speaker-specific traits learnt by the model were not specific to an individual group of non-native speakers, instead generalized statistical (invariant) properties corresponding to entire database was learnt. Hence, the models learnt on NLSC might not be optimal for individual non-native speaker group. However, the model trained on native English speakers' database represents an *optimal auditory code* [17], [27] that captured the common traits of non-native speakers. From Table 1, we also observe that the accuracy of handcrafted MFCC+SDC features is highest, i.e., it performs better than our proposed data-driven features (WSJ and AURORA) specifically with SDC. The possible reason could be that the model trained on WSJ0 and AURORA 4 does not incorporate any information from NLSC database. However, there is not huge difference in the performance of WSJ and AURORA, which could be improved in future by adapting models trained on WSJ0 and AURORA 4 to specific non-native speaker groups in NLSC database [28]. Among the source-based features, Table 1 shows that, log(F_0) has highest accuracy than phrase and accent features when used independently for classification. However, accuracy of log(F_0) feature significantly improves when accent component is fused with it at feature-level. Overall, it can be observed from Table 1 that the prosodic features based on Fujisaki model alone are not performing well for the proposed NLID task. Authors believe that this is primarily due to lack of dialectal knowledge about L1 speaker and its prominent effect on L2 [3], [4].

Table 1: *WA and UAR of spectral and source features*

Spectral* Features	WA (%)	UAR (%)	Source Features	WA (%)	UAR (%)
MFCC+DYN_150	37.4	37.7	logF_0	14.0	14.1
MFCC+SDC_100	**38.2**	**38.4**	Phrase	11.9	11.9
NLSC+DYN_150	35.6	35.8	Accent	13.9	13.9
NLSC+SDC_150	35.1	35.2	logF_0+Phrase	14.1	14.1
WSJ+DYN_150	**36.9**	**37.1**	logF_0+Accent	**14.9**	**15.1**
WSJ+SDC_100	36.5	36.6	Phrase+Accent	13.1	13.0
AURORA+DYN_100	36.8	**37.1**	logF_0+Phrase+Accent	14.8	14.8
AURORA+SDC_100	36.1	36.3			

* MFCC+DYN_150: Feature_Window Length (*ms*).

The best performing data-driven features (WSJ+DYN_150 and AURORA+DYN_100) are fused with the spectral MFCC and source-based features. Among all the source-based features, i.e., log(F_0), phrase and accent (with all its

combination and dynamic features), phrase+3Δ showed the improvement in accuracy of spectral features after score-level fusion. The selected features for which the better performance is obtained after fusion are shown in Table 2.

Table 2: *WA and UAR (in %) after score-level fusion*

Spectral Features	Fused Feature	α_2	WA (%) after fusion	UAR (%) after fusion
AURORA+DYN_100	MFCC+SDC_100	**0.6**	**39.6**	**39.8**
WSJ+DYN_150	MFCC+SDC_100	0.6	38.9	39
AURORA+DYN_100	Phrase+ 3 Δ	0.2	**38.3**	**38.5**
WSJ+DYN_150	Phrase+ 3 Δ	0.1	37.7	37.9

From Table 2, it can be observed that after fusion WA of both AURORA and WSJ features significantly improved. However, the improvement was more significant for AURORA even though they performed equally well without fusion. The possible reason is that AURORA 4 contains utterances from clean, noisy and mismatched conditions similar to NLSC database in which mismatched conditions was observed for few utterances. Thus, the model learnt using AURORA 4 is more generalized compared to the clean WSJ database. Thus, both at spectral-level and source-level, AURORA+DYN_100 is the common feature which give best fusion accuracy with MFCC+SDC_100 and phrase+3Δ. To further analyze the reason on improvement after fusion with MFCC+SDC_100 and phrase+3Δ features, the individual performance of the languages is considered (as shown in Table 3).

Table 3: *Effect of fusion on accuracy of individual L1*

	MFCC+ SDC_100	AURORA + DYN_100	AURORA +Phrase +3 Δ	AURORA+DYN_100+MFCC+ SDC_100	AURORA+DYN_100 + MFCC+SDC_100+ Phrase3 Δ
ARA	38	36	31	**41**	**37**
CHI	54	55	51	**55**	54
FRE	30	28	**45**	**28**	**38**
GER	**52**	48	46	**55**	**56**
HIN	33	34	37	**42**	**36**
ITA	**44**	37	31	**43**	**39**
JPN	34	38	36	35	35
KOR	41	44	38	42	43
SPA	**30**	19	**35**	**29**	**34**
TEL	**42**	40	36	**40**	**42**
TUR	**25**	29	**37**	28	**29**

It was observed from Table 3 that after fusion with MFCC, the accuracy of all languages showed improvement (except JPN, KOR and TUR). On the contrary, phrase+3Δ showed significant improvement for FRE, SPA and TUR languages. This indicates that phrase+3Δ carry significant information about nativeness of the FRE, SPA and TUR speakers. In order to exploit the advantage of these observations, all the three features (AURORA+DYN_100, MFCC+SDC_100 and Phrase+3Δ) were further fused using (3). The weighted combination of α_1=*0.5*, α_2=*0.4* and α_3=*0.1* gave relatively best WA *40.2 %* and UAR *40.4 %* for the challenge.

3.7. Result on Test set

The results of the test are shown for *2* out of the *5* trials that can be submitted for the challenge. The first trial include scores submitted for the feature MFCC+SDC_100 and for the second trial, scores from the fusion of AURORA, MFCC and phrase-based features (that performed best on development set) were submitted. The Confusion Matrix (CM) for first and second trial is shown in Table 4 (a)-4(b), respectively, sorted in geographical order from west to east. We obtain WA as *31.0 %* and *34.1 %*, UAR as *31.4 %* and *34.3 %* for first and second trial, respectively. The last row of CM indicates the accuracy on individual L1. The grey portions indicate higher confusion between languages. Table 4(a) shows that for MFCC+SDC_100, majority of languages were confused with ARA and CHI irrespective of geographical locations. However, the confusion for ARA is reduced in second trial; this is reflected with higher accuracy for all languages in second trial (except ARA and ITA). This may be due to the complementary information added due to AURORA+ DYN_100 and phrase+3Δ.

Table 4 (a): *The CM of test set for the first trial submission.*

Hypothesis \ Reference	GER	FRE	ITA	SPA	ARA	TUR	HIN	TEL	JPN	KOR	CHI
GER	**21**	2	4	4	2	4	4	1	1	3	3
FRE	4	**12**	3	2	1	2	1	1	0	0	0
ITA	11	9	**23**	4	6	2	1	3	6	3	3
SPA	6	9	6	**23**	4	5	4	8	2	5	5
ARA	8	17	4	11	**32**	24	10	15	8	8	10
TUR	2	2	0	1	0	**9**	0	0	1	3	1
HIN	0	2	1	4	2	1	**15**	11	1	1	4
TEL	3	5	5	5	7	13	28	**39**	1	2	2
JPN	0	2	2	4	8	3	2	2	**36**	14	6
KOR	1	5	3	4	4	5	5	1	5	**23**	4
CHI	19	13	17	15	14	22	12	7	14	18	**36**
Acc(%)	28	15.4	33.8	29.8	40	10	18.3	31.8	48	28.8	48.6

Table 4 (b): *The CM of test set for the second trial submission.*

Hypothesis \ Reference	GER	FRE	ITA	SPA	ARA	TUR	HIN	TEL	JPN	KOR	CHI
GER	**26**	8	3	5	2	5	4	1	1	2	4
FRE	6	**20**	6	5	1	4	3	0	1	2	1
ITA	8	8	**20**	6	6	4	1	1	4	4	3
SPA	6	9	10	**23**	9	12	4	6	5	5	6
ARA	6	10	6	10	**29**	18	12	11	8	7	8
TUR	3	1	0	2	2	**14**	0	1	0	4	1
HIN	0	0	0	2	4	1	**17**	11	0	1	4
TEL	1	5	3	2	4	9	22	**43**	0	0	3
JPN	0	1	2	3	10	3	2	2	**42**	12	4
KOR	1	5	2	5	3	4	6	6	4	**26**	4
CHI	18	11	16	14	11	16	11	6	10	17	**36**
Acc(%)	34.6	25.6	29.4	29.9	36.3	15.5	20.7	48.9	56	32.5	48.6

It was observed on the development set that, fusion of AURORA, MFCC and phrase-based features performed better for GER and FRE than MFCC only. This is reflected in test set as the confusion of GER and FRE to other languages reduced. Moreover, for both first and second trial majority of the languages are confused with CHI, which was not the case for development set. Thus, the fusion factor obtained over the development set did not generalize for the test set. In future, the effect on accuracy by taking the increased weightage of prosodic features for fusion will be explored.

4. Summary and Conclusions

Native speakers have linguistic knowledge of L1 that help in identifying non-native cues of L2 speakers. With respect to this idea, we attempt to learn features from native English speakers' database, i.e., WSJ0 and AURORA 4. Moreover, we also explored segmental information in the form dynamic and shifted delta features. The prosodic information from phrase and its dynamic features was found useful for NLID. On the test, among all languages CHI and JPN were found to perform better than the rest and the performance of TUR was low for all feature set. Our future research work will be directed towards finding prosodic and segmental cues motivated from the listener's perspective.

5. References

[1] J. Lopes, I. Trancoso, and A. Abad, "A nativeness classifier for TED talks," in Proc. Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP), Prague, Czech Republic, pp. 5672–5675, 2011.

[2] C. Bergmann, S.A. Sprenger, and M.S. Schmid,"The impact of language co-activation on L1 and L2 speech fluency", Acta Psychologica,vol. 161, pp. 25-35, 2015.

[3] H. Fujisaki and M. Sugito, "Word accent and sentence intonation in foreign language learning," In Preprints of Papers for the Working Group on Intonation, Edited by Hiroya Fujisaki and Eva Garding, The *13*th Int'l Congress of Linguists, Tokyo, pp.109-119, 1982.

[4] L. Rasier and P. Hiligsmann, "Prosodic transfer from L1 to L2. Theoretical issues," Nouveaux Cahiers De Linguistique Française, vol. 28, pp.41-66, 2007.

[5] Min Ma, Keelan Evanini, Anastassia Loukina, Xinhao Wang, "Using F_0 Contours to Assess Nativeness in a Sentence Repeat Task", in Proc. INTERSPEECH, Dresden,Germany, pp. 653-657, 2015.

[6] S. Sam, X. Xiao, L. Besacier, E. Castelli, H. Li, and E. S. Chng, "Speech modulation features for robust non-native speech accent detection," in Proc. INTERSPEECH, Florence, Italy, pp. 2417-2420, 2011.

[7] P. Macizo, "Phonological coactivation in the bilinguals' two languages: evidence from the color naming task", Biling. Lang. Cogn., vol. 19, no. 2, pp. 361-375, 2016.

[8] J. van Doremalen, C. Cucchiarini, H. Strik, "Optimizing automatic speech recognition for low-proficient non-native speakers", EURASIP Journal on Audio, Speech, and Music Processing, vol. 2010, no.1, pp. 1-13, 2009.

[9] J.E. Flege, E.M. Frieda, T. Nozawa, "Amount of native-language (L1) use affects the pronunciation of an L2", Journal of Phonetics, vol. 25, no.2, pp. 169–186, 1997.

[10] B. Schuller, S. Steidl, A. Batliner, J. Hirschberg, J. K. Burgoon, A. Baird, A. Elkins, Y. Zhang, E. Coutinho, K. Evanini,"The INTERSPEECH 2016 Computational Paralinguistics Challenge: Deception, Sincerity & Native Language", in Proc. INTERSPEECH, ISCA, San Francisco, USA, 2016.

[11] Teixeira et al., "Evaluation of speaker's degree of nativeness using text-independent prosodic features," in Proc. of the Workshop on Multilingual Speech and Language Processing, Aalborg, Denmark, 2001.

[12] D. B. Paul and J. M. Baker, "The design for the Wall Street Journal-based CSR corpus," in Proc. of the Workshop on Speech and Natural Language, Stroudsburg, PA, USA, HLT '91, pp. 357–362, Association for Computational Linguistics (ACL), 1992.

[13] N. Parihar and J. Picone, "Aurora working group: DSR front end LVCSR evaluation AU/384/02," Tech. Rep., Inst. for Signal and Information Process, Mississippi State University, 2002.

[14] D.R. González and J. R. Calvo de Lara. "Speaker verification with shifted delta cepstral features: Its Pseudo-Prosodic Behaviour," in Proc. I Iberian SLTech 2009.

[15] S. Furui, "Cepstral analysis for automatic speaker verification", IEEE Trans on Acoustics, Speech, and Signal Processing, vol 29, no. 2, pp. 254-272, 1981.

[16] F. Soong and A. Rosenberg. "On the use of instantaneous and transitional spectral information in speaker recognition." IEEE Trans on Audio Speech and Signal Proc. vol 36, no. 6, pp. 871-879, 1988.

[17] H. B. Sailor and H. A. Patil, "Filterbank learning using convolutional restricted boltzmann machine for speech recognition," in Proc. Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP), Shanghai, China, pp. 5895-5899, 2016.

[18] M. S. Lewicki, "Efficient coding of natural sounds," Nature Neuroscience, vol. 5, no. 4, pp. 356–363, 2002.

[19] P.A. Torres-Carrasquillo, E. Singer, M.A. Kohler, R.J. Greene, D.A. Reynolds, and J.R. Deller, Jr., "Approaches to language identification using Gaussian mixture models and shifted delta cepstral features," in Proc. Int'l Conference on Spoken Language Processing (ICSLP), Colorado, USA, pp. 89-92, 2002.

[20] M. V. Segbroeck, R. Travadi and S. S. Narayanan. " Rapid language identification," IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 23, no. 7, pp. 1118-1129, 2015.

[21] H. Fujisaki, R. Tomana, S. Narusawa, S. Ohno, C. Wang, "Physiological mechanisms for fundamental frequency control in Standard Chinese," in Proc. Int'l Conference on Spoken Language Processing (ICSLP), Beijing, China, pp. 9–12, 2000.

[22] H. Fujisaki and S. Nagashima, "A model for synthesis of pitch contours of connected speech," *Annual Report, Engg. Res. Inst., University of Tokyo*, vol. 28, pp. 53–60, 1969.

[23] K. S. R. Murty and B. Yegnanarayana, "Epoch extraction from speech signals," IEEE Trans. on Speech and Audio Process., vol. 16, no. 8, pp. 1602-1613, 2008

[24] H. Mixdorff, "A novel approach to the fully automatic extraction of Fujisaki model parameters," in Proc. Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP), Istanbul, Turkey, vol. 1, pp. 1281-1284, 2000.

[25] A. Rosenberg,"Classifying Skewed Data: Importance Weighting to Optimize Average Recall." in Proc. INTERSPEECH, Oregon, USA, pp. 2242-2245, 2012.

[26] A. Jain, K. Nandakumar, and A. Ross, "Score normalization in multimodal biometric systems," Pattern Recognition, vol. 38, no. 12, pp.2270–2285, 2005.

[27] E. Smith and M. S. Lewicki, "Efficient coding of time-relative structure using spikes," Neural Comput., vol. 17, no. 1, pp.19–45, 2005.

[28] Kashiwagi et.al,"Divergence estimation based on deep neural networks and its use for language identification," in Proc. Int'l Conference on Acoustics, Speech, and Signal Processing (ICASSP), Shanghai, China, pp. 5435-5439, 2016.

INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

Multimodal Fusion of Multirate Acoustic, Prosodic, and Lexical Speaker Characteristics for Native Language Identification

Prashanth Gurunath Shivakumar, Sandeep Nallan Chakravarthula, Panayiotis Georgiou

University of Southern California, Los Angeles, CA, USA
{pgurunat,nallanch}@usc.edu, georgiou@sipi.usc.edu

Abstract

Native language identification from acoustic signals of L2 speakers can be useful in a range of applications such as informing automatic speech recognition (ASR), speaker recognition, and speech biometrics. In this paper we follow a multistream and multi-rate approach, for native language identification, in feature extraction, classification, and fusion. On the feature front we employ acoustic features such as MFCC and PLP features, at different time scales and different transformations; we evaluate speaker normalization as a feature and as a transform; investigate phonemic confusability and its interplay with paralinguistic cues at both the frame and phone-level temporal scales; and automatically extract lexical features; in addition to baseline features. On the classification side we employ SVM, i-Vector, DNN and bottleneck features, and maximum-likelihood models. Finally we employ fusion for system combination and analyze the complementarity of the individual systems. Our proposed system significantly outperforms the baseline system on both development and test sets.

Index Terms: language nativity detection, i-vectors, VTLN, Phoneme-level prosodic features, phonemic log-likelihood features, Deep neural network, bottleneck features, L1, fMLLR

1. Introduction

Speech signals, in addition to the explicitly expressed lexical content, contain a diverse range of information about speakers such as age, emotions, speaker identity, environment characteristics, language background of the speaker *etc.*. Capturing and describing such diverse information enables adaptation and improved performance of speech processing systems. One of these important characteristics to capture is the native language of the speaker.

Identification of the native language (L1) of a non-native English speaker from English (L2) speech is a challenging research problem. Knowledge of the native language can aid Automatic Speech Recognition systems through specifically tuned models, can provide culturally aware machine-human interfaces and can provide cues towards more accurate speaker recognition, speech biometrics and speech forensics by effectively modeling the phonotactic variability of speakers across various languages.

There has been relatively less research in the area of native language detection. Most of the research involves study with 2 to 4 way classification. In [1], a support vector machine (SVM) was used to classify 8 native languages using ASR based features under a universal background model (UBM) framework. Shriberg *et al.* [2] used multiple approaches based on lexical systems by using phone and word N-gram language models (LM) to show that the word based N-gram LM was more effective than a phone based one. Several studies have shown prosodic information like energy, duration, pitch, and formant based functionals to be effective features [2–4]. The native language identification task was found to be particularly difficult for spontaneous speech [3]. On the acoustic front, Gaussian Mixture Models (GMM) have been used to train a model specific to different accents [5]. For training such GMMs front-end acoustic features in the form of Cepstral based features, like Perceptual Linear Prediction (PLP) [5] and Mel Frequency Cepstral Coefficients (MFCC) [3], and second and third formant features [4], have been employed. Different training techniques like Maximum Mutual Information (MMI) [5] and Minimum Phone Error (MPE) [1] were found to be useful. Stochastic trajectory models (STM) based on phonemes were successfully applied to capture the dynamics of accents specific to each phones [3]. An in-depth analysis of temporal characteristics of accents were performed in [6], showing significant differences between foreign accented English, hinting at the potential of the duration based features towards accent classification.

In this paper, we use acoustic features, MFCC and PLP of different time scales, in an i-Vector framework with probabilistic linear discriminant analysis (PLDA) to model the acoustic information. Deep neural networks (DNN) are used to derive bottleneck features, which in turn are used to train the i-vectors to boost the discriminative power of the frame level acoustic features. We introduce a *Pronounciation-Projection* (L1-ProP) feature by projecting acoustics in the English-language pronounciation space via an ASR, that can capture L1-specific phonemic mismatch. We also propose novel phoneme-level features in terms of *Phonemic Confusion* (PC) and *Phoneme Specific Prosodic Features* (PSPS) which are designed to capture the confusability and the short term prosody-dynamics on phone level. On the lexical front, the grammatical variations on word level persistent in specific languages are exploited. Finally, the introduced features are fused together along with the baseline features for classification. The experimental results are presented on the ETS corpus of non-native spoken English comprising of 11 distinct L1 speakers, as a part of Interspeech Native Language Sub-Challenge [7].

The rest of the paper is organized as follows. First, the database and baseline system are briefly described in Section 2. We then describe the features employed in Section 3 and the classification algorithms in Section 4. We provide a brief description of our fusion method in Section 5 before we proceed to analysis of our results in Section 6. We conclude and provide future directions in Section 7.

2. Database and Baseline System

2.1. Database

The Educational Testing Service (ETS) corpora used in this work is built on the spontaneous speech of non-native English

http://dx.doi.org/10.21437/Interspeech.2016-1312

speakers taking the TOEFL IBT exam. The corpora consists of 5,132 speakers from 11 L1 backgrounds with approximately 64 hours of speech (45s per speaker). The 11 L1 categories were Arabic, Chinese, French, German, Hindi, Italian, Japanese, Korean, Spanish, Telugu and Turkish. Additional details of division of data, speakers and L1 classes, for training, development and testing corpora is available in [7].

2.2. Baseline System

The baseline for our system is trained using utterance level statistics of the acoustics such as spectral (e.g., formants), cepstral (e.g., MFCC, RASTA), tonal (e.g., CHROMA, CENS) and voice quality (e.g., jitter, shimmer), for a total of 6373 dimensions, extracted using OpenSMILE [8]. The features are used to train a support vector machine (SVM) to classify among the 11 L1 categories and details can be found in [7].

3. Features

In our proposed system, we use multirate information from acoustic, prosodic, phoneme-confusability, phoneme-level prosodic, and lexical streams to train complementary multiple expert systems. The features were tailored to capture (i) discriminative information between the 11 non-native L1 language (ii) discriminative variability with respect to native English speaking patterns.

3.1. Frame-level Features

On the acoustic front-end, we use MFCC, PLP and log power-spectral features due to their success in prior work [3, 5]. We use multiple streams of acoustic features to capture variability in terms of multiple temporal (25ms to 150ms frame size) and spectral resolutions (23-69 mel-filterbanks with 13 to 39 MFCC). The delta and delta-delta features were computed and mean normalized.

VTLN: To reduce inter-speaker variability we can employ speaker normalization techniques such as Vocal Tract Length Normalization (VTLN) [9], Maximum Likelihood Linear Regression (MLLR) [10], and Speaker Adaptive Training techniques (SAT) [11]. In our work we employ linear-VTLN via an affine transformation to approximate the non-linear warping of the frequency axis similarly to the method in [12]. It is unclear however if such normalization also removes L1 specific features, something we intend to investigate.

3.2. Bottleneck features

Bottleneck features were shown to be useful for speaker recognition [13] and language identification [14] task. We generate bottleneck features via a DNN with a 23 frame context input of 257-dimensional log-spectra that mirror the human auditory system [15]. The DNN thus has a 5911 dimensional input and 3 hidden layers with 2000, 50 and 500 neurons.The 50-dimensional bottleneck features along with their delta and delta-delta features are mean normalized and used to train the total variability matrix of the i-vector framework.

3.3. Phoneme-level Features

Past studies have demonstrated the influence of L1 backgrounds on L2 speakers' pronunciation of English vowels and consonants [16–19]. Different backgrounds are associated with specific perceptual errors in recognition between different phonemes. For instance, strong confusion has been observed between Japanese speakers' pronunciation of $/l/$ and $/r/$ phonemes [20] and between $/n/$ and $/l/$ for Chinese speakers [21]. Wiltshire et al observed Gujarati and Tamil influences on pitch accents and slopes, similar to those that Arslan et al observed with Mandarin and German [6, 22]. Phoneme durations have been shown to be a prominent feature characterizing accents and dialects [6] as well.

Such traits are likely complementary to the frame-level acoustic features. Capturing such traits involves a projection of the speaker characteristics on the English-language space and the analysis of this projection. This can be practically implemented as the projection to the likelihood space of each phoneme via a speech recognizer. We can also employ this projection in several ways:

3.3.1. L1-Pronounciation Projection (L1-ProP)

The L1-ProP features are designed to capture the pronunciation variability between the L1 English speakers and the L2 speakers. Since different languages employ a different phonetic inventory, we hypothesize that this will create specific responses in the phonemic projection of L2 English speech on the native English speech space. To obtain a compact projection we used a mono-phone phoneme recognizer trained on native English speakers [23] using the Kaldi toolkit [24]. The frame level log-likelihood score is obtained from the ASR monophone model using the following criterion:

$$LL_p = \max_{s \in S_p} log(P(f|s)) \qquad \forall p \in P \tag{1}$$

where p is a phone from set of phones, P, s is the state from the set of states, S_p, specific to phoneme p, f is the frame. For each frame, we get a 41 dimensional vector corresponding to log-likelihoods for 39 non-silence and 2 silence phones. In short we select the best match per phoneme for all the various states belonging to that phoneme. We further explored projection on a range of different languages.

3.3.2. Phonemic Confusion

To obtain the phoneme confusion features, we used the phoneme likelihoods described in Sec. 3.3.1. We want to investigate phoneme confusion so we generated a pairwise-confusion matrix from the cross-product of the 39 dimensional confusion log likelihoods. We then vectorize the lower-triangular elements and obtain the average confusion vector per phoneme from its instances as determined by the ASR. Finally, we average this vector over all phonemes to obtain a 780-dimensional feature per file.

3.3.3. Phoneme Specific Prosodic Features (PSPS)

Prosodic variability has been shown to be useful in native language identification. The baseline features employ prosody with success. We also hypothesize that phone-specific prosodic variability can provide useful information. Based on phoneme alignments obtained by the ASR above we compute the mean, standard deviation, median, min, max and range of phoneme duration, short-time energy and pitch (only for voiced). We then average over each phoneme type (*i.e.,* over all "AA" phonemes, over all "B" phonemes *etc.*). This results in a 1062-length feature vector over all phonemes (30 features $\times$ 30 voiced phonemes, 18 $\times$ 9 unvoiced). In case a phoneme is not observed in a session, we impute its features using the global averages from other train sessions where it was observed.

3.4. Lexical features

We believe that lexical channel can capture 2 types of information: 1. the style of expression and language use errors will vary according to the native language of the speaker; and 2. an ASR transcript will contain consistent errors based on consistent mispronounciations resulting from L1 specific phonemic confusability. Given the limited lexical data and the error associated with recognizing L2 speech we decide to employ the 1000 n-best list of each utterance of each file as our lexical representation of each speaker. Decoding was done using a DNN-ASR system trained on the Fisher corpus.

3.5. fMLLR Transform based Features

Feature-space Constrained Maximum Likelihood Linear Regression (fMLLR) is a linear transformation used for speaker and environment adaptation in modern ASRs such that it maximizes the observation data likelihood given the model [11]. While it removes a lot of this variability it may also remove native language specific information, we thus decide to investigate whether the 39×40 dimensional fMLLR transform conveys native language information and employ it as a feature.

4. Classification Techniques

4.1. i-Vector

Recently, i-vector modeling was introduced in application to the task of speaker verification [25]. Its excellent state-of-the-art performance gained significant research interest among the signal processing community. The total variability modeling of i-vectors have since been applied to various tasks like language recognition [26], speaker recognition [27], speaker age recognition [28,29]. For our work we use total variability i-vector modeling. We train a full covariance GMM-UBM on the ETS Corpus training dataset. The UBM was trained using 2048 gaussian mixtures. The zeroth and the first order baum welch statistics are computed from the training data and the total variability matrix is estimated using Expectation-Maximization. Finally, we extract mean and length normalized i-vectors.

4.2. PLDA

For scoring, we use probabilistic linear discriminant analysis (PLDA), due to its state-of-the-art results in speaker recognition domain [27]. Given a pair of i-vectors, PLDA evaluates the ratio of probability that the two i-vectors belong to the same native background to the probability that the two i-vectors are from different native backgrounds [30]. The log-likelihood scores obtained after PLDA scoring are used for classification.

4.3. SVM based phoneme-level feature classification

We implemented the phonemic confusability and prosodic features as described in Secs. 3.3.2 and 3.3.3. The session-level features were trained and tested using the same parameters as the baseline system using PolyKernel SVM and Weka [31].

4.4. Maximum Likelihood Lexical Classification

Given the limited lexical data we decided to use a simple *Maximum Likelihood* (ML) classification framework. We considered alternatives, such as a word2vec front end, however the embedings may preserve the lexical similarity but not necessarily the actual word biases of L2 speakers that we desire to capture. Models were smoothed with background data to ensure robustness and to boost the importance of domain-salient words. For transcript we used the 1000 best of each utterance in the test file similarly to [32].

5. Fusion

Both feature and score level fusion techniques were explored in this work. Feature level fusion was used to emphasize the complementarity of the presented features to the baseline. Whereas, the score level fusion was employed for multiple combinations of all the presented modalities to improve performance.

Feature-level fusion: Features from different standalone systems were evaluated by concatenating them to the baseline features and training a SVM directly. Fusion on i-vector level was also tried by applying linear discriminant analysis (LDA) on individual systems first and then on the fused i-vector features. The fused i-vectors are used to train the PLDA system for obtaining the log-likelihood scores.

Score-level fusion: For score-level fusion, logistic regression is performed over the log-likelihood scores obtained from multiple systems using the Bosaris toolkit [33]. For i-vector based systems, the log-likelihood scores are directly obtained from the PLDA scoring. Whereas, for the SVM/DNN and lexical classifiers, the posteriors and perplexity are converted to log-likelihoods respectively. For training the fusion systems, we perform k-fold cross-validation on training data to obtain new set of perturbed log-likelihoods, which is more representative of the errors the i-vector framework makes on testing data.

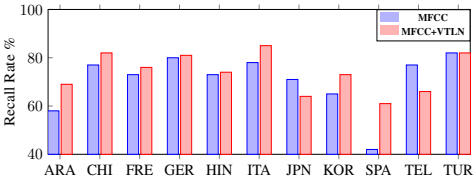

Figure 1: Effect of VTLN on recall rates of languages

6. Experimental results & Discussions

We present the results for individual systems first, and then finally we evaluate the fusion performance of multiple systems.

6.1. Standalone system performance

Table 1 gives the summary of performance for different standalone systems.

Acoustic i-vector modeling: We observe typical PLP and MFCC based acoustic features to be reliable giving the best individual-system results. We find that PLP outperforms the MFCC features by approximately 2% absolute both in terms of Accuracy and UAR.

Effect of VTLN: Figure 1 demonstrates the effect of VTLN on MFCC features. The recall of 11 different languages are plotted for raw MFCCs and VTLN-MFCCs. We see that the VTLN gives consistent improvement to most of the languages except Japanese and Telugu. We obtain a significant increase of absolute 19% recall for Spanish. Overall, we find VTLN to be useful providing 3.6% absolute increase in accuracy and recall rates.

L1-ProP and i-vector: We find that using VTLN-MFCCs to extract the log-likelihood features does not significantly improve the performance. Further, gaussianization of features and PCA dimension reduction (23 dimensions) were found to be useful providing a boost of 9% absolute. Overall, the phoneme confusability log-likelihood features prove to be less reliable compared to the acoustically trained i-vectors. L1-ProP features on other foreign languages like Spanish, Hindi, Telugu, Arabic, French and German were also experimented with and were seen to give similar performance to the Spanish. We retain the system for fusion to extract complementary information.

Bottleneck features: We observe that the bottleneck features never approach the performance of other acoustic features (MFCC or PLP). Since they are based on the same modality as MFCC and PLP they also do not provide complementary information thus we do not pursue these further.

Phoneme level features: While both the prosodic and confusability features fail to beat the baseline performance, the prosodic features are observed to provide complementarity to the baseline. Since they also perform similar to the baseline despite using elementary statistics, this supports the need for better phoneme-level modeling.

Lexical features: Lexical features provide performance similar to the baseline and given the different modality we expect them to provide complementary information.

fMLLR features: We see from the result that the raw fMLLR transforms inherit certain L1 characteristics and could be used

	Results on Development		
	Features ➞ Classifier	Accuracy	UAR
45s	Baseline	45.00%	45.10%
25ms	MFCC ➞ iVector ➞ PLDA	70.90%	70.90%
25ms	MFCC-VTLN ➞ iVector ➞ PLDA	74.20%	74.20%
25ms	PLP ➞ iVector ➞ PLDA	72.30%	72.50%
25ms	PLP-VTLN ➞ iVector ➞ PLDA	**76.40%**	**76.40%**
25ms	Bottleneck features on log power spectrogram ➞ iVector ➞ PLDA	36.40%	36.70%
45s	fMLLR ➞ SVM	42.30%	42.70%
Word	Lexical ➞ Maximum Likelihood w/ smoothing	44.60%	41.00%
25ms	L1-ProP ➞ iVector ➞ PLDA (English ASR)	60.50%	60.70%
25ms	L1-ProP ➞ Gaussianiazation ➞ PCA iVector ➞ PLDA (English ASR)	69.60%	69.80%
25ms	L1-ProP ➞ Gaussianiazation ➞ PCA iVector ➞ PLDA (Spanish ASR)	66.00%	66.30%
25ms	L1-ProP + VTLN ➞ iVector ➞ PLDA	60.90%	61.30%
~80ms	Phone Confusability Distribution ➞ SVM	25.50%	25.80%
~80ms	Phoneme Specific Prosodic Signature (PSPS) ➞ SVM	40.70%	41.10%
	Feature level fusion	Accuracy	UAR
25ms	Bottleneck + MFCC-VTLN ➞ iVector ➞ PLDA	46.40%	46.80%
	Baseline + Phone Confusability Distribution ➞ SVM (English ASR)	44.40%	44.50%
	Baseline + Phoneme Specific Prosodic Signature ➞ SVM	51.50%	51.70%
	Score level Fusion via Logistic Regression	Accuracy	UAR
	Baseline + (Bottleneck & MFCC-VTLN)	48.20%	48.60%
	Baseline + fMLLR	48.10%	48.30%
	Baseline + Lexical	52.10%	52.10%
	Baseline + Lexical + L1-ProP (English ASR)	66.50%	66.60%
	Baseline + Lexical + MFCC-VTLN	76.90%	77.00%
	Baseline + Lexical + PLP-VTLN	77.80%	77.90%
	Baseline + Lexical + PLP-VTLN + MFCC-VTLN + L1-ProP-VTLN (English ASR)	**78.50%**	**78.60%**
	+ PSPS	64.30%	65.40%
	+ Phone Confusion	74.70%	74.90%
	+ PSPS + Phone Confusion	74.90%	75.10%
	+ fMLLR	78.10%	78.20%
	Leave One Out (From best system) via Logistic Regression		
	Baseline + Lexical + PLP-VTLN + MFCC-VTLN + L1-ProP-VTLN	**78.50%**	**78.60%**
	- MFCC-VTLN	76.80%	76.90%
	- PLP-VTLN	75.60%	75.70%
	- Baseline	75.10%	75.30%
	- Lexical	76.70%	76.80%
	Results on Test		
	MFCC-VTLN + PLP-VTLN + Baseline + Lexical (Submission 3)	79.93%	80.13%

Table 1: Results of the various systems as described in text.

as a potential feature for L1 identification. It was also found to provide some complementarity to the baseline features.

6.2. Fusion Performance

Feature-level fusion: We attempted feature-level fusion for our lowest-performing features to increase performance. We can see from Table 1 that all three improve marginally above baseline, but not significantly so.

Score-level fusion: Analyzing the performance of multiple score level fusion combinations for i-vectors, on the acoustic front, we find that PLP and MFCC exhibit acoustic complementarity. Fusion of acoustic features with the baseline and lexical systems provide further improvements. Even-though the L1-ProP i-vector system doesn't provide noticeable increase in performance when fused with acoustic features, we see improvements when used along with the lexical and baseline features. However, we observe that the Phonemic Confusability (PC) and Phoneme Specific Prosodic Signature (PSPS) do not improve the overall performance of the system. We believe that the noise in the feature extraction may be responsible for the low performance and we intend to investigate further. We also believe that these features can provide improvements for discriminability of specific language pairs. The fMLLR features did not affect the performance of our best system significantly. We believe that the information captured by fMLLR features is redundant with the combination of other features.

Our best performing system is a combination of acoustic (MFCC-PLP), lexical, prosodic (Baseline), and L1-ProP. The best performing system achieves an Accuracy of 78.5% and UAR of 78.60% on the development test.

We perform leave-one-out from the best system to analyze the importance of each feature. We find PLP and Baseline features to be significant contributors in terms of complementary information giving approximately 3% improvements, whereas, MFCC and Lexical features contribute around 2%. Finally L1-ProP features improves the overall system by a small margin.

	ARA	CHI	FRE	GER	HIN	ITA	JPN	KOR	SPA	TEL	TUR
ARA	54	1	5	0	1	4	8	2	2	1	2
CHI	0	62	0	2	0	1	2	6	0	1	0
FRE	4	3	56	3	2	3	1	0	5	1	0
GER	0	0	4	69	0	0	0	0	2	0	0
HIN	0	0	0	0	64	1	0	0	1	16	0
ITA	1	1	2	0	0	59	1	0	2	0	2
JPN	1	1	0	0	0	0	71	2	0	0	0
KOR	2	5	0	1	0	0	12	60	0	0	0
SPA	1	0	4	2	2	8	2	2	54	1	1
TEL	0	0	0	0	19	0	0	0	0	69	0
TUR	7	1	2	1	0	1	1	0	2	0	75

Table 2: Confusion matrix of the best results on test, corresponding to an Accuracy = 79.93% and UAR = 80.13%

Across the modalities, we observe different features providing discriminability between specific language pairs. In future, we intend to employ a hierarchical classification method to exploit such properties.

6.3. Inter-class confusion analysis

Figure 2 shows the confusion matrix obtained for our best performing system on the development set. Italian and Turkey are the least confused languages and French is the most confused. The matrix shows inter-language confusions between Hindi - Telugu and Japanese - Korean languages correlating with demographics between the languages. In our human-analysis, that included three Indian speakers, we couldn't separate most of the confusable development set Hindi and Telugu pairs. Overall, comparing with the baseline system, we find the confusion to be significantly more sparse suggesting not only better performance but also less confusion among language pairs with our improved system.

6.4. Results on the test

For testing, we used the score level fusion MFCC-VTLN, PLP-VTLN i-vector system, Baseline and Lexical features to achieve a performance of **79.93%** Accuracy and **80.13%** UAR. We believe that inclusion of other systems and further calibration during fusion on per-language level basis rather than global 11 class classification metrics could boost the performance. Due to time constraints, we were unable to try further combinations and didn't incorporate L1-ProP features with Gausianization.

7. Conclusion

In this work, we have addressed a challenging research problem of detecting the L1 native language from spontaneous speech on 11 different L1 language categories. We exploit different modalities, multiple feature rates, and a range of methods towards robust classification. Each modality was shown to improve the performance of the baseline system when fused with the baseline features, demonstrating the complementarity of the proposed features. We also showed the effectiveness of speaker normalization. We successfully demonstrate that some L1 information exists in the normalization (fMLLR) feature and could be used as a potential feature for L1 detection. While the phoneme confusability and phoneme-level prosodic features did not improve the overall system performance, they were shown to be effective in improving the baseline. Different fusion techniques were applied to extract complementary information across various modalities.

By analyzing the confusion in the system, we observed inherent correlations with the demographics among certain languages. From an unscientific sampling of human listeners our system seems to face similar challenges to humans especially for the highly confusable language pairs. In short, we present an accurate multimodal, multirate L1 identification system via a range of feature, classification, and fusion methods.

8. References

[1] M. K. Omar and J. Pelecanos, "A novel approach to detecting non-native speakers and their native language," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*. IEEE, 2010, pp. 4398–4401.

[2] E. Shriberg, L. Ferrer, S. S. Kajarekar, N. Scheffer, A. Stolcke, and M. Akbacak, "Detecting nonnative speech using speaker recognition approaches." in *Odyssey*. Citeseer, 2008, p. 26.

[3] S. Gray and J. H. Hansen, "An integrated approach to the detection and classification of accents/dialects for a spoken document retrieval system," in *Automatic Speech Recognition and Understanding, 2005 IEEE Workshop on*. IEEE, 2005, pp. 35–40.

[4] S. Deshpande, S. Chikkerur, and V. Govindaraju, "Accent classification in speech," in *Automatic Identification Advanced Technologies, 2005. Fourth IEEE Workshop on*. IEEE, 2005, pp. 139–143.

[5] G. Choueiter, G. Zweig, and P. Nguyen, "An empirical study of automatic accent classification," in *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*. IEEE, 2008, pp. 4265–4268.

[6] L. M. Arslan and J. H. Hansen, "A study of temporal features and frequency characteristics in american english foreign accent," *The Journal of the Acoustical Society of America*, vol. 102, no. 1, pp. 28–40, 1997.

[7] B. Schuller, S. Steidl, A. Batliner, J. Hirschberg, J. K. Burgoon, A. Baird, A. Elkins, Y. Zhang, E. Coutinho, and K. Evanini, *The INTERSPEECH 2016 Computational Paralinguistics Challenge: Deception, Sincerity & Native Language*. Proceedings INTERSPEECH 2016, ISCA, San Francisco, USA, 2016.

[8] F. Eyben, M. Wöllmer, and B. Schuller, "Opensmile: the munich versatile and fast open-source audio feature extractor," in *Proceedings of the 18th ACM international conference on Multimedia*. ACM, 2010, pp. 1459–1462.

[9] E. Eide and H. Gish, "A parametric approach to vocal tract length normalization," in *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, vol. 1. IEEE, 1996, pp. 346–348.

[10] C. J. Leggetter and P. C. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models," *Computer Speech & Language*, vol. 9, no. 2, pp. 171–185, 1995.

[11] M. J. Gales, "Maximum likelihood linear transformations for hmm-based speech recognition," *Computer speech & language*, vol. 12, no. 2, pp. 75–98, 1998.

[12] D. Kim, S. Umesh, M. Gales, T. Hain, and P. Woodland, "Using vtln for broadcast news transcription," in *Proc. ICSLP*, vol. 4, 2004.

[13] T. Yamada, L. Wang, and A. Kai, "Improvement of distant-talking speaker identification using bottleneck features of dnn." in *INTERSPEECH*, 2013, pp. 3661–3664.

[14] K. Vesely, M. Karafiát, F. Grezl, M. Janda, and E. Egorova, "The language-independent bottleneck features," in *Spoken Language Technology Workshop (SLT), 2012 IEEE*. IEEE, 2012, pp. 336–341.

[15] F. Xie and D. C. Van, "A family of mlp based nonlinear spectral estimators for noise reduction," in *Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference on*, vol. 2. IEEE, 1994, pp. II–53.

[16] T. Piske, I. R. MacKay, and J. E. Flege, "Factors affecting degree of foreign accent in an l2: A review," *Journal of phonetics*, vol. 29, no. 2, pp. 191–215, 2001.

[17] J. E. Flege, O.-S. Bohn, and S. Jang, "Effects of experience on non-native speakers' production and perception of english vowels," *Journal of phonetics*, vol. 25, no. 4, pp. 437–470, 1997.

[18] R. K. Bansal, "The pronunciation of english in india," *Studies in the pronunciation of English: A commemorative volume in honour of AC Gimson*, pp. 219–230, 1990.

[19] J. E. Flege, "Assessing constraints on second-language segmental production and perception," *Phonetics and phonology in language comprehension and production: Differences and similarities*, vol. 6, pp. 319–355, 2003.

[20] A. Sheldon and W. Strange, "The acquisition of/r/and/l/by japanese learners of english: Evidence that speech production can precede speech perception," *Applied Psycholinguistics*, vol. 3, no. 03, pp. 243–261, 1982.

[21] H. Meng, Y. Y. Lo, L. Wang, and W. Y. Lau, "Deriving salient learners mispronunciations from cross-language phonological comparisons," in *Automatic Speech Recognition & Understanding, 2007. ASRU. IEEE Workshop on*. IEEE, 2007, pp. 437–442.

[22] C. R. Wiltshire and J. D. Harnsberger, "The influence of gujarati and tamil l1s on indian english: A preliminary study," *World Englishes*, vol. 25, no. 1, pp. 91–104, 2006.

[23] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 5206–5210.

[24] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, Dec. 2011, iEEE Catalog No.: CFP11SRW-USB.

[25] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 19, no. 4, pp. 788–798, 2011.

[26] N. Dehak, P. A. Torres-Carrasquillo, D. A. Reynolds, and R. Dehak, "Language recognition via i-vectors and dimensionality reduction." in *INTERSPEECH*, 2011, pp. 857–860.

[27] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems." in *Interspeech*, 2011, pp. 249–252.

[28] M. H. Bahari, M. McLaren, D. A. van Leeuwen *et al.*, "Speaker age estimation using i-vectors," *Engineering Applications of Artificial Intelligence*, vol. 34, pp. 99–108, 2014.

[29] P. G. Shivakumar, M. Li, V. Dhandhania, and S. S. Narayanan, "Simplified and supervised i-vector modeling for speaker age regression," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 4833–4837.

[30] M. Senoussaoui, P. Kenny, N. Brümmer, E. De Villiers, and P. Dumouchel, "Mixture of plda models in i-vector space for gender-independent speaker recognition." in *INTERSPEECH*, 2011, pp. 25–28.

[31] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[32] P. G. Georgiou, M. P. Black, A. Lammert, B. Baucom, and S. S. Narayanan, ""That's aggravating, very aggravating": Is it possible to classify behaviors in couple interactions using automatically derived lexical features?" in *Proceedings of Affective Computing and Intelligent Interaction (ACII), Lecture Notes in Computer Science*, October 2011.

[33] N. Brümmer and E. de Villiers, "The bosaris toolkit: Theory, algorithms and code for surviving the new dcf," *arXiv preprint arXiv:1304.2865*, 2013.

INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

Within-Speaker Features for Native Language Recognition in the Interspeech 2016 Computational Paralinguistics Challenge

Mark Huckvale

Speech, Hearing and Phonetic Sciences, University College London, U.K.

m.huckvale@ucl.ac.uk

Abstract

The Interspeech 2016 Native Language recognition challenge was to identify the first language of 867 speakers from their spoken English. Effectively this was an L2 accent recognition task where the L1 was one of eleven languages. The lack of transcripts of the spontaneous speech recordings meant that the currently best performing accent recognition approach (ACCDIST) developed by the author could not be applied. Instead, the objectives of this study were to explore whether within-speaker features found to be effective in ACCDIST would also have value within a contemporary GMM-based accent recognition approach. We show that while Gaussian mean supervectors provide the best performance on this task, small gains may be had by fusing the mean supervector system with a system based on within-speaker Gaussian mixture distances.

Index Terms: accent recognition, second language, computational paralinguistics.

1. Introduction

The goal of the Interspeech 2016 Native Language challenge was to identify the first language (L1) of 867 speakers from 11 language groups given recordings of their extemporaneous second language (L2) performance in response to a range of different question topics. The recordings were collected by the Educational Testing Service (ETS) as part of an English proficiency test. The recordings were made in the speakers' home environment with a range of different audio equipment and background noise levels and contain an average of 40s of speech. A larger set of 4265 similar recordings but labelled for L1 were also provided for the training and development of machine learning systems, see Schuller *et al* [1] for more details.

In this paper we outline the different ways in which the L1 of a speaker influences their L2 accent then consider how current machine approaches to accent recognition make best use of those influences. We argue that good L1 identification will occur when systems are sensitive to the phonetic forms the speaker uses to realise known L2 phonological contexts. We propose a novel method for extracting more accent information from Gaussian Mixture Models (GMM) of accented speech using Gaussian mixture distances. We describe the training and testing protocols and compare the performance of the new technique against established methods within the Native Language challenge task.

2. L2 Accent Recognition

2.1 Characteristics of L2 Accents

When a speaker speaks in an L2 that they have learned later in life, their accent is often influenced by properties of their L1. This is because an adult speaker's production and perceptual systems are highly tuned to speaking and listening in their L1, making them less accurate in phonetic performance and less sensitive in phonetic perception when speaking the L2. While there is much variation across speakers, it appears that some aspects of L2 accents are predictable from phonetic and phonological differences between the two languages [2]. In terms of the Native Language challenge, it is the influence of these differences on the L2 which needs to be recognised.

Following Wells's analysis of L1 accent differences [3, p.72], we may describe the L1 influence on L2 accents under these headings:

a) *Phonetic realisation*: these are influences of the L1 on the phonetic form of L2 phonological units, for example the exact quality chosen for the L2 phonemes. These qualities may vary with L1 because some L1 phones may be copied rather than adapted for the L2. Examples are the re-use of L1 vowel qualities or the use by Hindi speakers of the voiced retroflex plosive [ɖ] for English /d/.

b) *Phonemic system*: these are differences in the phoneme inventory between the two languages. These can cause problems for speakers particularly where phonological choices found in the L2 are not exploited in the L1. Examples are the /l/-/r/ difference in English not exploited in Japanese, or the /i/-/ɪ/ difference not exploited in Spanish.

c) *Phonotactics*: these are differences not in the inventory of phonemes but on allowable sequences of phonemes in the L2 compared to the L1. A speaker may struggle with phone sequences found in the L2 that are not used in the L1 – the novelty, for example, of the English consonant cluster /skw-/ for German speakers.

d) *Lexical distribution*: these are differences between L2 accents in terms of which phonological units are found in the lexical pronunciation of some words. These differences can arise because of conflicting letter-to-sound rules in the L1 and L2 – a particular problem given the vagaries of English spelling. A typical error might be a Spanish or Italian speaker pronouncing the past tense morpheme "-ed" as /-ɪd/ rather than /t/ in a word like "watched".

http://dx.doi.org/10.21437/Interspeech.2016-1466

e) *Prosody*: differences in lexical stress, timing, intonation and voice quality also occur across L2 accents. English lexical stress can be unpredictable and false friends from the L1 may give non-native forms. Rhythmical characteristics of the L1 may also influence the L2, for example the differences between so-called stress-timed and syllable-timed languages. The forms of pitch contours that realise intonational functions vary with the L1, for example the preference of Italian speakers to complete pitch movements within the accented syllable, or the influence of L1 lexical tone on L2 intonation in a tone language such as Chinese.

From this analysis it is important to see that the influence of the L1 on the L2 is not limited to the *range* of sounds produced by the speaker but also includes *which* sounds are used to implement known phonological choices. A French speaker may articulate a perfectly English like [z] phone, which only becomes an L2 accent difference if it is used to represent /ð/ in "this". We also see that the L1 can influence sounds in particular contexts, such as specific phone sequences, specific lexical stress patterns, or specific intonational tunes. Thus for an accurate understanding of the L1 influence on the L2 accent we need to consider both the sounds and the phonological contexts in which they occur.

2.2 Approaches to Machine Accent Recognition

Previous approaches to machine accent recognition have included: (i) analysis of linguistic-phonetic forms, (ii) variation in the global acoustic signature of the speech signal, (iii) variation in the probability density function (pdf) of the short-time spectrum, and (iv) variation in the phonetic labelling given by automatic speech recognition (ASR) systems. We will discuss each of these in turn.

Systems based on a linguistic-phonetic analysis use measurements of the phonetic forms of known phonological entities and compare measurements across accents. For L2 accent recognition such an approach makes use of the actual phonetic form of the expected underlying phonological units, for example how does this speaker produce the /uː/ vowel in “boot”? This approach is particularly powerful when all speakers read the same text as even differences in lexical distribution can be detected. The ACCDIST method developed by the author [4] is an example of this approach.

Systems based on global spectral properties assume that the L1 affects the L2 accent in terms of changes to the overall spectral signature. Such systems convert a recording into a fixed length pattern vector comprising features such as summative statistics of spectral energies, pitch variation, voice-quality and amplitude envelope modulations. The challenge baseline system [1] is of this type and exploits the SMILE feature extraction toolkit to generate a feature vector [5].

Systems based on changes to the short-term spectrum probability distribution compute short-term spectral envelope features (at say 100/sec) and model these with Gaussian Mixture Models (GMMs). The spectral pdfs may be computed separately for different accents and exploited to find the accent giving the highest posterior probability of generating a recording. A number of variants of such systems exist that generate feature vectors from the spectral pdf such as Gaussian Mean Supervectors, Gaussian Posterior Probability Supervectors, or i-Vectors, see Bahari et al [6] for example implementations.

Systems based on the phonotactics of ASR outputs exploit the sensitivity of speech recognition systems to accents that differ from the ones used to train them. If recognisers produce phone transcriptions which have different statistical properties across accents, then these may be used to identify the accent. These properties not only include the relative frequency of each phone, but also the relative frequency of common diphones and triphones. Tellingly, the recognition systems do not need to be of the L2 language nor even of one language, indeed it has been shown to be beneficial to use a mix of languages to estimate phone sequence probabilities [7].

A performance comparison of three of these types of accent recognition may be found in [7] and repeated in Table 1. The task was identification of 14 regional accents of the British Isles from read speech of good quality. Overall the GMM systems performed worse than the phone recognition systems, but both were worse than the linguistic-phonetic systems represented by ACCDIST. We interpret this outcome as showing the value of interpreting phonetic form in the context of phonological structure, since of the systems tested it is only the linguistic-phonetic approach that exploits knowledge of *what* was spoken.

Table 1. *Percentage accent recognition accuracy for different systems tested by Hanani et al [7].*

System Description	Accuracy %
GMM-UBM	61.13
GMM-SVM	76.11
Phonotactic	82.14
Fused GMM+Phonotactic	89.96
ACCDIST-Correlation	93.17
ACCDIST-SVM	95.18

2.3 System Choice for the Native Language Challenge

The ETS corpus of non-native spoken English used in the challenge places constraints on the type of system most suited to the challenge. Firstly, since the speech was spontaneous, the recordings contain different linguistic content which adds noise to the spectral signature used in the global spectral approach. Secondly the speech is not transcribed which means that an approach based on identifying differences in the phonetic realisation (across accents) of the same underlying phonological representations is not possible except in so far as an automatic speech recognition system might provide an accurate transcription. This in turn would seem to be unlikely given the poor and variable audio quality of the recordings.

The goal of this work then, is to find means by which some advantages of the linguistic-phonetic approach might be exploited within a GMM or phonotactic system. We cannot rely on the ASR systems used in the phonotactic method to establish reliable phonological labels (indeed it is the phoneme errors rather than the phoneme truths that give the method its power), so instead we turn to GMM systems.

A typical GMM system for accent recognition models individual speakers in terms of their difference to an average speaker described by a universal background model (UBM) built from a large population of speakers. A feature vector that represents the speaker can be calculated by observing the change in mixture means caused by MAP adaptation of the

UBM to the speaker. In previous accent recognition approaches this supervector of means can be used directly for recognition using a classifier [6,7], or can be reduced to i-vector form and processed by linear discriminant analysis to increase its sensitivity to accents over speakers and channels [8,9].

However, other useful information can be found in the UBM, for example the relative frequency of use of the mixtures might capture the relative frequency of phone types in the speech, analogous in some way to the phone frequencies estimated in phonotactic systems. A feature vector based on the Gaussian mixture posterior probabilities has been shown to be useful in accent recognition [6].

In ACCDIST, the phonetic properties of known phonological units are identified within one recording and compared to the properties of other units in the same recording. This table of segment similarities is then correlated across speakers such that the resulting similarity scores are relatively insensitive to speaker or channel. Could this idea be exploited within the GMM framework? If the Gaussian mixtures had systematic relationships with the underlying phonological forms, then the distances between mixtures might stand as proxies for the distances between phones used in ACCDIST. We can measure the distances between the MAP adapted means for the speaker and normalise the mixture distance table to create a feature vector that might be less sensitive to speaker and channel than the means themselves.

In summary we have identified three feature vector representations that might be used to classify accents in the GMM framework: Gaussian Mean Supervectors (GMS), Gaussian Posterior Probability Supervectors (GPPS) and Gaussian Mixture Distance Supervectors (GMDS). In this study we evaluate the relative merit of these representations on the Native Language development data set and also investigate the performance of the best performing systems on the chellenge test set.

3. Experimental Methods

3.1 Acoustic Feature Analysis

Acoustic analysis was chosen to be straightforward, since the main focus in this paper was on the different feature vector representations.

Each file was high-pass filtered at 100Hz to remove any DC component or main hum. The signals were normalised to unit variance and twelve mel-frequency-scaled cepstral coefficients plus RMS amplitude were extracted from 30ms windows every 10ms. Delta parameters were added to create a basic acoustic vector of 26 values.

Silent frames were detected as frames with RMS amplitude more than 50dB below the maximum in the file and removed. The acoustic parameters in the remaining frames in each recording were then normalised by z-scores (i.e. cepstral mean subtraction and variance normalisation).

3.2 Machine Learning Framework

Gaussian Mixture Modelling

Gaussian Mixture modelling was performed using the MSR Identity Toolbox [10]. Universal Background Models with diagonal covariance were built from all training samples with 64, 128, 256, 512, 1024 and 2048 mixtures. The MSR Identity Toolbox was also used for MAP adaptation of the UBM to individual speaker recordings.

Classification

Classification of feature vectors into accents was performed using a Support Vector Machine classifier [11]. The SVM systems were trained using a radial-basis function kernel. Best values for the hyper-parameters of cost and gamma were found by a simple grid search on the development test set. Grid spacing was a factor of two for each hyperparameter.

System Fusion

The classifier output for each test recording was stored as a list of 11 posterior probabilities, using the SVM facility for producing probability estimates. This allowed the classifications of different systems to be fused by a weighted linear combination trained on development data. System fusion was performed with the FoCal Multi-class toolkit [12].

3.3 Feature Vector Configurations

The feature vector construction is given below, and a summary may be found in Table 2.

Gaussian mean supervectors

The supervectors are concatenations of the acoustic feature means for each UBM mixture after adaptation with a specific recording. The length of each supervector is thus 26*number of mixtures. MAP adaptation of the mixture means only was performed with the MAP relevance factor τ=10.

Each element in the supervector is subsequently normalised by z-scores computed across all speakers prior to presentation to the SVM.

Gaussian posterior probability supervectors

The Gaussian posteriors for each UBM mixture are calculated as the average occupancy of the mixture taken over the whole speaker recording. Thus for T observations **o**, and J mixtures represented by means μ, covariance Σ and weight w, the average occupancy κ for mixture j is computed as:

$$\kappa_j = \frac{1}{T}\sum_{t=1}^{T}\frac{w_j p(o_t|\mu_j,\Sigma_j)}{\sum_{j=1}^{J} w_j p(o_t|\mu_j,\Sigma_j)}$$

The feature vector is then just the concatenation of the occupancy estimates for each recording, and has length equal to the number of mixtures. The Gaussian posterior probabilities are normalised to the range 0..1 computed over all speakers before presentation to the SVM.

Gaussian mixture distance supervectors

The distances between all pairs of mixture mean vectors is computed using the Bhattacharyya distance metric based on the means μ and variances σ^2 of each feature in the respective mixtures p & q:

$$d(p,q) = \frac{1}{4} ln\left(\frac{1}{4}\left(\frac{\sigma_p^2}{\sigma_q^2}+\frac{\sigma_q^2}{\sigma_p^2}+2\right)\right)+\frac{1}{4}\left(\frac{\left(\mu_p-\mu_q\right)^2}{\sigma_p^2+\sigma_q^2}\right)$$

The mixture distance is then just the sum over all features, and the supervector is the concatenation of the $\binom{J}{2}$ mixture distances. The distances in each individual supervector are

converted to z-scores to convert absolute to relative differences. Each element in the supervector is then further normalised by z-scores computed over all speakers prior to presentation to the SVM.

Table 2. *Characteristics of the best performing feature vector configurations found on development data.*

System Description	#GMM Mixtures	Vector Size	SVM Cost	SVM Gamma
Gaussian Means	512	13312	16	0.00005
Gaussian Posteriors	2048	2048	8	0.02
Mixture Distances	256	32640	8	0.0001

4. System Performance

Performance is measured in terms of unweighted average recall (UAR), which is the average accuracy when all accents are weighted equally. The relative quality of the different systems is also expressed in terms of the C_{llr} multi-class measure of goodness of log-likelihood ratios computed using the FoCal toolkit. Lower values of C_{llr} are better, and on this task a null system would have C_{llr}= 3.46bits.

Performance on the development data test set is shown in Table 3. Performance on the Native Language Challenge test set is shown in Table 4. An accent confusion matrix from the best performing system is shown in Figure 1.

Table 3. *Percentage accent recognition accuracy for different systems on development data. The C_{llr} value is measured after calibration with the FoCal toolkit.*

System Description	C_{llr} (bits)	UAR %
Mixture Distances	1.971	56.40
Gaussian Posteriors	1.679	63.07
Mean Supervector	1.560	66.45
Mean Supervector + Mixture Distances Fused	1.497	67.68
Mean Supervector + Gaussian Posteriors Fused	1.452	69.67
Mean Supervector + Gaussian Posteriors + Mixture Distances Fused	1.429	69.72

Table 4. *Percentage accent recognition accuracy for different systems on test data.*

System Description	UAR %
Baseline	47.5
Mean Supervector	68.84
Mean Supervector + Gaussian Posteriors + Mixture Distances Fused	69.80

	ARA	CHI	FRE	GER	HIN	ITA	JPN	KOR	SPA	TEL	TUR
ARA	61	1	5	2	1	3	3	2	4	2	2
CHI	1	62	4	0	2	0	6	4	4	0	1
FRE	4	1	55	4	0	4	4	1	4	0	3
GER	1	2	5	63	0	5	1	2	5	0	1
HIN	1	0	1	0	58	0	1	1	0	21	0
ITA	7	4	6	3	0	60	0	1	11	0	2
JPN	0	3	2	2	0	1	56	16	4	0	1
KOR	1	10	2	1	0	0	7	66	3	0	0
SPA	6	3	6	3	0	6	5	4	64	1	2
TEL	1	0	0	1	20	1	1	1	1	57	0
TUR	2	0	3	5	0	2	3	2	6	2	70

Figure 1: *Accent confusions of best performing system on development data. Rows = true accent, columns = recognised accent.*

5. Discussion

The nature of the Native Language challenge placed constraints on the kind of accent recognition system that could be deployed. Since ACCDIST could not be used we have explored whether ACCDIST-like within-speaker features would be useful within a GMM system. Effectively we tested how well GMM mixtures serve as proxies for phonological units. We found that while performance on accent recognition using mixture distances alone is worse than with mean supervectors or with posterior probability supervectors, mixture distances do contain useful accent information which marginally improve overall system performance after fusion. The contrast with the effectiveness of phone distances in ACCDIST is likely because single mixtures are rarely formed from the realisations of single phonological units, particularly when measured across speakers.

Overall performance on the Native Language challenge is somewhat worse than that found for regional accents [4, 7, 8]. There may be because the audio quality is poor and variable, because the language background of the speakers may be varied within the identified L1 language group, or because of variation in the proficiency of the speakers in English. It is well known that the nativeness of an L2 accent is strongly affected by the age at which speakers learn a second language [2]. The effect of proficiency on accent recognition was also reported for L2 speakers of Finnish in [9].

Finally, the language confusions shown in Figure 1 are interesting as they show confusions between related language groups. The greatest confusions are between Spanish & Italian, Hindi & Telugu, and between Chinese, Japanese & Korean. This pattern of confusions does provide some evidence that the accent recognition does exploit phonetic and phonological features of the L1 and does not just treat languages as arbitrary classes of sound.

6. Acknowledgements

Thanks to the organisers of the Interspeech 2016 Computational Paralinguistics Challenge for making this study possible.

7. References

[1] B. Schuller, S. Steidl, A. Batliner, J. Hirschberg, J. Burgoon, A. Baird, A. Elkins, Y. Zhang, E. Coutinho, K. Evanini, "The INTERSPEECH 2016 Computational Paralinguistics Challenge: Deception, Sincerity & Native Language", Interspeech 2016, San Francisco, USA, September 2016.

[2] J.E. Flege, C. Schirru, I.R.A. MacKay, "Interaction between the native and second language phonetic subsystems", Speech Communication., 40 (2003), 467–491.

[3] J.C. Wells, Accents of English 1 – An Introduction, Cambridge University Press, 1982.

[4] M. Huckvale, "ACCDIST: an accent similarity metric for accent recognition and diagnosis", in C. Muller (ed) Speaker Classification II, Springer-Verlag, Berlin/Heidelberg, Germany, pp 258-275.

[5] F. Eyben, M. Wöllmer, and B. Schuller, "openSMILE – The Munich Versatile and Fast Open-Source Audio Feature Extractor," in Proceedings of ACM Multimedia. Florence, Italy: ACM, 2010, pp 1459–1462.

[6] M. Bahari, R. Saeidi, H. Van Hamme, D. Van Leeuwen, "Accent recognition using i-vector, Gaussian mean supervector and gaussian posterior probability supervector for spontaneous telephone speech", ICASSP 2013, Vancouver, Canada , 7344-7348.

[7] A. Hanani, M. Russell, M. Carey, "Human and computer recognition of regional accents and ethnic groups from British English speech", J. Computer, Speech and Language 27 (2013) 59-74.

[8] A. DeMarco, S. Cox, "Iterative classification of regional British accents in i-vector space", Symposium on Machine Learning in Speech and Language Processing, MLSLP 2012, Portland, Oregon, USA, September 14, 2012

[9] H. Behravan, V. Hautamaki, T. Kinnunen, "Factors Affecting i-Vector Based Foreign Accent Recognition: a Case Study in Spoken Finnish", Speech Communication, 66 (2015) 118-129.

[10] S. Sadjadi, M. Slaney, L. Heck, "MSR Identity Toolbox v1.0: A MATLAB Toolbox for Speaker-Recognition Research", http://www.signalprocessingsociety.org/technical-committees/list/sl-tc/spl-nl/2013-11/IdentityToolbox/

[11] C. Chang and C. Lin. "LIBSVM: a library for support vector machines", ACM Transactions on Intelligent Systems and Technology, 2 (2011) 27:1-27:27.

[12] N. Brümmer, "FoCal Multi-class Toolkit", https://sites.google.com/site/nikobrummer/focalmulticlass

INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

Native Language Detection using the I-Vector Framework

Mohammed Senoussaoui[1], *Patrick Cardinal*[1], *Najim Dehak*[2], *Alessandro L. Koerich*[1]

[1]École de Technologie Supérieure, Montréal, Canada
[2]Johns Hopkins University, Baltimore, USA

mohammed.senoussaoui.1@ens.etsmtl.ca, patrick.cardinal@etsmtl.ca
ndehak3@jhu.edu, alessandro.koerich@etsmtl.ca

Abstract

Native-language identification is the task of determining a speaker's native language based only on their speeches in a second language. In this paper we propose the use of the well-known i-vector representation of the speech signal to detect the native language of an English speaker. The i-vector representation has shown an excellent performance on the quite similar task of distinguishing between different languages. We have evaluated different ways to extract i-vectors in order to adapt them to the specificities of the native language detection task. The experimental results on the 2016 ComParE Native language sub-challenge test set have shown that the proposed system based on a conventional i-vector extractor outperforms the baseline system with a 42% relative improvement.
Index Terms: native language, i-vector, computational paralinguistics, ComParE.

1. Introduction

The task of recognizing the native language of a speaker or his mother tongue is called Native language identification (NLI). This task can be considered similar to the Language Identification task (LID), which consists of identifying the language spoken in a given speech segment. It is well known that even if a multilingual speaker is able to speak correctly a second language (L2), his native language (L1) can still influence the pronunciation of the second one. This speaking behavior can be seen as soft biometrics information and it can be helpful for identifying the speaker identity. An automatic NLI system can be useful for several other speech applications such as speech and speaker recognition. In such applications, the speaker?s accent is considered as a nuisance that needs to be compensated.

Several levels of information extracted from speech signal have been studied for NLI. Acoustics features and prosodic cues [1] are considered the most popular characteristics for NLI. Phonetic characteristics such as phonemes frequent appearance and their duration also provide a good indication of speaker?s mother tongue [2]. Note that the same task can be achieve by analysing the text instead of the speech signal [3].

Since seven years ago, the i-vector framework became the state-of-the-art speech representation for text-independent speaker recognition as well as for many other speech related fields such as language recognition [4, 5, 6, 7] and speaker adaptation for automatic speech recognition [8]. The i-vector is an elegant way to represent speech segments of variable lengths in the same space of moderate dimension (typically in the range of hundreds) [9]. In this space, several kind of machine learning approaches can be applied to solve different kind of audio classification problems. Therefore, the i-vector will be our main speech representation investigated in this NLI challenge.

In this paper we propose two different i-vector extraction strategies to the problem of native language detection: language-independent and language-dependent. In the former, a unique language-independent i-vector extractor is estimated for all the native language classes. The latter strategy consists of training one native language-dependent i-vector for each native language class [10]. Furthermore, we also evaluated three normalization strategies for the language-dependent i-vector representation. The different i-vector implementations are evaluated and compared within the scoop of the ninth edition Computational Paralinguistics Challenge (ComParE) [11, 12].

The remainder of this paper is organized as follows. Section 2 reviews the conventional i-vector speech representation as well as its related intersession compensation methods. Section 3 describes the language-dependent i-vector representation and the three related intersession compensation strategies. In Section 4, all the experiments and the results are presented and discussed. Finally, in the last section we present some future research directions as well as the conclusions of this work.

2. Conventional i-vector framework

The i-vector framework [9] is based on modeling speech segments using a universal background model (UBM), which is typically a large Gaussian Mixture Models (GMM) where the UBM is usually trained on a large quantity of data to represent general feature characteristics. This model plays the role of a prior on how all sounds look like. The i-vector approach is a powerful technique that summarizes all the updates happening during the adaptation of the UBM mean components to a given speech recording. All this information is modeled in a low dimensional subspace referred to as the total variability space. In the i-vector framework, each speech utterance can be represented by a GMM supervector, which is assumed to be generated as follows:

$$\boldsymbol{M} = \boldsymbol{m} + \boldsymbol{T}\boldsymbol{x} \tag{1}$$

where $\boldsymbol{m}$ is the language and dialect independent supervector (which can be taken to be the UBM supervector), $\boldsymbol{T}$ is a rectangular matrix of low rank and $\boldsymbol{x}$ are the factors that best describe the utterance-dependent mean offset. The vector $\boldsymbol{x}$ is treated as a latent variable with the i-vector being its maximum-a-posteriori point estimate. The subspace matrix $\boldsymbol{T}$ is estimated using maximum likelihood on a large training set [9]. Figure 1 summarizes the steps used to extract the i-vectors.

It is well known that the i-vector space models a wide variety of variability embedded in the speech signal and this is mainly due to the fact of using unsupervised methods for the

http://dx.doi.org/10.21437/Interspeech.2016-1473

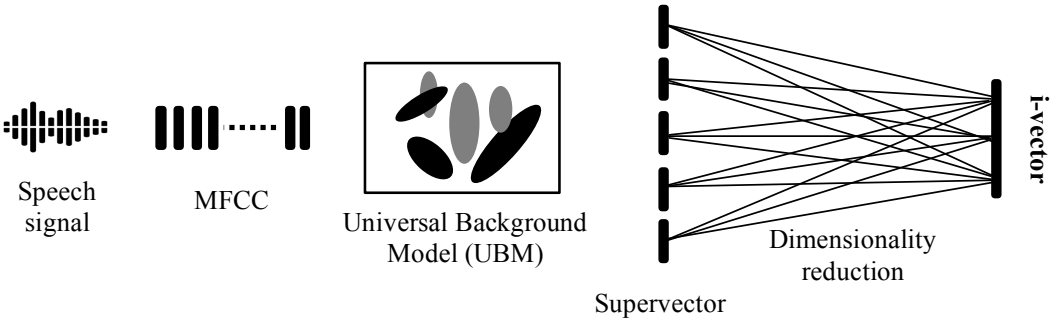

Figure 1: *Conventional i-vector feature extraction from a given speech signal.*

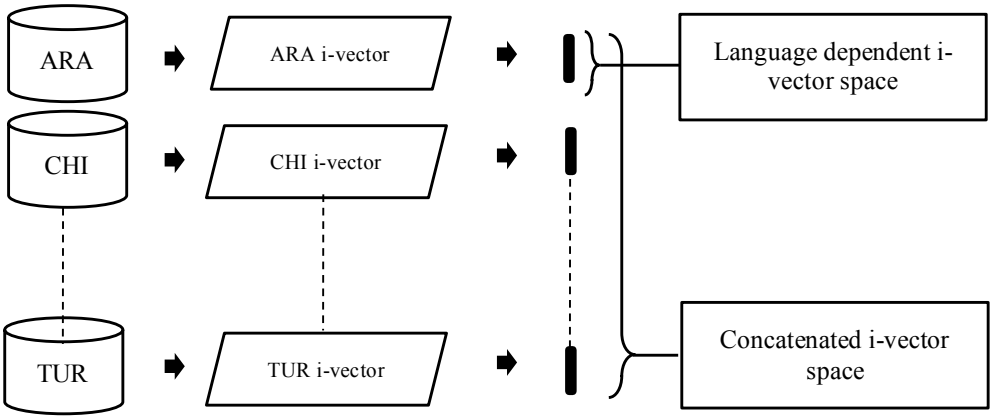

Figure 2: *Per-Native-language i-vector feature spaces.*

supervector space dimensionality reduction. In order to extract any specific information from this space (e.g. the language nativeness), task-dependent normalization (i.e. intersession compensation) needs to be performed in the raw i-vector before carrying out any recognition task. The main objective of the task-dependent normalization is the mapping of the raw i-vector into a new space that maximizes only the useful information and minimizes the effect of unwanted variability. In the i-vector context, the most known normalization procedure [13] is based on a combination of the Linear Discriminant Analysis (LDA) followed by the Within Class Covariance Normalization (WCCN) [14]. Moreover, it has been found that the normalization of the i-vector to the unit length (i.e. norm = 1) is helpful for speaker recognition systems [15]. The detailed procedure of i-vector space normalization adopted in this work is depicted in Algorithm 1.

Algorithm 1 i-vector space normalization

1: Normalize the length of the i-vector $\boldsymbol{x}$ to one.
2: Map the length normalized i-vector to the LDA subspace using the optimal dimension (in our case 10 was the best dimension).
3: Normalize again the length of the LDA normalized i-vector.
4: Rotate the length-, LDA- and length-normalized i-vector using the Cholesky decomposition of the inverse Within Class Covariance.

3. Language-dependent i-vector

A class-dependent i-vector extractor is trained using each class dependent subset of data individually in order to model the class aspect in our native language dataset. This approach is similar to an approach that has been already successfully implemented for the problem of language Identification [10]. In this modeling, eleven i-vector extractors (including UBMs and the total variability matrices $\boldsymbol{T}$) were trained on the ComParE native language training set (i.e. the ETS Corpus Of Non-Native Spoken English) [11]. All the obtained language-dependent subspaces can be exploited in two different ways: i) Each language-dependent subspace can be processed individually; ii) All language-dependent i-vectors can be concatenated to create a larger and unique subspace (c.f. Figure 2). In this work the dimension of the language-dependent space is chosen to be 50, which means that the dimension of the concatenated space is $50 \times 11 = 550$.

We have adopted three different strategies to deal with undesirable variability in the different i-vector spaces. However, all these strategies are based on the same normalization technique as described in Algorithm 1. The first strategy consists

Table 1: Unweighted Average Recall (UAR %) for the development and test sets.

	Conventional i-vectors		Language-dependent i-vector						Baseline	
			Norm. Strategy I		Norm. Strategy II		Norm. Strategy III			
	Devel	Test	Devel	Test	Devel	Test	Devel	Test	Devel	Test
UBM 512 dim 400	**66.8**	**66.4**	-	-	-	-	-	-	45.1	47.5
UBM 256 dim 50	-	-	38.2	-	43.9	-	44.6	-		
UBM 128 dim 50	-	-	43.2	-	45.9	-	46.9	-		
UBM 64 dim 50	-	-	51.8	-	45.2	-	49.2	-		
UBM 32 dim 50	-	-	53.4	-	44.9	-	45.8	-		
UBM 16 dim 50	-	-	49.7	-	43.6	-	44.9	-		

Table 2: Confusion matrix in percent for the development set using the conventional i-vector.

	ARA	CHI	FRE	GER	HIN	ITA	JAP	KOR	SPA	TEL	TUR
ARA	52	1	6	4	3	5	2	2	4	1	6
CHI	0	62	0	1	4	0	4	10	3	0	0
FRE	3	1	51	8	2	5	2	1	6	0	1
GER	1	4	4	72	0	0	0	1	2	0	1
HIN	0	0	0	0	56	0	0	1	0	26	0
ITA	1	0	9	3	0	72	0	3	5	0	1
JAP	0	6	4	0	1	1	54	16	3	0	0
KOR	1	12	0	1	0	2	15	57	1	0	1
SPA	5	5	9	3	1	13	8	9	39	3	5
TEL	1	0	0	0	22	0	0	2	1	56	1
TUR	1	0	1	5	1	4	3	1	4	1	74

of running the space normalization procedure (c.f. Algorithm 1) directly on the concatenated language-dependent space to generate a 10-dimensional normalized i-vector. In the second strategy, we first apply separately the normalization procedure to each language-dependent subspace to obtain eleven 10-dimensional normalized i-vectors which are then stacked to form a single 110-dimensional i-vector. In the third strategy, we simply carry out the normalization procedure on the output of the second strategy (i.e. the 110-dimensional i-vector). The main objective behind this strategy is to map the concatenated space into a more homogeneous space of dimension 10.

4. Experimental Results

The performance of the proposed approach was evaluated on the ETS Corpus of Non-native spoken English which includes more than 64 hours of speech from 5,132 non-native speakers of English, with eleven different L1 backgrounds (Arabic (ARA), Chinese (CHI), French (FRE), German (GER), Hindi (HIN), Italian (ITA), Japanese (JAP), Korean (KOR), Spanish (SPA), Telugu (TEL), and Turkish (TUR)). The dataset was divided into three stratified partitions: 3,300 instances (64%) were selected as training set, 965 instances (19%) for the development set, and 867 responses (17%) used as test data. As measure of performance, we employ Unweighted Average Recall (UAR). For more details on this dataset please refer to [11].

The i-vectors have been extracted using Kaldi, a free open source toolkit for speech recognition [16]. The experiments have been conducted with support vector machine (SVM) using the sequential minimal optimization (Weka toolkit [17]) and with neural networks (NN) using the TensorFlow library [18].

4.1. Results with SVM

The fist step of the i-vector estimation process consists of extracting a sequence of short-term features (Figure 1). In this work, the Mel Frequency Cepstral Coefficients (MFCCs) augmented by their Shifted Delta Cepstral (SDC) are used as short-term features. The SDC features are able to model long-term speech temporal information which seems to be very useful for the Language Identification task [19]. The parameters used to to generate the SDC features were set as follows: The number of the cepstral coefficients was set to 20 (i.e. static MFCC); the distance between blocks was set to 3; the size of delta advance and delay was set to 1 and the number of blocks in advance of each frame to be concatenated was set to 7. This setup produces feature vectors of 160 dimensions.

Table 1 shows the results achieved by both conventional i-vector and language dependent i-vectors. A linear kernel had been used for all experiments. The best result was achieved with the language-independent i-vector (UAR of 66.4%) which also outperforms the baseline system (UAR of 47.5%). Regarding the language-dependent i-vectors, since the amount of data per language is very limited, several experiments have been conducted with different UBM sizes. It is not clear from the different UBM size performances that there is an optimal configuration and it mainly depends on the normalization strategy. The overall results show that the language-independent approach seems to be less effective for this task than using language-independent i-vectors.

The i-vector extractor used in these experiments is based on GMMs with full covariance UBMs, which requires a large amount of data for training. In another set of experiments, we have used UBMs with diagonal covariance matrices. Unfortu-

nately, no improvement has been observed. Another possibility to explore is the reduction of the MFCC-SDC features that are quite large (160 dimensions) for a GMM.

4.2. Results with NN

Table 3 summarizes the two best results obtained with NNs. These results are very close to those obtained with a SVM with i-vector as input, with a slight improvement of 0.2%. The best architectures for the NNs were with a single hidden layer and with two hidden layers. However, both architectures achieved very close UARs. The best result was achieved by combining two neural networks. The first neural network had one hidden layer of 1024 neurons with the sigmoid activation function and 11 neurons in the softmax output layer. The output of such a NN was concatenated with the i-vector and used as input for a second NN with one hidden layer of 256 neurons. This architecture led to the best result on the development set with a UAR of 68.4%.

Table 3: Results on neural network (NN) measured by the Unweighted Average Recall (UAR %).

Description	UAR (Devel)
ivec	67.0%
ivec + dist	68.4%

4.3. Combining SVM and NN

The confusion matrix (Table 2) generated by the system using both the i-vector and the output layer of a NN with 256 neurons in a single hidden layer shows that there is a high confusion between some languages (e.g. Hindi and Telugu). The confusion matrices of other systems exhibit the same behavior, meaning that the combination should improve the results.

Therefore, the last experiments were carried out to explore the use of the NN as a feature extractor for the SVM. Table 3 shows the main results achieved with this approach. A conventional i-vector was used as input for a single hidden layer NN. Results with two or three hidden layers are not reported because they were very similar or worst than those obtained with a single hidden layer.

Table 4: Results with NN as feature extractor for the SVM (measured by the Unweighted Average Recall (UAR %)).

Description	UAR (Devel)
1024 neurons	66.5%
512 neurons	67.4%
256 neurons	67.6%
Output dist, 1024 neurons	67.2%
Output dist, 256 neurons	67.2%
ivec + dist (256)	66.9%
ivec + dist +output (256)	66.4%
ivec + dist (1024)	67.3%
ivec + dist (1024) on NN	68.4%

The experiments use the output of the hidden layer (before the activation function) as input of the SVM. The best UAR achieved with this scheme is 67.6%, which is almost 1% higher than the best result achieved by the SVM using the i-vector as input. This result shows that the NN is able to extract complementory information from the i-vector. Other experiments have explored the use of NN output layer (after the softmax) as input of the SVM. The results achieved with this schema are almost as good as those obtained with the hidden layer and better than the best result obtained with the SVM alone. Note that this result also represents a small improvement over the NN alone (67%). The third experiment reported in Table 3 has explored the combination of the i-vector and the features of NN for the SVM. This approach led to a very small improvement with a UAR of 67.3%. Finally, the last experiment was to use another NN with the concatenation of the i-vector and the NN output layer as input. This configuration achieved the best result on the development set with a UAR of 68.4%.

4.4. Results on the test set

Considering the limited number of five trials to evaluate the proposed approach on the test set for the 2016 ComParE Native language sub-challenge, we have selected the approaches that have provided the highest UAR on the development set. The first three rows of Table 5 correspond to the best approaches on the development set using the SVM classifier. The fourth row was achieved by a SVM for which the input is the concatenation of SVM output confidence scores and NN output distribution. Both classifiers have a conventional i-vector as input. Surprisingly, we did not have observed any improvement with such a fusion of scores for the test set. The fifth row of Table 5 corresponds to the approach for which we have achieved the best UAR on the development set. Unfortunately, such an approach did not provided the best result on the test set as we should expect. Compared to the other approaches, it seems that the NN was overfitted during the training process. Finally, the last row shows the UAR obtained by the official baseline system [11].

Table 5: Results on the test set by the Unweighted Average Recall (UAR %).

Description	UAR (Test)
ivec on SVM	66.4%
ivec on NN	66.6%
ivec + dist (1024) on SVM	67.4%
score fusion on SVM	67.4%
ivec + dist (1024) on NN	67.1%
Official baseline [11]	47.5%

5. Conclusions

In this paper we have explored two different implementations of the well-known i-vector representation of speech to deal with the problem of native language (L1) identification for an English (L2) speaker. The first implementation based on the conventional language-independent i-vector outperforms by far the baseline system (UAR of 66.4% vs. 47.5% on the test set) on the development and test sets of the ComParE 2016 challenge. The combination of i-vector with features extracted by a NN using the same i-vector representation as input led to UAR up to 67.4% on the test set, an improvement of 1% absolute.

6. Acknowledgments

The authors acknowledge funding from FRQNT and would like to thank Jim Glass of the CSAIL Lab for giving us access to their cluster.

7. References

[1] R. Todd, "On Non-Native Speaker Prososdy: Identifying 'Just Noticeable-Differences' of Speaker-Ethnicity," in *The 1st International Conference on Speech Prosody*, 2002.

[2] E. Shriberg, L. Ferrer, S. Kajarekar, N. Scheffer, A. Stolcke, and M. Akbacak, "Detecting nonnative speech using speaker recognition approaches," in *Odyssey Workshop*, 2008.

[3] V. Kríz, M. Holub, and P. Pecina, "Feature extraction for native language identification using languagemodeling," in *Recent Advances in Natural Language Processing, RANLP 2015, 7-9 September, 2015, Hissar, Bulgaria*, 2015, pp. 298–306.

[4] N. Dehak, P. A. Torres-Carrasquillo, D. Reynolds, and R. Dehak, "Language Recognition via I-Vectors and Dimensionality Reduction," in *INTERSPEECH 2011*, Florence, Italy, Aug. 2011, pp. 857–860.

[5] D. Martnez, L. Burget, L. Ferrer, and N. Scheffer, "ivector-based prosodic system for language identification," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, March 2012, pp. 4861–4864.

[6] M. Soufifar, M. Kockmann, L. s Burget, O. Plchot, O. Glembek, and T. Svendsen, "ivector approach to phonotactic language recognition," in *INTERSPEECH*, 2011.

[7] D. Martnez, E. Lleida, A. Ortega, and A. Miguel, "Prosodic features and formant modeling for an ivector-based language recognition system," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, May 2013, pp. 6847–6851.

[8] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, Dec 2013, pp. 55–59.

[9] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech and Language Processing*, 2011.

[10] P. Matejka, O. Plchot, M. Soufifar, O. Glembek, L. F. D'Haro, K. Vesel, F. Grzl, J. Z. Ma, S. Matsoukas, and N. Dehak, "Patrol team language identification system for darpa rats p1 evaluation." in *INTERSPEECH*. ISCA, 2012, pp. 50–53.

[11] B. Schuller, S. Steidl, A. Batliner, J. Hirschberg, J. K. Burgoon, A. Baird, A. Elkins, Y. Zhang, E. Coutinho, and K. Evanini, "The interspeech 2016 computational paralinguistics challenge: Deception, sincerity & native language," in *Submitted to INTERSPEECH 2016*, Sept 2016.

[12] B. Schuller, A. Batliner, S. Steidl, and D. Seppi, "Recognising realistic emotions and affect in speech: State of the art and lessons learnt from the first challenge," *Speech Communication*, vol. 53, no. 910, pp. 1062 – 1087, 2011, sensing Emotion and Affect - Facing Realism in Speech Processing.

[13] N. Dehak, R. Dehak, P. Kenny, N. Brümmer, P. Ouellet, and P. Dumouchel, "Support vector machines versus fast scoring in the low-dimensional total variability space for speaker verification." in *Interspeech*, vol. 9, 2009, pp. 1559–1562.

[14] A. O. Hatch, S. S. Kajarekar, and A. Stolcke, "Within-class covariance normalization for svm-based speaker recognition." in *Interspeech*, 2006.

[15] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems." in *Interspeech*, 2011, pp. 249–252.

[16] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, Dec. 2011, iEEE Catalog No.: CFP11SRW-USB.

[17] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: An update," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, Nov. 2009. [Online]. Available: http://doi.acm.org/10.1145/1656274.1656278

[18] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: http://tensorflow.org/

[19] P. Torres-Carrasquillo, E. Singer, M. Kohler, R. Greene, D. Reynolds, and J. Deller, "Approaches to language identification using gaussian mixture models and shifted delta cepstral features," in *ICSLP*, Sept 2002, pp. 89–92.

INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

Exploiting phone log-likelihood ratio features for the detection of the native language of non-native English speakers

Alberto Abad[1,2], *Eugénio Ribeiro*[1,2], *Fábio Kepler*[1], *Ramon Astudillo*[1,3], *Isabel Trancoso*[1,2]

[1]L[2]F - Spoken Language Systems Lab, INESC-ID Lisboa
[2]IST - Instituto Superior Técnico, University of Lisbon
[3]Unbabel Inc.
alberto.abad@l2f.inesc-id.pt

Abstract

Detecting the native language (L1) of non-native English speakers may be of great relevance in some applications, such as computer assisted language learning or IVR services. In fact, the L1 detection problem closely resembles the problem of spoken language and dialect recognition. In particular, log-likelihood ratios of phone posterior probabilities, known as Phone LogLikelihood Ratios (PLLR), have been recently introduced as features for spoken language recognition systems. This representation has proven to be an effective way of retrieving acoustic-phonotactic information at frame-level, which allows for its use in state-of-the-art systems, that is, in i-vector systems. In this paper, we explore the use of PLLR-based i-vector systems for L1 native language detection. We also investigate several linear and non-linear L1 classification schemes on top of the PLLR i-vector front-ends. Moreover, we compare PLLR based systems with both conventional phonotactic systems based on n-gram modelling of phoneme sequences and acoustic-based i-vector systems. Finally, the potential complementarity of the different approaches is investigated based on a set of system fusion experiments.

Index Terms: computational paralinguistics, native language recognition, PLLR, i-vectors.

1. Introduction

This paper presents INESC-ID's system for the Native Language (N) Sub-Challenge of the Computational Paralinguistics Challenge (ComParE) 2016 [1]. The task consists of identifying the native language (L1) of non-native English speakers. Detecting the L1 of the speakers is relevant for spoken language applications, since it provides information about the users that can be used to improve the interaction and the application performance. For instance, L1-specific ASR models can be used to improve recognition accuracy. Also, cultural information deduced from the identified L1 can lead to a more personal and context-aware dialog. Finally, accurate L1 detection can also play a role in software tools aimed at Computer Assisted Language Learning (CALL).

The data for the challenge is the ETS Corpus of Non-Native Spoken English, which contains 45-second answers from speakers with eleven different L1 backgrounds – Arabic, Chinese, French, German, Hindi, Italian, Japanese, Korean, Spanish, Telugu and Turkish – in the context of the TOEFL iBT® assessment. 3300 instances (41.3 hours) were selected for training, 965 (12.1 hours) for the development set, and 867 (10.8 hours) for testing.

In this paper, we explore the performance of i-vector systems based on Phone Log-Likelihood Ratios (PLLR) [2] on this task. We opted for this approach since it has been recently introduced and proved very effective in similar tasks. We also explore acoustic-based i-vector systems, and phonotactic systems based on Phone Recognition followed by Language Modelling (PRLM) for a matter of comparison. Furthermore, we investigate linear and non-linear models for language characterization based on i-vectors as an alternative to the simple, but effective, single Gaussian mixture model proposed in [3]. Finally, we also explore the combination of multiple subsystems through fusion based on logistic regression.

The following section presents some previous work related to the task. After that, Sections 3, 4, and 5 thoroughly describe the features, the classifiers, and the calibration and fusion approaches explored in this work for L1 identification. Results are presented and discussed in Section 6. Finally, the conclusions of the work are stated in Section 7.

2. Related Work

Automatic native language identification is a relatively recent task. For textual data, the most common approaches explore features related to spelling errors and the quality of writing, such as character, word, and POS n-grams, function words, and dependency relations [4, 5, 6, 7].

On the other hand, for spoken interactions, the literature is still very scarce. Different aspects of L2 speech may be explored, both at the segmental and supra-segmental level. The first level concerns the mispronunciations that are mainly due to the fact that some L2 phonemes are missing from the L1 inventory of speech sounds, which causes the non-native speakers to often replace an L2 phoneme by an L1 phoneme with a similar place or manner of articulation. Teaching L2 pronunciation traditionally focuses on segmentals. Supra-segmental features, related to intonation, are particularly hard to acquire for adults in L2. Hence, such features are also often used by humans to identify the native language. In [8], for instance, the authors use both cepstral and prosodic features, to identify 3 South Indian languages as L1 in non-native English speech.

The native language identification task is similar to the language, accent, and dialect identification tasks in Spoken Language Recognition (SLR). Thus, it makes sense to describe some of the most successful approaches used on those tasks. SLR approaches can be generally classified according to the source of information that they rely on. The most successful

Work partially supported by FCT project UID/CEC/50021/2013 and EU project H2020-EU.3.7. 653587

http://dx.doi.org/10.21437/Interspeech.2016-1491

systems are based on the exploitation of the acoustic [3, 9] and phonotactic [10, 11] characteristics of each language. While the first govern how a given language sounds, the latter are the rules that govern the possible phone combinations in a language. Usually, the combination of different sources of knowledge and systems of different characteristics tends to provide increased language recognition performances [12]. Recently, a new set of features known as Phone Log-Likelihood Ratios (PLLR) have been introduced for SLR [2]. These features convey frame-by-frame acoustic-phonetic information, which can be used in conventional acoustic systems like those based on the well-known Total Variability Factor Analysis (i-vector) approach [13]. The use of PLLR in SLR has been proven to lead to one of the best individual system results reported on relevant benchmarks [14].

3. Features for L1 recognition

In this work we decided to use acoustic-phonetic, acoustic, and phonotactic features for L1 recognition. The specific features of each category are described below.

3.1. Acoustic-Phonetic Features

In terms of acoustic-phonetic features, we use PLLR features as described in [14]. This section presents a brief summary of the PLLR extraction process, including post-processing stages, together with implementation details of the phonetic decoders.

3.1.1. PLLR definition

Considering a phone decoder that provides frame-by-frame phone posteriors p_i for each phone unit ($1 \leq i \leq N$), so that $\sum_{i=1}^{N} p_i = 1$ and $p_i \in [0, 1]$, the PLLR features are computed from these phone posteriors as follows [15]:

$$r_i = \text{logit}(p_i) = \log \frac{p_i}{(1-p_i)} \quad i = 1, \ldots, N. \qquad (1)$$

One of the main advantages of the transformation of phone posteriors into PLLRs is that it allows for the gaussianization of the resulting features, which makes them more suitable for typical GMM modelling. However, as pointed out in [16], the PLLR feature space as defined by Equation 1 is bounded, which limits the distribution of the features. In order to avoid the bounding effect, PLLRs are projected as described in [14]. Then, Principal Component Analysis (PCA) is applied to decorrelate the parameters and to reduce the feature dimensionality. After that, shifted delta cepstra (SDC) coefficients [17] are obtained. Following [18], the SDC configuration for PLLR features is 13-2-3-7, resulting in a feature vector of 104 components.

3.1.2. Phonetic classifiers

The phonetic classifiers used in this work are part of our hybrid Automatic Speech Recognition (ASR) system, AUDIMUS [19]. The phonetic models are neural networks of the MultiLayer Perceptron (MLP) type trained to estimate the posterior probabilities of the different phonemes of a specific language for a given input speech frame (and its context). In this case, we have used four language-dependent phonetic decoders: European Portuguese (*pt*), Brazilian Portuguese (*br*), European Spanish (*es*) and American English (*en*). For each phonetic decoder, an independent set of PLLR features is obtained based on the generated frame-by-frame posterior probabilities. In practice, frames that have silence as the most probable class are removed.

Each of the recognizers combines four MLP outputs trained with Perceptual Linear Prediction features (PLP, 13 static + first derivative), PLP with log-RelAtive SpecTrAl speech processing features (PLP-RASTA, 13 static + first derivative), Modulation SpectroGram features (MSG, 28 static) and Advanced Front-End from ETSI features (ETSI, 13 static + first and second derivatives). Each MLP network is characterized by the size of its input layer that depends on the particular parameterization and the frame context size (13 for PLP, PLP-RASTA and ETSI; 15 for MSG), the number of units of the two hidden layers (500), and the size of the output layer. In this case, only monophone units are modelled, resulting in MLP networks of 41 (39 phonemes +1 silence + 1 breathing) soft-max outputs in the case of *en*, 39 for *pt* (38 phonemes + 1 silence), 40 for *br* (39 phonemes + 1 silence) and 30 for *es* (29 phonemes + 1 silence). The output size corresponds to the length of the PLLR feature vectors before dimensionality reduction.

The language-dependent MLP networks were trained with different amounts of annotated data. For the *pt* acoustic models, 57 hours of Broadcast News (BN) down-sampled data and 58 hours of mixed fixed-telephone and mobile-telephone data were used. The *br* models were trained with around 13 hours of BN down-sampled data. The *es* networks used 36 hours of BN down-sampled data and 21 hours of fixed-telephone data. The *en* system was trained with the HUB-4 96 and HUB-4 97 down-sampled data sets, that contain around 142 hours of TV and Radio Broadcast data.

3.2. Acoustic Features

Acoustic features typically used for SLR have also been adopted in this work. In particular, we used shifted delta cepstra (SDC) of Mel-frequency Cepstrum Coefficients (MFCC) [17]. First, 7 MFCC static features are obtained and SDC features with a 7-1-3-7 configuration are computed, resulting in a feature vector of 56 components. Then, frames of each segment that are simultaneously labeled as silence by the four previously described language-dependent phonetic decoders are removed. Finally, cepstral mean normalization is applied in a per segment basis.

3.3. Phonotactic Features

In order to model the phonotactics of each target native language, a phonetic tokenization is obtained using the previously described classifiers. Likewise, we use the same 4 language-dependent ASR systems: *pt*, *br*, *es* and *en*. In this case, a decoding process is performed for each speech sequence (in contrast to simple frame-by-frame posterior probability computation). The AUDIMUS decoder is based on a weighted finite-state transducer (WFST) approach [20]. A phone-loop grammar with minimum phoneme duration of three frames is used to obtain the phonetic sequences.

4. Front-end models for L1 characterization

4.1. The i-vector front-end

Total-variability modelling [13] has emerged as one of the most powerful approaches to the problems of speaker and language verification. In this approach, the variability present in the high-dimensional GMM supervector is jointly modelled as a single low-rank total-variability space. The low-dimensionality total variability factors extracted from a given speech segment form a vector, named i-vector, which represents the speech segment

in a very compact and efficient way. Thus, the total-variability modelling is used as a factor analysis based front-end extractor. The success of i-vector based speaker recognition has motivated the investigation of its application to other related fields, including language recognition [3, 9], where it has become the current de facto standard for acoustic SLR. In this work, we have developed i-vector based LR sub-systems very similar to the one in [3], where the distribution of i-vectors for each language is modelled with a single Gaussian.

It is worth mentioning that, as an alternative to the i-vector approach, classifiers based on feed-forward networks were also trained for acoustic and acoustic-phonetic features. These approaches employed various techniques to account for the variable length of the feature matrices including Convolutional and Recurrent neural networks (CNN, RNN). In general these approaches led to poor results compared to i-vector modelling.

4.1.1. Total variability and i-vector extraction

The first step of i-vector system development consists of training a GMM-UBM. In this case, a GMM-UBM of 1024 mixtures is trained using all the training data available for the challenge. Then, the total variability factor matrix (**T**) is estimated according to [21]. The dimension of the total variability sub-space is fixed to 400. Next, zero and first-order sufficient statistics of the training set are used for training **T**. In order to do so, 10 Expectation-Maximization (EM) iterations of consecutive Maximum Likelihood (ML) and minimum divergence estimation updates are applied. The covariance matrix is not updated in any of the EM iterations. The estimated **T** matrix is used for extraction of the total variability factors of the processing speech segments as described in [21]. Additionally, we apply i-vector centering and whitening [22] that is known to contribute to a reduction of the channel variability. Finally, the resulting factor vectors are normalized to be of unit length, which we will henceforth refer to as i-vectors.

4.1.2. Language modelling and scoring

Like in [3], all the extracted i-vectors of each target L1 language are used to train a single mixture Gaussian distribution with full covariance matrix shared across different target languages. As an alternative to this approach, Log-linear and non-linear classifiers based on feed-forward networks were also investigated. In fact, it could be observed that the i-vector front-end already provided a very good separation of the classes which led to similar results for the different modelling techniques. For this reason, experimental results on alternative classifiers on the top of i-vectors are not reported. Finally, for a given test i-vector, each Gaussian model is evaluated and log- likelihood scores are obtained. The 11 likelihoods of the 11 L1 target languages form a vector of scores that is used for later calibration and fusion.

4.2. PRLM-LR sub-systems

The Phone Recognition followed by Language Modelling (PRLM) systems used in this work exploit the phonotactic information extracted by the four individual tokenizers described previously. For each target L1 language and for each tokenizer a different phonotactic *n-gram* language model is trained using the phonetic sequences of the challenge training data set. For that purpose, the SRILM toolkit has been used[1] and 3-gram back-off models smoothened using Witten-Bell discounting are obtained. During test, the phonetic sequence of a given speech signal is extracted with the phonetic decoders and the likelihood of each target language model is evaluated. Like the i-vector sub-systems, the likelihoods of the 11 L1 target languages form a vector of scores that is later used for calibration and fusion.

Similarly to [23], we tried likelihood scores obtained with phonotactic models of an arbitrary set of languages trained on external data as possibly discriminant features for L1 recognition. This approach, however, did not reveal useful for this task.

5. Calibration and Fusion Back-End

Calibration and fusion was carried out using a combination of linear Gaussian Back-Ends (GBE) followed by a Linear Logistic Regression (LLR). GBE were applied after every single subsystem to transform the score-vector $\mathbf{x}_i$ into a 11-element log-likelihood vector $\mathbf{s}_i$, corresponding to each of the target languages, using the following equation:

$$\mathbf{s}_i = \mathbf{A}_i \mathbf{x}_i + \mathbf{o}_i, \tag{2}$$

where $\mathbf{A}_i$ is the transformation matrix for system i and $\mathbf{o}_i$ is the offset vector. Notice that in this work the dimension of $\mathbf{x}_i$ for all the considered sub-systems is 11. Nevertheless, we kept the GBEs given that they contributed for improved language identification in the development experiments.

Then, LLR was used to fuse the log-likelihood outputs generated by the linear GBEs of the selected sub-systems to produce fused log-likelihoods **l** as follows:

$$\mathbf{l} = \sum_i \alpha_i \mathbf{s}_i + \mathbf{b}, \tag{3}$$

where α_i is the weight for sub-system i and **b** is the language-dependent shift. For this challenge, the language with the highest fused log-likelihood is the hypothesized L1 language.

During the development of our systems, the GBEs and the LLR fusion parameters were trained and evaluated on the development set using a kind of 2-fold cross-validation [24]: development data was randomly split in two halves, one for parameter estimation and the other for assessment. This process was repeated using 10 different random partitions so that the mean and variance of the systems' performance could be computed. This method allowed for a comparison and ranking of the different sub-systems under study. Then, for the trial submissions, no partition was made and all the development data was used to simultaneously calibrate the GBEs and the LLR fusion. Calibration was carried out using the FoCal Multi-class Toolkit [2].

6. Experimental Results

6.1. Baseline System

Similarly to previous years, the baseline system proposed for the Nativeness challenge employs the ComParE features set. This comprises 6373 features resulting from the computation of various functionals over low-level descriptor (LLD) contours. The features are computed with openSMILE [25] employing the configuration file `IS13_ComParE.conf`. All features were normalized to the mean and standard deviation of the training set. The classifier used is a Support Vector Machine (SVM) with epsilon insensitive loss, and a fixed ϵ of 1.0. Sequential Minimal Optimization (SMO) is used as the training algorithm.

[1] `http://www-speech.sri.com/projects/srilm/`

[2] `https://sites.google.com/site/nikobrummer/focalmulticlass`

The optimal complexity was set to 10^{-2}, based on the development set. The SVM implementation in WEKA 3 [26] was used for this purpose. See [1] for a complete description.

Table 1 shows the baseline performance in terms of Accuracy, Unweighted Average Recall (UAR) and Recall for each of the languages. Table 2 also provides the confusion matrix across languages for comparison purposes.

Table 1: Accuracy, UAR, and per language Recall of the baseline and our best submitted system over the development set.

Metric	Baseline	L^2F submission
Accuracy	45%	84%
UAR	45%	84%
Recall (ARA)	33%	86%
Recall (CHI)	45%	94%
Recall (FRE)	36%	83%
Recall (GER)	64%	91%
Recall (HIN)	56%	77%
Recall (ITA)	48%	87%
Recall (JPN)	42%	82%
Recall (KOR)	35%	81%
Recall (SPA)	32%	75%
Recall (TEL)	51%	75%
Recall (TUR)	48%	89%

Table 2: Baseline Confusion Matrix over the development set (rows: reference; columns: hypothesis).

	ARA	CHI	FRE	GER	HIN	ITA	JPN	KOR	SPA	TEL	TUR
ARA	**29**	3	5	7	5	5	6	6	7	6	7
CHI	4	**38**	5	4	5	2	5	10	6	4	1
FRE	11	7	**29**	8	0	4	3	1	11	0	6
GER	5	3	5	**55**	1	7	1	2	5	1	0
HIN	4	1	1	0	**47**	2	2	2	2	21	1
ITA	6	2	9	6	6	**46**	0	4	10	1	4
JPN	4	13	4	2	2	1	**36**	11	10	1	1
KOR	4	19	1	2	2	3	14	**32**	5	3	5
SPA	6	11	15	6	2	4	9	9	**32**	1	5
TEL	2	0	2	2	24	2	2	2	2	**43**	2
TUR	6	5	5	5	2	6	7	8	5	0	**46**

6.2. Proposed System

Table 3 presents the results obtained by our phonotactic and i-vector approaches on the challenge's development set. Notice that these results are on the complete development set, that is, the back-end cross-validation strategy described previously was not applied to obtain these results. The first thing to notice is that both approaches were able to surpass the baseline. However, the i-vector approach did so by a much larger margin. In this sense, the fusion of the 4 phonotactic sub-systems obtained 63.3% UAR, which represents an improvement of 18.2 percentage points over the baseline. On the other hand, even the worse individual i-vector approach was able to improve the baseline by a large margin. It is worth noticing that all the individual i-vector sub-systems based on PLLR features outperformed the one based on acoustic features. Moreover, the combination of the 4 PLLR based sub-systems provides a remarkable improvement with respect to the conventional i-vector acoustic subsystem. Nevertheless, the fusion of the 5 i-vector sub-systems resulted in additional performance gains. This combination of PLLR and MFCC i-vector approaches corresponds to the L^2F primary submission to the challenge.

Table 3: UAR [%] and Accuracy [%] results obtained by the phonotactic and i-vector approaches on the development set.

	UAR [%]	Acc [%]
Baseline	45.1	44.9
Phonotactic (BR)	46.4	46.2
Phonotactic (EN)	51.4	51.4
Phonotactic (ES)	50.0	49.8
Phonotactic (PT)	53.1	53.1
Phonotactic (All) **(I)**	63.3	63.2
i-vectors (MFCC) **(II)**	76.2	76.3
i-vectors (BR-PLLR)	76.9	76.9
i-vectors (EN-PLLR)	79.2	79.2
i-vectors (ES-PLLR)	77.6	77.4
i-vectors (PT-PLLR)	80.6	80.5
i-vectors (ALL-PLLR) **(III)**	83.0	82.9
(I) + (II)	78.6	78.7
(II) + (III)	84.6	84.6

Table 4 shows the confusion matrix of the L^2F system submitted to the challenge over the development set. The most confused classes are the same as in the baseline system (Table 2), namely, between Telugu and Hindi, although with a lower frequency. On the other hand, the large confusion shown by the baseline system over Chinese, Japanese, and Korean is not a problem for the proposed system.

Finally, the proposed L^2F system achieves 81.3% UAR in the challenge test set, in contrast to the 47.5% of the baseline system, which is again surpassed by a large margin. The performance drop with respect to the development set results can be partially due to a slight over-fitting of the back-end estimation. Nevertheless, we consider these results very promising.

Table 4: Confusion Matrix of the i-vector system over the development set (rows: reference; columns: hypothesis).

	ARA	CHI	FRE	GER	HIN	ITA	JPN	KOR	SPA	TEL	TUR
ARA	**77**	0	3	1	0	1	1	0	1	0	2
CHI	0	**78**	0	1	0	1	2	0	1	1	0
FRE	3	0	**64**	2	0	2	2	0	5	0	2
GER	2	1	2	**78**	0	0	0	1	0	0	1
HIN	0	0	0	0	**67**	0	0	0	0	16	0
ITA	1	0	5	2	0	**79**	1	1	3	0	2
JPN	1	1	1	0	0	0	**70**	8	4	0	0
KOR	2	4	1	1	0	0	5	**77**	1	0	0
SPA	2	1	2	1	0	5	4	5	**77**	1	2
TEL	0	0	0	0	18	0	0	0	0	**65**	0
TUR	0	1	1	3	1	2	0	2	1	0	**84**

7. Conclusions

This paper explored the use of PLLR-based i-vector systems for native language detection, building on the good results this method achieves for the closely related task of spoken language and dialect recognition. Results on the Native Language (N) Sub-Challenge of the Computational Paralinguistics Challenge (ComParE) 2016 confirm the potential of the approach outperforming the baseline by a large margin. A possible cause for the observed performance differences is the fact that the i-vector approach is able to better leverage the information present in large data-sets. As future work direction, it would be worth investigating approaches to identify the segments in each sentence that provide a better L1 discrimination.

8. References

[1] B. Schuller, S. Steidl, A. Batliner, J. Hirschberg, F. K. Burgoon, A. Baird, A. Elkins, Y. Zhang, E. Coutinho, and K. Evanini, "The interspeech 2016 computational paralinguistics challenge: Deception, sincerity & native language," in *Proceedings of Interspeech*, 2016.

[2] M. Diez, A. Varona, M. Penagarikano, L. J. Rodriguez-Fuentes, and G. Bordel, "On the use of phone log-likelihood ratios as features in spoken language recognition," in *Spoken Language Technology Workshop (SLT), 2012 IEEE*. IEEE, 2012, pp. 274–279.

[3] D. Martınez, O. Plchot, L. Burget, O. Glembek, and P. Matejka, "Language recognition in ivectors space," *Proceedings of Interspeech, Firenze, Italy*, pp. 861–864, 2011.

[4] S.-M. J. Wong and M. Dras, "Exploiting parse structures for native language identification," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2011, pp. 1600–1610.

[5] J. Brooke and G. Hirst, "Measuring interlanguage: Native language identification with l1-influence metrics," in *LREC*, 2012, pp. 779–784.

[6] J. R. Tetreault, D. Blanchard, A. Cahill, and M. Chodorow, "Native tongues, lost and found: Resources and empirical evaluations in native language identification," in *COLING*, 2012, pp. 2585–2602.

[7] S. Malmasi and M. Dras, "Language transfer hypotheses with linear svm weights," in *EMNLP*, 2014, pp. 1385–1390.

[8] R. K. Guntur and R. Krishnan, "Influence of mother tongue on english accent," in *ICON-2014*, 2014.

[9] N. Dehak, P. A. Torres-Carrasquillo, D. A. Reynolds, and R. Dehak, "Language recognition via i-vectors and dimensionality reduction." in *INTERSPEECH*, 2011, pp. 857–860.

[10] M. A. Zissman *et al.*, "Comparison of four approaches to automatic language identification of telephone speech," *IEEE Transactions on Speech and Audio Processing*, vol. 4, no. 1, p. 31, 1996.

[11] H. Li, B. Ma, and C.-H. Lee, "A vector space modeling approach to spoken language identification," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 15, no. 1, pp. 271–284, 2007.

[12] L. J. Rodriguez-Fuentes, M. Penagarikano, A. Varona, M. Diez, G. Bordel, D. Martinez, J. Villalba, A. Miguel, A. Ortega, E. Lleida, A. Abad, O. Koller, I. Trancoso, P. Lopez-Otero, L. Docio-Fernandez, C. Garcia-Mateo, R. Saeidi, M. Soufifar, T. Kinnunen, T. Svendsen, and P. Franti, "Multi-site heterogeneous system fusions for the albayzin 2010 language recognition evaluation," in *IEEE Automatic Speech Recognition and Understanding Workshop*, Waikoloa, HI, USA, Dec. 2011, pp. 377–382.

[13] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 19, no. 4, pp. 788–798, 2011.

[14] M. Diez, A. Varona, M. Penagarikano, L. J. Rodriguez-Fuentes, and G. Bordel, "New insight into the use of phone log-likelihood ratios as features for language recognition." in *INTERSPEECH*, 2014, pp. 1841–1845.

[15] ——, "On the complementarity of phone posterior probabilities for improved speaker recognition," *Signal Processing Letters, IEEE*, vol. 21, no. 6, pp. 649–652, 2014.

[16] ——, "On the projection of pllrs for unbounded feature distributions in spoken language recognition," *Signal Processing Letters, IEEE*, vol. 21, no. 9, pp. 1073–1077, 2014.

[17] P. A. Torres-Carrasquillo, E. Singer, M. A. Kohler, R. J. Greene, D. A. Reynolds, and J. R. Deller Jr, "Approaches to language identification using gaussian mixture models and shifted delta cepstral features." in *INTERSPEECH*, 2002.

[18] M. Diez, A. Varona, M. Penagarikano, L. J. Rodriguez-Fuentes, and G. Bordel, "Optimizing pllr features for spoken language recognition," in *Pattern Recognition (ICPR), 2014 22nd International Conference on*. IEEE, 2014, pp. 779–784.

[19] H. Meinedo, A. Abad, T. Pellegrini, I. Trancoso, and J. Neto, "The l2f broadcast news speech recognition system," *Proc. Fala*, pp. 93–96, 2010.

[20] D. Caseiro and I. Trancoso, "A Specialized On-The-Fly Algorithm for Lexicon and Language Model Composition," *IEEE Transactions on Audio, Speech and Lang. Proc.*, vol. 14, no. 4, Jul. 2005.

[21] P. Kenny, P. Ouellet, N. Dehak, V. Gupta, and P. Dumouchel, "A study of interspeaker variability in speaker verification," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 16, no. 5, pp. 980–988, 2008.

[22] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems," in *Interspeech*, 2011.

[23] E. Ribeiro, J. Ferreira, J. Olcoz, A. Abad, H. Moniz, F. Batista, and I. Trancoso, "Combining multiple approaches to predict the degree of nativeness," in *Interspeech*, 2015.

[24] L. J. Rodrıguez-Fuentes, M. Penagarikano, A. Varona, M. Dıez, G. Bordel, A. Abad, D. Martınez, J. Villalba, A. Ortega, and E. Lleida, "The blz systems for the 2011 nist language recognition evaluation," 2012.

[25] F. Eyben, M. Wöllmer, and B. Schuller, "Opensmile: the munich versatile and fast open-source audio feature extractor," in *Proceedings of the 18th ACM international conference on Multimedia*. ACM, 2010, pp. 1459–1462.

[26] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.