Short phrase spoken language detection using automatically encoded speech features

Mårten Nilsson
marten3@kth.se

Axel Demborg
demborg@kth.se

Patrik Barkman
barkm@kth.se

Abstract

In this project, a language identification system for Swedish, Slovenian, French and English was built and evaluated. The model used was a GMM-HMM for each language trained on both MFCC:s and features extracted from the bottleneck of an unsupervised automatic encoding of shorter speech samples. Both fully connected and convolutional neural networks were used as autoencoders. The results from the study showed that although the system manged to learn some patterns in the training data it performed no better than random chance on testing data. Possible explanations for these results may be a shortage of training data or that more pre-processing to make the samples more comparable is required.

1 Introduction

This study set out to investigate the usefulness of automatically encoded speech features for spoken language detection. Spoken language detection is the process of automatically identifying the language of a speech sample. This is a technology that enables automatic machine translation of speech and for technologies like multilingual speech recognition.

Language detection of written languages using the Latin alphabet has been thoroughly studied and is considered a solved problem [1]. Since words and letter combinations in written language are manifestations of the phonological rules used in spoken language it has been conjectured that spoken language classification can be solved in much the same way. However it has become apparent that spoken language recognition is a much harder problem, this is mostly due to the fact that phoneme recognizers are prone to errors in transcription [1].

Some experiments have also been performed on the human ability for language detection [2]. These experiments show that humans have the ability to classify languages they don't understand on a prelexical level, this further implies that the problem should be solvable for a machine without it actually having to learn to understand the languages in question.

Phonological studies have also shown that the actual phonemes used in languages overlap significantly. However, the way phonemes are ordered and combined are more distinct between languages [3]. The study of sequences of phonemes is called phonotactics and many language identification systems use phonotactic models [4, 5, 6].

Most studies in the field of language recognition have focused on systems requiring data transcribed on a phoneme-based level directly or indirectly [5, 1, 7, 8, 9, 10]. For example, many language identification approaches use separate phoneme recognizers for each language to be distinguished [5, 7, 11]. The phoneme recognizers can then be used to support a phonotactic recognition system. V. Ramasubramanian et. al. [11] used n-gram language models as their phonotactic system based on HMM as phoneme recognizers. They achieved up to 70% accuracy when testing their system on 45 second utterances from 6 languages using maximum likelihood classifiers. Recently Ganapathy et. al. [8] used features from a convolutional neural network phoneme recognizers together with a support vector machine for language identification. They achieved a 88% accuracy when identifying three languages. A deep learning method has been proposed by Jiang et. al. [9, 10] where deep

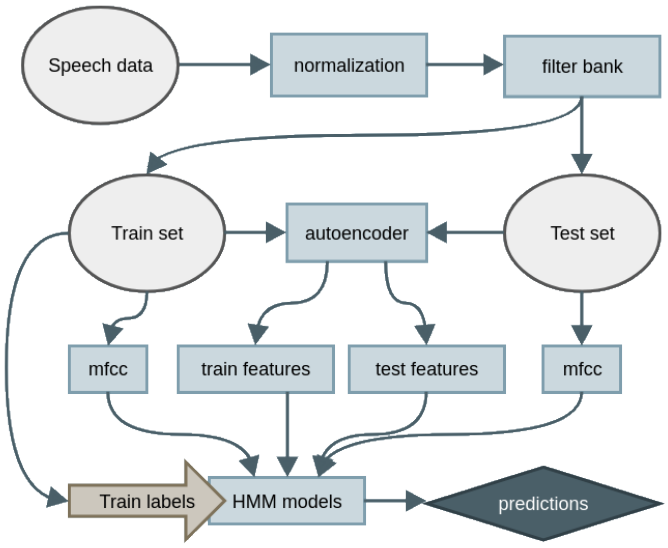


Figure 1: Overview of the complete prediction process. Both the mfcc features and the autoencoder features are present in this illustration, but keep in mind that for each individual prediction only one of these kinds of features were used.

neural networks were trained to generate powerful and robust representations of speech data using stacked Boltzmann machines. They extracted the features from a bottleneck layer in the network where one anticipates that the network has learned a compact representation of the data. Their system was evaluated using a Gaussian Mixture Model–Universal Background Model and achieved an equal error rate of 13.34% on 3 second utterances when distinguishing between 23 languages.

Training systems relying on phoneme recognizers requires labeled data from every language of interest. However, transcribing data on a phoneme-based level is expensive and time consuming. Therefore, a method that does not require phoneme level transcriptions is highly desired. Recently progress in unsupervised feature extraction suggests that neural networks can be used to extract sub word features [12] for speech recognition tasks. However, to our knowledge this has not been applied to language recognition.

This study aims to evaluate a set of phoneme transcription independent systems for short phrase spoken language detection on a subset of the SpeechDat [13] database. Based on the recent breakthroughs for convolutional networks in many fields [14, 8] the use of convolutional autoencoder bottlenecks for feature extraction was evaluated.

2 Method

As previously introduced, most of the language information in spoken language is contained in the transitions between sounds and not in the sounds themselves. To model this temporal aspect, Hidden Markov Models (HMMs) were used. In short, a HMM defines probabilities of feature sequences conditioned on a latent Markov chain. The features in this project are continuous in all cases and have been modeled in the HMM as Gaussian mixtures. HMM models are used to estimate the likelihood $P(X|\Theta)$ for a sequence of features X given the model parameters Θ. For language recognition systems it is common to train different HMM models on the separate languages. In this setup one can interpret the output probabilities from the HMM models as an estimate of the likelihood of the sequence given the language L the model is trained on. The classification can then be performed by either a maximum likelihood (ML) or a maximum a posteriori (MAP) estimation. For a system in production it is sensible to use the MAP estimation to be able to tweak the performance by introducing context dependent priors. However due to the nature of this study, the ML estimation have been used for the classification. This means that the inferred language of an utterance is

$$L_{est} = argmax_L P(X|\Theta_L)$$

where Θ_L is the parameters for the HMM trained on language L.

Historically Gaussian Mixture HMMs have been applied on mel-frequency cepstral coefficients (MFCCs) for the purpose of language recognition [15]. Therefore MFCC features have been used to create a baseline for the evaluation.

Next, language identification with features learned by a neural network directly from spectrogram data were evaluated. This project has focused on an autoencoder architecture for this kind of feature extraction. An autoencoder is a bottlenecked neural network that maps an input onto itself. By setting one of the hidden layers (the bottleneck) in the neural network to be significantly smaller than the dimension of the input data, one enforces the network to learn a compact representation of the data. In other words, training such a neural network is an unsupervised method for finding an alternative representation of the data. Ideally, this representation would provide a better representation for the GMM-HMM models for modelling the separate languages.

Two different autoencoder architectures were used in this project: a shallow fully connected network (FC-AE) and a deep convolutional network (Conv-AE). The input to each network was a sequence of features. More specifically, this sequence corresponded to a window of some specified width applied to a spectogram. The input can thus be thought of a one-channel image. FC-AE flattens this image to a high dimensional vector which is then passed through a number of linear transformations combined with non-linear activation functions. Conv-AE on the other hand attempts to preserve the topological properties of the spectogram. The window used for generating a single input to the network can then be slid along the entire spectogram to map the spectogram to a sequence of features extracted from the bottleneck layer in the network. Note that if one does not take unit steps when sliding this window across the spectogram, the resulting sequence of bottleneck features becomes shorter than the sequence of features in the spectogram. This new sequence was then used as input to a GMM-HMM model. The spectrogram that was used for generating inputs to the network contained log mel filter bank representations of the speech data. The complete prediction process is illustrated in fig. 1.

3 Experiments

For the evaluation, three different experiments were set up. In the first experiment plain MFCC features were used by the GMM-HMM. The features were extracted using the `python_speech_features` package [16] and 13 cepstral coefficients were used. The GMM-HMM implementation was provided by the `hmmlearn` package [17].

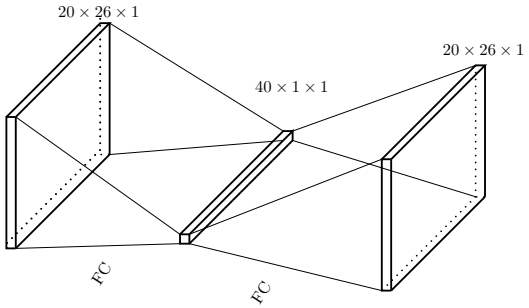
In the next experiments, the aforementioned autoencoder network architectures were used to generate alternative represenations of the data. As mentioned in the method, the input was windowed log mel filter bank features, extracted with the same package as for the MFCC features. The number of filters in the filter bank was 26 and the window size was set to 20 frames (corresponding to 200 ms) which was slid along the spectogram with steps of 10 frames.

In the second experiment a two-layered autoencoder with ReLU activation (FC-AE) was used for feature extraction on windowed log mel filter bank features. The filter bank features were obtained using the same package as for the MFCC features. 26 filters were used in the filter bank. The autoencoder was built with a 40-dimensional bottleneck (fig. 2a).

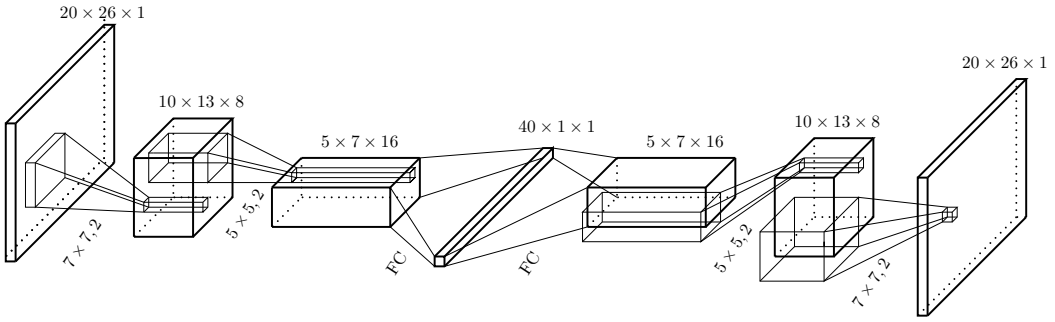
The third and last experiment was performed in an analogous setting as the second experiment but with a deeper autoencoder with convolutional layers (Conv-AE). The network was constructed with two convolutional layers and one fully connected layer to project the input to the 40 dimensional representation. Then one fully connected and two deconvolutional layers were used to up-sample the 40 dimensional bottleneck to the original input dimensions. An illustration of this network is provided in figure 2b. All layers were built with ReLU activation functions in all networks.

All the networks were implemented in TensorFlow [18] and trained with the built in Adam optimizer with the default parameters until convergence.

The experiments were run on a subset of the SpeechDat database containing the four languages Swedish, English, French and Slovenian. The train set was selected as sessions 20-30 from the first block of the first CD in each language and the test set was selected as sessions 31-35 in the same blocks. The languages were selected to represent some of the main language groups present in the database: Germanic, Romance and Slavic languages. This resulted in the training set and test set



(a) FC-AE



(b) Conv-AE

Figure 2: Illustrations of the fully connected autoencoder (FC-AE) and convolutional autoencoder (Conv-AE) architectures used for the second and third experiments respectively.

containing 3.0 hours and 1.4 hours of speech data respectively. The mean sample length was 5.35 s and 5.49 s for each set respectively. Each sample contained some silence which was not removed for these experiments. However, the utterances were preprocessed by normalizing them to a level of -1dB before feature extraction to level out any differences in volume in the recordings.

4 Results

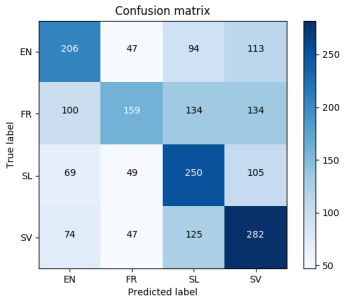
The resulting accuracy on the test and training sets can be seen in table 1 and the confusion matrices between languages are presented in figs. 3 to 5. Interestingly none of the methods perform better than random chance on the testing utterances.

Table 1: Resulting accuracy on test and training set using different the different features for the HMM-GMM

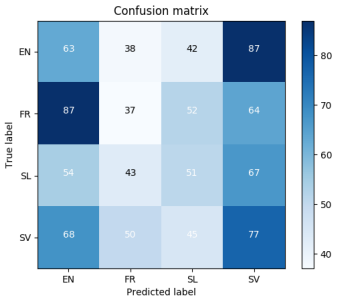
Feature	Accuracy on train data	Accuracy on test data
MFCC	45.12%	24.65%
FC-AE	66.90%	25.84%
Conv-AE	55.53%	24.54%

5 Discussion and conclusions

The results from these test were rather underwhelming, none of the recognizers performed significantly better than random chance on the testing set. However, all of the methods managed to find some sort of pattern in the training data implying one of two possible scenarios. Firstly, the data set could have been too small and the models too powerful allowing them to remember the training sequences without actually learning any generalizable features of the languages. Secondly, the samples in the training and test sets could have been too dissimilar for the model to be able to generalize

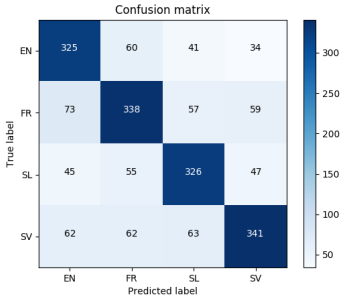


(a) Training data

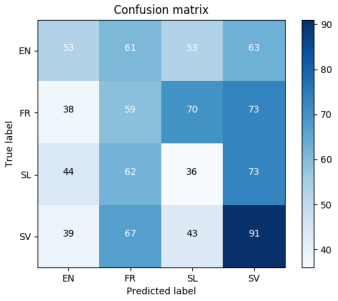


(b) Test data

Figure 3: Confusion matrix on the training and test set for the GMM-HMM trained on the MFCC features.

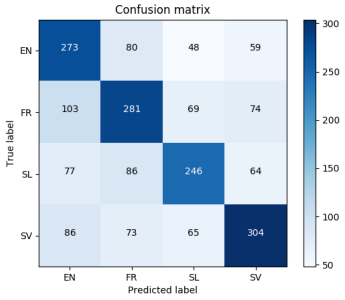


(a) Training data

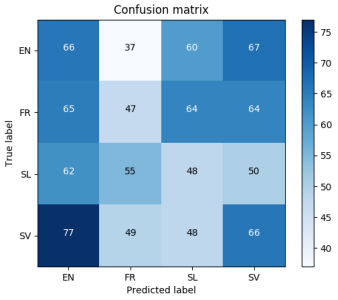


(b) Test data

Figure 4: Confusion matrix on the training and test set for the GMM-HMM trained on the features obtained by the fully connected autoencoder (FC-AE).



(a) Training data



(b) Test data

Figure 5: Confusion matrix on the training and test set for the GMM-HMM trained on the features obtained by the convolutional autoencoder (Conv-AE).

between them. The second alternative seems to be the most likely in this case since the GMM-HMM trained has quite few parameters and especially since the training and testing utterances are sampled from different sessions in the data set between which recording circumstances can have changed significantly. It might be tempting to sample both training and testing data from the same sessions but this risks putting different recordings from the same speaker in the both sets and thus reducing the problem to speaker identification.

One way of making the model better at generalizing would be to use more data for training. This was tested in the study but the library used for HMM broke down when the data set was increased. It is possible though that other more mature tool-kits like HTK [19] would manage this better. An alternative approach would be to use another more data capable model like recurrent neural networks for the temporal modeling.

It is also noteworthy that the Baum Welch algorithm used for HMM training is very sensitive to initialization [20] and that better results might have been achieved if a more clever initialization would have been performed.

Another area of improvement for these models is that since they are never actually trained for discrimination, they could be biased towards extracting features that explain as much of the sounds as possible rather than the languages. This might be remedied by instead using a layer just before output in a classifier for feature extraction. Such an supervised approach may give more distinguishing features, but might be practically impossible to train since there is so little language information in the short samples it is given to learn from.

A possible problem with the autoencoders is that they might have overfitted to the training data with no ability to encode a useful representation for new data. Since no validation data was tested on the autonencoders during training this is hard to rule out even though the quite modest result even on training data indicate that this should not be the case. In addition, some overfitting could be attributed to the GMM-HMMs.

A possible contributing cause for the underwhelming results is the small length of the test sequences. As previously mentioned the average test sequence length was 5.49 seconds. Considering also that a large portion of this time is silence and that the recognition accuracy generally increases with increased sequence length [15] one can expect that the results would improve if the system was deployed on longer spoken sequences. However, due to the fact that the results for all HMMs are comparable to results obtained by pure chance the effect would have to be stronger than the pure averaging effect that comes with the increased sequence length. This could be achieved for longer sequences by capturing long-term dependencies. Classical HMMs as the ones used in this project are not particularly good at this task [21] and thus it is possible that the results would not improve with longer sequences.

Apart from the sequences being short they also contain a lot of silence as previously mentioned. This silence might be problematic since it carries no language information but still is modeled by the HMM:s, if the amount of silence differs significantly between samples it may then affect the probabilities of samples significantly. This would result in the model finding the language with the most closely matching amount of silence for a sample rather than the most similar language features.

One unanticipated finding was that the features obtained from the fully connected autoencoder were the easiest for the GMM-HMM to overfit yielding the highest training performance. This indicates that these features had the best representational power. Since the convolutional model is more complex it was believed that the features obtained from this model would be the most representative. A plausible reason for why this was not the case is that the arcitecture of the convolutional model might not have been suitable for this problem. Another possible reason for this is that the convolutionall operations might not be suitable for preserving the language information that needs to be encoded in this project.

To conclude, the methods evaluated in this project are not applicable to language identification in its current state, but many possible paths for future improvements have been outlined.

References

[1] Haizhou Li, Bin Ma, and Kong Aik Lee. Spoken language recognition: from fundamentals to practice. *Proceedings of the IEEE*, 101(5):1136–1159, 2013.

[2] Yeshwant K Muthusamy, Neena Jain, and Ronald A Cole. Perceptual benchmarks for automatic language identification. In *Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference on*, volume 1, pages I–333. IEEE, 1994.

[3] Marc A Zissman et al. Comparison of four approaches to automatic language identification of telephone speech. *IEEE Transactions on speech and audio processing*, 4(1):31, 1996.

[4] Wei-Wei Liu, Meng Cai, Hua Yuan, Xiao-Bei Shi, Wei-Qiang Zhang, and Jia Liu. Phonotactic language recognition based on dnn-hmm acoustic model. In *Chinese Spoken Language Processing (ISCSLP), 2014 9th International Symposium on*, pages 153–157. IEEE, 2014.

[5] Pavel Matejka, Petr Schwarz, Jan Cernockỳ, and Pavel Chytil. Phonotactic language identification using high quality phoneme recognition. In *Interspeech*, pages 2237–2240, 2005.

[6] Marc A Zissman and Elliot Singer. Automatic language identification of telephone speech messages using phoneme recognition and n-gram modeling. In *Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference on*, volume 1, pages I–305. IEEE, 1994.

[7] Christine S Chou. *Language Identification through Parallel Phone Recognition*. PhD thesis, Massachusetts Institute of Technology, 1994.

[8] Sriram Ganapathy, Kyu Han, Samuel Thomas, Mohamed Omar, Maarten Van Segbroeck, and Shrikanth S Narayanan. Robust language identification using convolutional neural network features. In *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[9] Bing Jiang, Yan Song, Si Wei, Jun-Hua Liu, Ian Vince McLoughlin, and Li-Rong Dai. Deep bottleneck features for spoken language identification. *PloS one*, 9(7):e100795, 2014.

[10] Bing Jiang, Yan Song, Si Wei, Meng-Ge Wang, Ian McLoughlin, and Li-Rong Dai. Performance evaluation of deep bottleneck features for spoken language identification. In *Chinese Spoken Language Processing (ISCSLP), 2014 9th International Symposium on*, pages 143–147. IEEE, 2014.

[11] V Ramasubramanian, AKV Sai Jayram, and TV Sreenivas. Language identification using parallel phone recognition. In *Workshop on Spoken Language Processing*, 2003.

[12] Leonardo Badino, Claudia Canevari, Luciano Fadiga, and Giorgio Metta. An auto-encoder based approach to unsupervised learning of subword units. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 7634–7638. IEEE, 2014.

[13] H Hoge, Herbert S Tropf, Richard Winski, Henk van den Heuvel, Reinhold Haeb-Umbach, and Khalid Choukri. European speech databases for telephone applications. In *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, volume 3, pages 1771–1774. IEEE, 1997.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[15] Marc A Zissman. Automatic language identification using gaussian mixture and hidden markov models. In *Acoustics, Speech, and Signal Processing, 1993. ICASSP-93., 1993 IEEE International Conference on*, volume 2, pages 399–402. IEEE, 1993.

[16] James Lyons. `python_speech_features`, 2013. `http://python-speech-features.readthedocs.io/` [Accessed: 2017-05-17].

[17] `hmmlearn`, 2016. `http://hmmlearn.readthedocs.io` [Accessed: 2017-05-17].

[18] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[19] Steve J Young and Sj Young. *The HTK hidden Markov model toolkit: Design and philosophy*. University of Cambridge, Department of Engineering, 1993.

[20] Pauline Larue, Pierre Jallon, and Bertrand Rivet. Modified k-mean clustering method of hmm states for initialization of baum-welch training algorithm. In *Signal Processing Conference, 2011 19th European*, pages 951–955. IEEE, 2011.

[21] J Callut and P Dupont. Learning hidden markov models to fit long-term dependencies. *UCL-INGI Research Report RR 2005-09*, 2005.