English Accent Classification using Convolutional Neural Networks & LSTMs

Viktor G. Holmér
School of Electrical Engineering & Computer Science
Royal Institute of Technology KTH
Stockholm, Sweden
vholmer@kth.se

Velisarios Miloulis
School of Electrical Engineering & Computer Science
Royal Institute of Technology KTH
Stockholm, Sweden
miloulis@kth.se

Abstract

In this project we implemented an LSTM-based CNN architecture for english accent classification on the TIMIT dataset. By reducing the problem to an image classification one, we hoped to show that it is possible to train CNNs for this task with temporality learned by the LSTM layers. However, due to several factors including lack of data the models failed to properly generalize and instead only overfitted to training data, despite regularization efforts. The models managed to achieve a classification accuracy of $\approx 17\%$ on test data, which is slightly better than random guesses but doesn't show that the architecture generalized.

We employed gaussian noise data augmentation, which is commonly performed to increase robustness to noisy test signals. Ko et al suggested augmenting the data by varying the speed of the signals in 2017. With such augmentation, a larger dataset or more data, we believe that the architecture has potential.

Introduction

Accent classification is the problem of deciding to which accent a sentence, word or utterance belongs. In speech recognition, the problem of accent classification has been thoroughly explored. However, much work remains in accent classification, especially as speech recognition becomes more prevalent in society. Consider for example the prospect of using recorded audio surveillance to deduce the accent of a criminal down to his town of birth. In this paper we approached the problem by extracting mel spectrogram (henceforth referred to as MSPEC) features from the TIMIT dataset and reducing the task to an image classification one.

The TIMIT dataset[1] is a corpus of 630 speakers each reading ten different English sentences. These 630 speakers are each from a total of 8 different dialectal regions. The dataset was hand verified and contains 4158 training sentences and 2142 testing sentences.

MSPECs are commonly used features in speech recognition. For example Shon et al. successfully used MSPEC features to train a dialect identification system for spoken arabic[3]. MSPECs are created from raw audio samples by moving to the frequency domain through Fourier transforms, along with applying filters and smoothing windows to emulate human hearing[2]. These spectrograms show energies at different frequencies and times. An example of an MSPEC can be seen in figure 1. A

Submitted to course DT2112 at KTH Royal Institute of Technology, Stockholm, Spring Semester 2018.

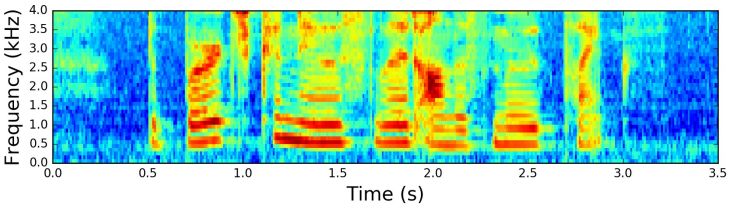


Figure 1: An example MSPEC, taken from [2].

well established machine learning architecture that deals with these types of features are *convolutional neural networks* (henceforth referred to as CNNs).

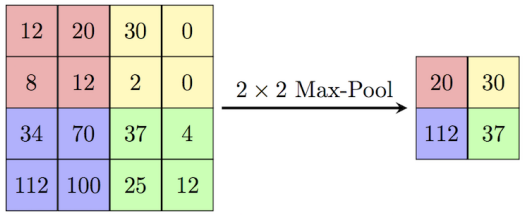


Figure 2: An example of 2x2 max pooling. Taken from computersciencewiki.org.

A convolutional neural network, or CNN, is a type of network designed to learn from image data that was first made popular after its success in image classification[4]. A CNN has several feature maps. Each of these are learned through the training process by sliding a kernel (one per feature map) of fixed size across the image, multiplying its weights with the pixel values in the image. Through backpropagation features are then learned into each feature map. These are then usually down sampled through *max pooling*. Max pooling is a process in which another small window strides across the feature maps and down samples them by simply for each window region outputting the maximum value in the feature map. See figure 2 for an example. Usually pooling is done by a 2x2-window stepping 2 pixels at a time (just as in figure 2). CNNs have proven to be powerful many times over, but unfortunately they are not capable of retaining "memory" and thus cannot learn temporal relationships between inputs. Enter LSTMs.

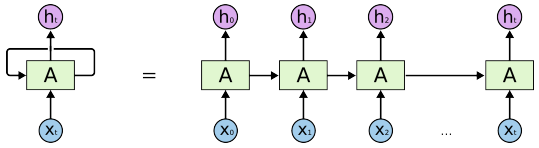


Figure 3: A basic RNN structure, unrolled. Taken from [5].

Long Short-Term Memory or LSTMs[5] are a type of recurrent neural network (RNN). They are capable of "remembering" previous values over time, hence the "memory" part of the name. As any RNN, they are specially designed for dealing with temporal features e.g. time series. This makes them particularly suitable for speech recognition tasks. An LSTM works in principle as any RNN, with some added steps. They train a set of weights that are passed, along with each LSTM unit's output, to the following LSTM nodes. By "following nodes" one can imagine the output is passed back into the input node. Unrolled however, we can view this as several separate nodes[5]. In figure 3 we can see what this looks like for a basic RNN.

In this paper, we combine all of the above concepts into one architecture. Images are processed in a convolutional layer (or several) and are then passed to one or more LSTM layers for temporal consideration. This approach has been successfully used previously[6].

Related Work

The problem of accent classification has been well studied. Shon et al successfully implemented models to accurately classify arabic dialects[3]. They successfully used CNNs to achieve a classification accuracy of 73.39%. However, there have been much earlier works in this area. Arslan & Hansen showed in 1996 that accent classification is not only possible but could out-perform human listeners[7]. They trained four HMMs on four foreign language accents in English and showed that utterance length correlates nicely with accuracy. The longer the utterance, the better the accuracy i.e. more data is better.

The idea of essentially reducing audio classification problems into image classification ones came about when Sainath et al proposed the CNN LSTM architecture in 2015[6]. In this paper, they introduced the idea of combining the automatic feature extraction capabilities of CNNs combined with temporal qualities of LSTM networks. Doing this, and applying the model to a variety of large (200-2000 hours) speech recognition datasets, they found that this architecture was an improvement on regular LSTM networks which was the previous standard in the field[6].

Training above mentioned models can be a daunting task. Occasionally, one might be dealing with datasets that are quite small and thus difficult to learn from. A way to expand these datasets is to *augment* them i.e. artificially create new data from existing sets. In speech recognition, the standard way of doing this is to alter speed in the audio signals, as laid out by Ko et al[8]. In their paper, they mention how adding noise to training signals can improve model robustness against noise in real data. Furthermore they suggest the aforementioned speed perturbation as an improved way of augmenting speech corpora.

Method

Previous work on accent recognition has made use of word level audio sample recognition. This process involves segmenting an audio signal into words, but the boundaries between words can be very hard to identify, especially in an unsupervised manner. Besides, interspeech pauses may hold important information for the overall classification task [9]. In our approach, we attempt to recognize the accent of the whole speaker utterance, so we can avoid the complexity and information loss overhead that comes with word level accent recognition.

The audio files were presented to us in a hierarchical order, broken down by accent and by speaker in each accent collection. The files were in the Waveform Audio File format `(wav)`, so we used the Python wrapper of `libsndfile`, `pysndfile`, and its `sndio` module to read each file. That way we were able to extract the samples of each audio file, along with its sampling rate. We produced the amplitude mel filterbank with 64 filters of each sample using the `librosa`[10] library. Each filterbank energy was converted to log space, to accommodate for the non-linear sound perception in human hearing. Fig. 4 shows the non-scaled and scaled amplitude spectrograms. Each spectrogram has 64 filters, but each recording has different length, so the produced frames are of unequal size.

The scaled mel spectrograms define the basic input for our models, but we experimented with further preprocessing of the utterances in terms of noise addition and standardization.

Standardization Normalization of the features took the form of removing the mean and forcing unit variance. This could be done on at least two different levels, over the whole dataset or over the same speaker. Normalization over the dataset consists of removing the full dataset mean from each sample and dividing by the full dataset variance. On the other hand, normalization over each individual speaker refers to the same process, but with the mean and variance calculated for all the samples of a specific speaker. We experimented with both of those approaches.

Noise addition Noise addition in speech applications can be used for data augmentation purposes. In our case, this took the form of added noise drawn from a standard Normal to each sample. The noisy sample was labelled with the same accent as the original sample, thus doubling our training set. There are more involved ways to perform data augmentation for speech applications [8], such as speed warping of the original audio signal, or Vocal Tract Length Perturbation [11], but we were not able to experiment with those.

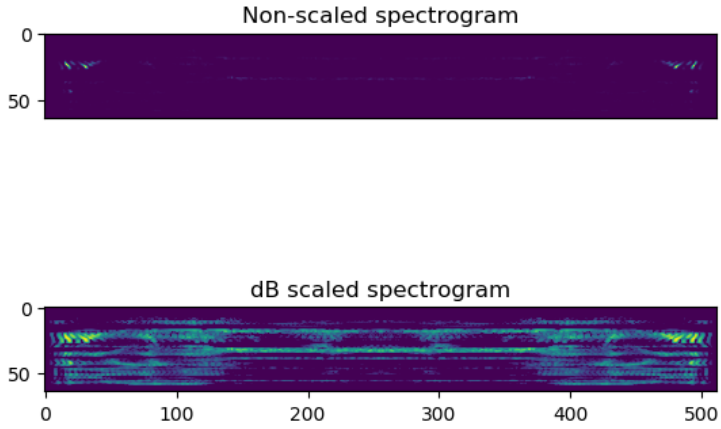


Figure 4: The filterbank energies spectrogram, without (top) and with(bottom) logarithmic scaling.

Experiments In order to evaluate the proposed CNN LSTM architecture we created two models. Model A (in figure 5a), which is shallow, and model B (in figure 5b), which is deep. We trained both of these with the same hyperparameters for all of the data with added noise, and then evaluated the best model by varying the amounts of regularization to further optimize the generalization. Note that our models differ from Sainath et al's in that we do not have any "skip connections" which they employed to feed data around layers[6].

The models have a fairly straightforward construction. Both of them begin with a few convolutional layers with appropriate 2x2 max pooling and batch normalization. Then, the input is reshaped into the proper shapes for the LSTM layers and fed into these. Finally, a few dense layers are applied. The main differences between the models are that model A only utilizes one convolutional layer and one LSTM layer, whereas model B has four convolutional layers and two LSTM layers. The structure of model A was chosen to be as shallow as possible while still retaining the CNN-LSTM structure, whereas model B was constructed to be of a size more consistent with models in literature. For example, Sainath et al[6] used a similar structure with two convolutional layers and two very big (800+ nodes) LSTM layers.

Results

We ran both models without scaling, with dataset level scaling and with speaker level scaling. Finally, we tried to tune the best model (the one with lowest test loss) by varying the degree of regularization, but this yielded no new results. All models were trained with our gaussian noise augmentation. We employed early stopping with a patience of 15 epochs. We used a learning rate of $\eta = 0.001$ and a batch size of $b = 64$. As can be seen in figure 5, we used 32 filters in the convolutional layers. The kernel was of size 3x3 and we employed a stride of 1. The results of all trained models with and without scaling can be seen in figures 6, 7, 8, 9, 10 and 11.

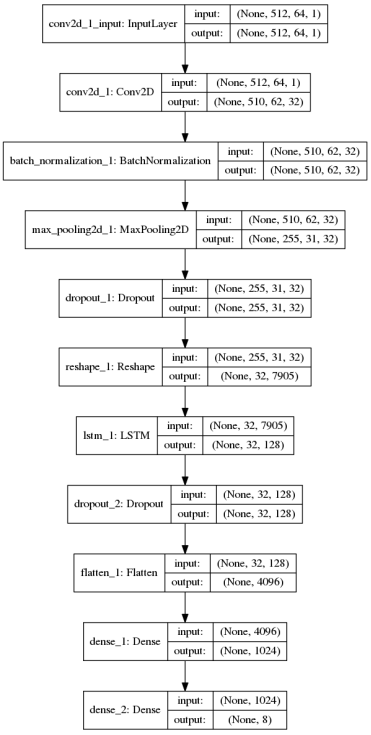
Discussion

Our results, especially the loss graphs for the train and validation sets, show that our architectures overfit to the training data. Despite regularization efforts, our models were not able to generalize properly. We speculate that this is due to several reasons. The first is lack of data, which could effectively be solved with proper data augmentation (note that we tried augmenting the data by drawing noise from a standard gaussian distribution, however this did not work and is not the standard in speech recognition as we came to realize). The second is trying to classify accents on a sentence level. In most related literature, this approach is usually only taken if an abundance of data are available due to its notorious difficulty in achieving generalization.

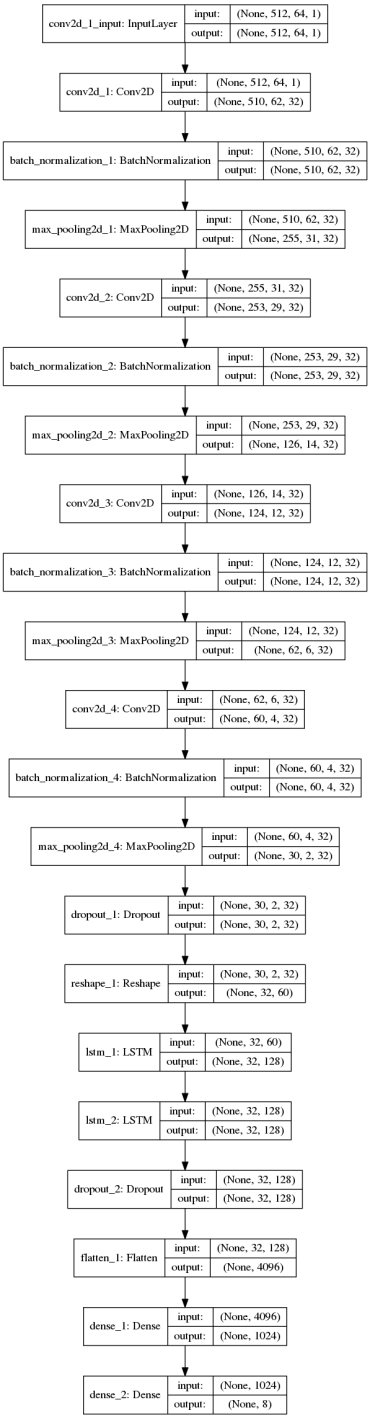
The loss graphs (figures 6, 7, 8, 9, 10 and 11) are text book examples of what overfitting looks like. The regularization rates applied (be it dropout or L_2-regularization), however, show that no matter the

regularization our models always seem to overfit. This points to the suggestion previously mentioned; more data. Augmentation is needed in order to yield properly generalizable models. The TIMIT dataset contains about 5 hours of recordings, but literature shows that for a task of this magnitude one would usually need 40+ hours of data. For example, Shon et al used a dataset of more than 50 hours of data to train their deep models for Arabic dialect classification[3]. This suggests that the amount of augmentation we need is considerable. It is not inconceivable that a simpler path is to simply change datasets to a larger one.

Despite our efforts we could not properly train on the TIMIT dataset. However, as shown in previous work, the CNN LSTM approach has potential. It can be applied to similar tasks as shown in the master's thesis by Blomqvist & Lidberg[12]. We believe that with either sufficient data augmentation for TIMIT or a change to a larger dataset we could properly train our models and locate at least a minimum amount of data in recording hours needed to perform the task of accent classification with the methods and models presented here.



(a) The architecture of model A.



(b) The architecture of model B.

Figure 5: Model architectures that the experiments were performed with.

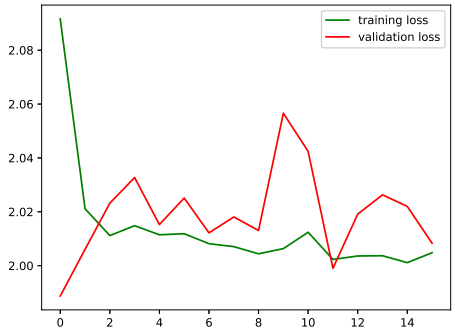


Figure 6: Loss graphs of model A without scaling. Loss: 2.0160. Accuracy: 0.1709.

Figure 7: Loss graphs of model A with dataset scaling. Loss: 2.5174. Accuracy: 0.1051.

Figure 8: Loss graphs of model A with speaker scaling. Loss: 2.5127. Accuracy: 0.1438.

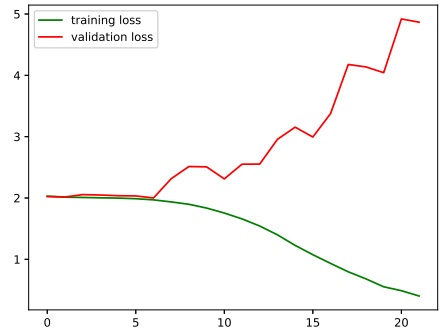


Figure 9: Loss graphs of model B without scaling. Loss: 4.6831. Accuracy: 0.1580.

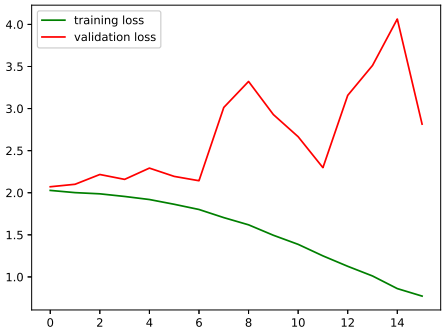


Figure 10: Loss graphs of model B with dataset scaling. Loss: 3.1324. Accuracy: 0.0856.

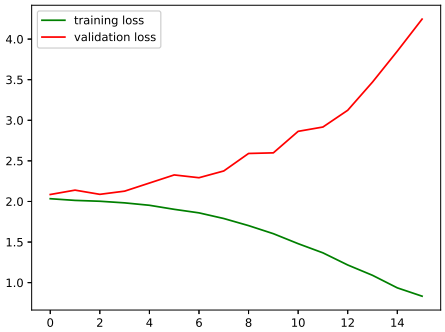


Figure 11: Loss graphs of model B with speaker scaling. Loss: 3.861. Accuracy: 0.1347.

References

[1] John S. Garofolo et al. *TIMIT Acoustic-Phonetic Continuous Speech Corpus*. 1993. URL: https://catalog.ldc.upenn.edu/ldc93s1 (visited on May 21, 2018).

[2] Haytham Fayek. *Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between*. 2016. URL: http://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html (visited on May 21, 2018).

[3] Suwon Shon, Ahmed Ali, and James Glass. "Convolutional Neural Networks and Language Embeddings for End-to-End Dialect Recognition". In: *CoRR* abs/1803.04567 (2018). arXiv: 1803.04567. URL: http://arxiv.org/abs/1803.04567.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*. 2012, pp. 1097–1105. URL: https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf (visited on May 21, 2018).

[5] Christopher Olah. *Understanding LSTM Networks*. 2015. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs/ (visited on May 24, 2018).

[6] Tara N Sainath et al. "Convolutional, long short-term memory, fully connected deep neural networks". In: *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE. 2015, pp. 4580–4584. URL: https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/43455.pdf (visited on May 24, 2018).

[7] Levent M Arslan and John HL Hansen. "Language accent classification in American English". In: *Speech Communication* 18.4 (1996), pp. 353–367. URL: https://www.utdallas.edu/~jxh052100/Publications/JP-15-SpeechComm-Accent-LeventArslan-JohnHansen-OnlyPaper-Jul96.pdf (visited on May 24, 2018).

[8] Tom Ko et al. "Audio augmentation for speech recognition". In: *Sixteenth Annual Conference of the International Speech Communication Association*. 2015. URL: https://www.danielpovey.com/files/2015_interspeech_augmentation.pdf (visited on May 24, 2018).

[9] D. H. Whalen, Charles E. Hoequist, and Sonya M. Sheffert. "The effects of breath sounds on the perception of synthetic speech". In: *The Journal of the Acoustical Society of America* 97.5 (1995), pp. 3147–3153. DOI: 10.1121/1.411875. eprint: https://doi.org/10.1121/1.411875. URL: https://doi.org/10.1121/1.411875.

[10] Brian McFee et al. *LibROSA*. 2018. URL: http://librosa.github.io/librosa/ (visited on May 24, 2018).

[11] Navdeep Jaitly and Geoffrey E. Hinton. *Vocal Tract Length Perturbation (VTLP) improves speech recognition*. 2013.

[12] Viktor Blomqvist and David Lidberg. "Swedish Dialect Classification using Artificial Neural Networks and Guassian Mixture Models". 86. MA thesis. 2017. URL: http://publications.lib.chalmers.se/records/fulltext/251852/251852.pdf (visited on May 15, 2018).

Appendix - Peer Review Fixes

There were a few suggestions presented to us in the peer review. These suggestions are listed here, along with how we remedied or changed the report based on them.

Missing LSTM reference

There was a reference missing when LSTMs were introduced. Mind that it was present in the reference list, it was simply not cited when LSTMs were introduced as a concept in the paper.

Changes made: Added proper citation in the beginning of the section that introduces LSTMs in the paper.

Lack of novelty

Criticisms were raised that we took plenty of inspiration from Shon et als paper on arabic dialect recognition[3]. While quite valid criticisms, we made this choice on purpose as a novel approach to our area of interest, namely CNN-LSTM models, were difficult to procure. Thus we were quite satisfied despite the low novelty score (3/6) in the peer review.

Changes made: None. Novelty was not a concern in this paper.

Clarity of presentation

There were some issues with the clarity of presentation. For example, the use of the term dB scaling was not clear, so we replaced it with an explicit reference to taking the logarithm of the Mel filterbanks. Furthermore the reviewer wanted us to explain in further detail how our standardization was made. These points were taken to heart, and the sections were these were mentioned were expanded to make things clearer. However we did not expand the section on standardization, as "standardization is forcing unit mean and variance" should be sufficient to understand what we did in our opinion. Furthermore the peer wanted more explanations relating to our two models and how they are built. This section (in the method part) was also expanded a bit to introduce in more detail how the models were created and what the differences between model A and B are. Finally, the peer thought that our loss graphs needed units for the axes. This, we chose not to remedy. They are standard loss graphs, we believe it is quite clear that the x-axis is number of epochs and that the y-axis is the loss.

Changes made: Expanded parts concerning dB scaling and information concerning our two tested models.