Different approaches for automatic language identification

Carles Balsells Rodas
carlesbr@kth.se

Matteo Demartis
demartis@kth.se

Flavia García Vázquez
flaviagv@kth.se

Theodoros Panagiotakopoulos
thepan@kth.se

Abstract

Speaker language identification (LID) is a task that has been largely reviewed by the scientific community. In this project, we will go through the most common deep learning approaches to tackle this problem, namely deep neural networks (DNNs), long short term memory recurrent neural networks (LSTM RNNs) and convolutional neural networks (CNNs). Shallow versions of these methods and also a more complex model, as its Resnet-34, were implemented. Our main goal was not only to compare these models but to also identify which features are most suited for each of the architectures considered. The features extracted were Perceptual Linear Prediction (PLP), Mel-Frequency Cepstral Coefficients (MFCC) and log-Mel spectrograms. The experiments show that in general the networks with PLP input were performing better than those with MFCC, except for the CNN that the best performance was given by inputing log-Mel spectrograms. In addition, the best model is the Resnet-34 with log-mel spectrogram as input, obtaining an average F1-score of 92.8%.

1 Introduction

The problem of automatic language identification (LID) consists of classifying the language that is being spoken. LID systems can be used to identify automatically which subtitles to be shown in a video. Another example of its usage could be in call centers, to route international calls to an operator who is fluent in that identified language. These systems are already deployed in Google translate when *Detect language* is chosen or in multi-lingual voice-controlled information retrieval systems, such as Amazon Alexa or Google Home.

A plethora of approaches to tackle the LID challenge have been introduced during the years, from small vocabularies of words to HMMs [1]. Fairly new methods utilize deep neural architectures and specifically networks that handle temporal data such as RNNs[2] and LSTMs[3]. In this project, we focused on detecting how different methods and different types of features influenced the system's accuracy. In particular, we will follow the paper introduced by Gonzalez-Dominguez et al. [4] and aim to perform similar comparisons. Furthermore, we will evaluate different kinds of features, such as MFCC[5] and spectrograms. Other types of networks such as a 2-layer CNN[6] and a more complex CNN, namely ResNet-34[7], will be tested in addition to the models presented by the original author. The models will be trained and evaluated using the Mozilla common voice dataset [8], as opposed to the Voice of America dataset that is used in the aforementioned paper, due to licensing restrictions.

2 Related Work

Similar to other speech classification tasks, a classical approach for language classification is extracting language-dependent features like phonemes and i-vectors [9, 10]. Both approaches require significant domain knowledge.

With the rise of Deep Learning, many researches started combining traditional feature extraction techniques with shallow network architectures [4, 11]. This new combination produced better results compared to applying traditional techniques. Over time neural networks gained more and more popularity and even deeper architectures were tested. Nowadays, most of the research on spoken language identification relies on neural networks for meaningful feature extraction and classification.

Different types of deep networks have been tested over audio: Deep Neural Networks [12] (DNNs), Recurrent Neural Networks[2] (RNNs), Convolutional Neural Networks (CNNs) and Convolutional Recurrent Neural Networks (CRNNs). An example of the latter, is the work of Bartz et al. [2] were a CRNN that achieved an accuracy of 91.0% on the Youtube News Dataset[13] was implemented. Another successful implementation is the work of Revay et al. [14], were they translated the audio classification task into image classification. They utilized the state-of-the-art image classifier ResNet-50 for classifying log-Mel spectrograms of audio samples into language classes. The model was able to classify six languages (English, French, Spanish, Russian, Italian and German) with an accuracy of 89.0%.

3 Data and feature extraction

3.1 Mozilla Common Voice Dataset

The Mozilla Common Voice Dataset [8] has been used as a main resource for the project, it is a community-driven open dataset, which includes audio clips of many different languages. Together with the samples, a description is included specifying the spoken sentence, the gender and accent of the speaker, as well as the number of up and down votes that particular sample had. From this dataset the 8 most common classes where downloaded. These are: English (en), German (ger), French (fr), Italian (it), Spanish (sp), Russian (ru), Persian (pr) and Kabyle (kab), making the total speech time $2,676$ hours. The samples are recorded in 48 kHz and have a bit rate of 64 kbps.

Before extracting the features from the audio clips, some pre-processing has to be applied. As mentioned beforehand, the Mozilla common voice dataset is crowdfunded, meaning that anybody can contribute. Thus, there are plenty of muffled and silent samples that need to be removed. Firstly, samples that contained negative votes were discarded. Afterward, samples were trimmed so that silent parts leading and trailing from an audio signal are cut off. Samples that were too silent were also removed and the remaining ones were re-sampled into a smaller rate of 16kHz to reduce resource demands. For each of the 8 languages, we took utterances of 2.5 seconds long up to a maximum of 100 hours of audio data per class. For evaluation and model selection purposes, 4000 audio samples were taken to create test and validation splits of equal size. The rest of the data is used for training the models. Table 1 shows the distribution of the data for each class as well as the split that was used. Notice that the dataset is imbalanced. We will observe in Section 5 how this affects on the results.

Split	en	ger	fr	it	sp	ru	ps	kb	Total
Train	140k	140k	140k	62.5k	124k	61.7k	131.3k	140k	939.5k
Validation	2k	2k	2k	2k	2k	2k	2k	2k	16k
Test	2k	2k	2k	2k	2k	2k	2k	2k	16k

Table 1: Data distribution of the training, validation and test splits that will be used to train and evaluate the models.

3.2 Features extraction

Audio wave-forms need to be processed so as to obtain the actual features which can be fed to the network. In our experiments, we account to train our models using different features as an ablation study to investigate which are the ones that suit best each model. The features considered are Perceptual Linear Prediction (PLP), log-Mel spectrograms and Mel Frequency Cepstral Coefficients (MFCC). One important choice of parameters that has to be taken into account is the window length and slide shift used to take frames of a given sample. Contrary to the usual conventions in speech processing tasks [15], we choose a window size of 128ms and a slide shift of 32ms for the extraction of the different features. We decide to use this configuration to reduce the computational expenses of our experiments and following default configurations of the tool-kits used to process the raw audio samples. The different features are extracted using *librosa* and *sidekit* tool-kits.

3.2.1 Perceptual Linear Prediction (PLP)

Following the approach taken by *Gonzalez-Dominguez et al.* [4], our models will be trained on 39-dimensional PLP coefficients. The Perceptual Linear Prediction coefficients have been applied successfully in many speech processing tasks so as to reduce noise, suppress reverberation and cancel echo. In general, the performance is increased while the computational load is reduced [16].

The calculation of the perceptual linear prediction coefficients can be summarized in three main steps. First, perceptual processing of the raw audio sample is performed. Then, Linear prediction of the resulting signal is performed via Autoregressive Moving Average (ARMA) modeling of the time series. Finally, cepstral conversion is performed so as to reduce numerical errors and sensitivity to frame synchronization presented by the linear prediction coefficients.

If one applies the previous procedure, 13 PLP coefficients are obtained for each frame of an audio sample. The rest of the coefficients are obtained by taking the first-order (Δ) and second-order ($\Delta\Delta$) discrete derivative of the result.

3.2.2 Log-Mel Spectrogram

Computing the Log-Mel Spectrogram involves a series of steps. Firstly, a pre-emphasis filter is applied to balance the frequencies. Afterward, the signal is split into overlapping hamming windows and Fourier transform is applied [5]. By utilizing the Fourier transform the signal can be represented in the frequency space. At this point, the signal has been converted into a matrix, where one axis represents the frame number and the other the frequency. Further on, by utilizing the log-Mel scale to this matrix, the log-Mel spectrogram is retrieved. It has been shown that humans perceive sound frequencies in a nonlinear way [17]. The mel scale aims to adapt the audio signal to human perception. The spectrograms will be exploited by the CNN models similarly to an image.

3.2.3 Mel-frequency cepstral coefficients (MFCC)

The Mel-frequency cepstral coefficient is considered as the most essential feature in speech classification and can be derived easily using the Mel spectrogram calculated previously (Section 3.2.2). The Mel spectrogram output is highly correlated, which might cause problems in some machine learning algorithms, thus the MFCC applies a discrete cosine transform to the spectrogram to create a compressed, less correlated representation of it. For speech recognition tasks, it is common to consider only the first 13 coefficients together with the first (Δ) and second ($\Delta\Delta$) order discrete derivative [5].

4 Methods

In this section, the different methods used will be elucidated, while the experimental results can be found in Section 5.

4.1 Deep Neural Networks (DNNs)

As a first approach, we will try to identify the language by using the feed-forward DNNs introduced in the main paper [4]. The main idea was to tune the number of layers and nodes, but due to time constraints (very long training time), this has not been possible. The proposed architecture in the aforementioned paper is a feed-forward network, with ReLU activations for the hidden layers and a softmax activation for the output layer. The hidden layers are made of 2,560 units while the output layer dimension is equal to the number of classes. The network takes as input the features that represent a frame concatenated with features of the ± 10 left-right adjacent frames. The added features are referred to as dynamic features and are used to show the context of the actual frame, a feature which is highly important when working with such architectures. The output scores are computed at utterance level averaging the log of the softmax output for all frames.

4.2 Long Short Term Memory (LSTM) RNNs

RNNs are types of artificial neural networks designed for temporal data and can deal with variable input sequence length. A key problem these networks have is that long-range interactions are forgotten by the network. A solution for this is LSTM RNNs [3]. The issue of RNN is solved using memory blocks in the recurrent hidden layer. The memory block contains memory cells with self-connections that store the network state along time, an input gate, an output gate and a forget gate. The input gate controls the amount of information flowing into the cell, the output gate controls the cell activation going towards the rest of the network and the forget gate controls how much cell activation should flow through the self-recurrent connection. Figure 1 demonstrates the structure of the LSTM unit.

The LSTM architecture implemented is based on the one proposed by *Gonzalez-Rodriguez et al.*[4]. This is composed of 512 memory cells and it is followed by a fully-connected layer with as many output nodes as the classes. The activation function of these output nodes would be a softmax and as for DNNs (Section 4.1), the output scores at utterance level will be the result of averaging the log of the softmax output for all frames. The input nodes of this network will not include dynamic features. In other words, the input of the network will be the extracted features of a specific frame.

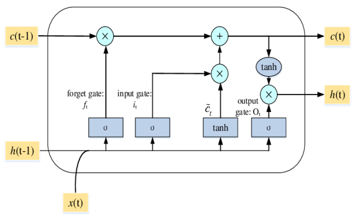

Figure 1: Structure of LSTM unit. Extracted from *Yuan et al.* [18].

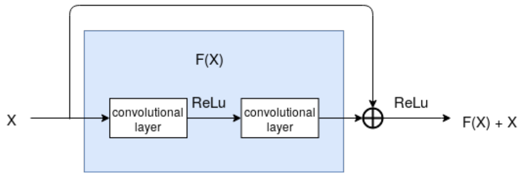

Figure 2: ResNet residual block.

4.3 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks are networks that facilitate signal convolution and are commonly used on images. They have enjoyed tremendous success in many computer vision applications, especially in image classification [6, 19]. Because of this great success, many works emerged incorporating CNN architectures in audio tasks. As mentioned in Section 3.2.2, a spectrogram is a 2D image representation of an audio signal. Therefore, it is possible to use CNNs in the audio signals by having as input the spectrogram of the signal.

CNNs are composed of two blocks: feature learning block and classification block. The feature learning block consists of alternating convolutional layers and pooling layers. The convolution operation performed in the convolutional layers makes the network have as main features locality and translation equivariance (very important characteristic for image classification and even audio-related tasks: you will want that an energy pattern in a spectrogram is recognized regardless of its time or even frequency position). The output of a convolutional layer is called *feature map* and its dimension will be reduced by the pooling layer. Furthermore, the previously mentioned classification block will be composed of fully connected layers.

4.3.1 Two-layer CNN

One of the CNNs implemented for this project was taken from *A Convolutional Neural Network Approach for Acoustic Scene Classification* [20] and it is represented in Figure 3. This network is composed of two convolutional layers of 128 and 256 filters of size 5x5 each followed by ReLu activations. These convolutions are combined with two pooling layers: the first one has non-overlapping kernels of size 5 and the second one will do the pooling over four non-overlapping frequency bands, consequently erasing the time axis. In order to classify the spectrograms, the last

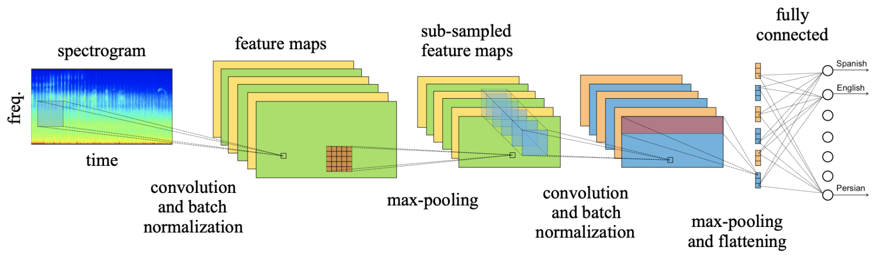

Figure 3: Proposed CNN architecture in *A Convolutional Neural Network Approach for Acoustic Scene Classification*. Extracted from the original paper [20].

step will be to flatten the features and connect them with a fully connected layer of 8 nodes (one for each class) and a softmax activation.

4.3.2 ResNet-34

The ResNet-34 is a deep neural network composed of 34 layers. The innovation behind ResNet-34 architecture is the use of a residual block. The residual block is composed of two connected layers and a skipping connection between the input and the output. The connection skipping is used to avoid the vanishing gradient problem. Figure 2 shows the structure of the residual block.

The ResNet-34 consists of four consecutive main stacks of 3, 4, 6, 3 residual blocks respectively. In each stack the filter size is doubled each time, starting from 64. All the layers inside the stacks use a kernel of size 3x3 and at the beginning of each stack, the input is down-sampled by a factor of two. At the beginning of the network, a convolution using a 7x7 kernel followed by a max-pooling operation with pool size 3x3 is used. Moreover, an average-pooling operation is used right after the last stack.

4.4 Feature combination (MFCC+Log-Mel Spectrogram)

Different works have applied CNNs over MFCC or log-Mel spectrograms. The most popular approach is to apply the network to log-mel spectrogram since MFCC just adds the characteristic of not having correlated variables. This MFCC's characteristic was very important for traditional machine learning algorithms that did not accept correlated variables; for Deep Learning this is not the case [21].

In the work of Suraj Tripathi et al. [22], they proposed to use both, MFCC and log-Mel spectrograms, as input data to perform emotion recognition. They observed an improvement in the accuracy of 2% compared to using one type of audio features.

We try the aforementioned approach to our problem. Therefore, we combined MFCC and log-Mel spectrograms in a two-layer CNN network (Section 4.3.1). For each feature, we concatenate the resulting flattened feature maps and finally we apply two fully connected layers of 200 and 400 nodes each. Again, the number of nodes in the output layer is equal to the number of different languages and softmax activation is used. A sketch of this proposed network is shown in Figure 4.

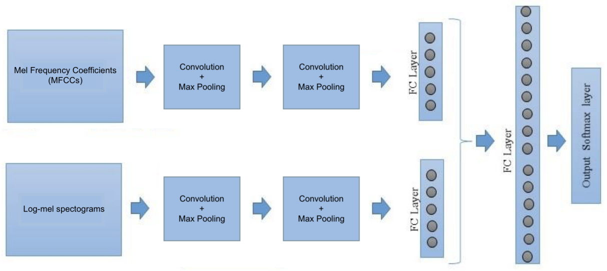
Mel Frequency Coefficients (MFCCs)
Convolution + Max Pooling
Convolution + Max Pooling
FC Layer
Log-mel spectograms
Convolution + Max Pooling
Convolution + Max Pooling
FC Layer
FC Layer
Output Softmax layer

Figure 4: Structure of proposed network for having as inputs MFCC and log-Mel spectrograms. Image taken form. Adapted from *Suraj Tripathi et al.* [22].

5 Experiments

In this section, we present the results of the experiments performed. Our aim is to determine which of the models that we presented in the previous section are able to perform better in the automatic language identification task. As mentioned before, the models will be trained on a different set of features to see which ones are the best suited for each case.

To compare the performance of the different models, we will quantitatively evaluate the predictions of the different architectures and compare the classification F1-score, both per class and overall. Furthermore, we will gather the results reported by *Gonzalez-Rodriguez et al.* [4] and compare them to ours.

The different models implemented for our experiments are:

- **2-/4-layer DNN**: this models will be trained using the PLP coefficients and the MFCC.
- **LSTM RNN**: same as DNN, we train this network using PLP and MFCC.
- **2-layer CNN**: it will be tested over the three different presented sets of features (PLP, MFCC and log-Mel spectrogram)
- **Resnet-34**: we test it using the log-mel spectra.
- **Feature Combined CNN**: it takes both the log-Mel spectrogram and MFCC as inputs.

5.1 Training specifications

All the models have been trained using the training split defined in Section 3 and the objective function selected is the categorical cross-entropy. It is important to mention that no re-balancing techniques have been applied. Therefore, our results may be affected by the biases introduced by the imbalances in the dataset. However, we argue that this is unlikely to happen due to the large amount of data that we provide to the networks.

Due to resource and time constraints, we did not perform hyper-parameter tuning to see which is the optimal training scheme to find the best possible model. Therefore, we have defined an arbitrary optimization schedule and performed *early-stopping* on the validation set to prevent over-fitting.

The networks have been optimized using Adam optimizer [23] with 10^{-3} as learning rate and default beta, for a maximum of 10 epochs. Batch normalization [24] is also applied during the training phase. Future work should be focused on trying different training configurations, and also selecting different values for the window size and shift in the feature extraction process. Considering the LSTM, utilizing the Adam optimizer caused instabilities, thus SGD with a smaller learning rate of 10^{-4} was applied instead.

5.2 Quantitative evaluation

We use the test data split presented in Section 3 to evaluate and compare the different approaches proposed for the automatic language identification task. The metric that will be used to measure the performance of the different models is the F1-score. The F1-score metric is defined as the harmonic mean of the precision (P) and recall (R) [25]. Precision is defined as the number of true positives (T_p) over the sum of true positives and false positives (F_p). Recall is defined as the number of true positives over the sum of true positives and false negatives (F_n).

$$F1 = 2\frac{P \cdot R}{P + R} \qquad P = \frac{T_p}{T_p + F_p} \qquad R = \frac{T_p}{T_p + F_n} \tag{1}$$

All the models make predictions at utterance level except for the DNNs. It has been mentioned previously that the DNNs architectures are fed using the actual frame and some adjacent frames. Therefore, they predict the language of an utterance frame by frame. Table 2 shows the *per-frame* accuracy and *per-utterance* accuracy of the DNNs used in our experiments. In order to compute predictions at utterance level, we follow the approach taken by *Gonzalez-Dominguez et. al.* [4] and average the log of the softmax output of all the frames of an utterance. It can be observed that the *per-utterance* score is higher than the *per-frame* score. This result should not be surprising to us taking into account that the predictions obtained at utterance level are very robust compared to the

Features	System	*per-frame*	*per-utterance*
PLP	2-layer DNN	58.4	76.8
MFCC	2-layer DNN	59.6	78.1
PLP	4-layer DNN	67.7	81.9
MFCC	4-layer DNN	65.1	81.0

Table 2: Comparison between *per-frame* and *per-utterance* average F1-score achieved by the DNN architectures using different features.

ones that involve a single frame and its context. One should also note that we are able to draw very decent predictions at utterance level by using these simple DNN architectures.

Table 3 shows the results of the rest of the models used in our experiments. In order to check for class imbalances in our predictions, we also show the scores achieved for each class separately. In this table, we include all the models trained with various features as described at the beginning of this section.

F1-score (%)										
Features	System	en	ger	fr	it	sp	ru	ps	kb	*Avg*
PLP	2-layer DNN [4]	81.9	73.7	76.1	53.3	68.5	87.2	82.0	88.0	76.8
MFCC	2-layer DNN [4]	80.7	78.8	75.4	61.5	74.1	87.0	76.4	88.2	78.1
PLP	4-layer DNN [4]	89.8	89.3	85.4	31.4	66.6	93.6	90.4	94.9	81.9
MFCC	4-layer DNN [4]	82.9	77.1	81.2	70.4	73.8	88.7	83.9	90.9	81.0
PLP	LSTM RNN [4]	84.2	85.4	85.4	79.2	8.14	91.5	88.9	91.0	85.9
MFCC	LSTM RNN [4]	82.4	82.2	82.4	72.2	77.3	89.6	87.1	90.3	83.0
PLP	CNN [20]	77.3	77.1	71.3	67.6	72.1	80.5	78.3	83.5	75.9
MFCC	CNN [20]	73.3	71.3	69.4	55.0	66.8	79.9	75.6	80.8	72.0
Log-Mel Spect	CNN [20]	69.6	74.5	69.6	68.6	71.9	83.3	80.5	86.5	76.4
Log-Mel Spect	ResNet-34 [7]	92.2	92.5	91.7	90.0	89.9	95.4	94.4	96.1	**92.8**
MFCC+Log-Mel Spect	CNN [22]	81.3	79.1	83.8	81.3	73.6	87.6	89.0	90.2	83.2

Table 3: Performance of different systems with different audio features.

From the results that we obtained, we observe that the model that performed best is the ResNet-34 architecture, which achieves 92.8% score. Considering the two-layer CNN model, we can see that it achieves better results when using log-Mel spectrograms, rather than training it with PLP or MFCC features. One should notice that our proposed feature combination model (of both MFCC and log-Mel spectrograms) is able to boost the performance of the original 2-layer CNN, from 76.4% to 83.2%. Moreover, in most of the cases PLP features give better scores compared to the ones reported by MFCC. This result is interesting, as usually the MFCCs are the most common feature values used when applying machine learning techniques for human hearing related tasks.

If we take a look at the results reported for each class individually (also in Table 3), we should notice that the Kabyle (kab), Persian (ps) and Russian (ru) languages always achieve the highest scores. This observation may happen because these languages do not share any common ancestor, contrary to the rest of the languages. This observation is clearer in figure 5 which shows that Spanish (sp) and Italian (it) were the harder classes to distinguish. In fact, in many cases we observe that the Italian class is the language that gives the lowest scores. This, added to the fact that the data set was imbalanced with respect to the Italian language, might be the reason why this is occurring.

Table 4 shows the Equal Error Rate (EER) of the two best performing models. The achieved results are similar to the ones presented by Gonzalez-Dominguez et al. [4]. More specifically, their best model obtained an average EER of 6.26 while the LSTM an EER of 8.35. We want to stress out that it is hard to compare their results with ours since the models are tested on different datasets as well as different languages.

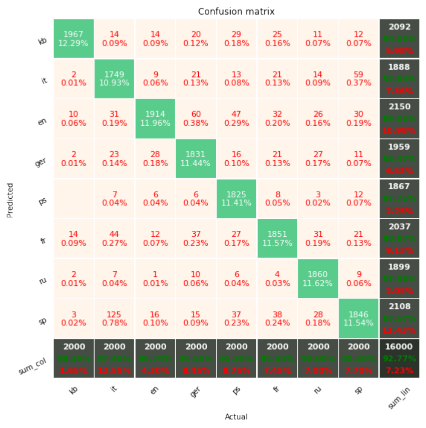

Figure 5: Confusion matrix of ResNet-34 model on the test set.

Features	System	EER_{avg}
PLP	LSTM	5.8
Log-Mel Spect	ResNet-34	3.0

Table 4: Equal Error Rate (EER in %) of the two best models found in Table 3.

6 Conclusions

In this paper, we proposed different methods to address the automatic language identification (LID) task. Taking inspiration from the aforementioned papers we compared different models, such as DNN, CNN, LSTM, ResNet-34 and features combination CNN, with different extracted features (PLP, MFCC and log-Mel spectrograms). The best performance is achieved with the ResNet-34, the most complex network implemented. This one reached an average F1-score of 92.8%. Furthermore, we saw in the experiments that the most suited feature for CNNs was the log-Mel spectrograms, as expected.

As explained in the previous section, the Italian (it) class performance is the lowest in every method. This may have been affected by the unbalance of the data. Future work on this line could be focused on applying class balancing techniques to solve this issue.

Overall, the results obtained from the experiments are very good (the smallest F1-score is 72.0%). Therefore, we can state that the results we got show that we can draw very decent predictions in the automatic language identification task. Nevertheless, we believe that our models may have found shortcuts due to dataset regularities, such as recognizing the voices of each speaker individually rather than the language that is spoken. Therefore, we suggest that more exhaustive evaluation involving tests on different datasets should be carried out. Unfortunately, language identification datasets are limited and hardly any of them is publicly available.

References

[1] *Springer Handbook of Speech Processing*. Springer Handbooks. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[2] Christian Bartz, Tom Herold, Haojin Yang, and Christoph Meinel. Language identification using deep convolutional recurrent neural networks. In *International Conference on Neural Information Processing*, pages 880–889. Springer, 2017.

[3] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[4] Javier Gonzalez-Dominguez, Ignacio Lopez-Moreno, Haşim Sak, Joaquin Gonzalez-Rodriguez, and Pedro J Moreno. Automatic language identification using long short-term memory recurrent neural networks. In *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[5] Haytham M. Fayek. *Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between*, 2016 (accessed May 20, 2020). https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html.

[6] 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.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[8] Mozilla common voice dataset. `voice.mozilla.org`. Accessed: 2020-05-20.

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

[10] David Martinez, Oldřich Plchot, Lukáš Burget, Ondřej Glembek, and Pavel Matějka. Language recognition in ivectors space. In *Twelfth annual conference of the international speech communication association*, 2011.

[11] David Snyder, Daniel Garcia-Romero, Daniel Povey, and Sanjeev Khudanpur. Deep neural network embeddings for text-independent speaker verification. In *Interspeech*, pages 999–1003, 2017.

[12] Valentin Gazeau and Cihan Varol. Automatic spoken language recognition with neural networks. *Int. J. Inf. Technol. Comput. Sci.(IJITCS)*, 10(8):11–17, 2018.

[13] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*, 2016.

[14] Shauna Revay and Matthew Teschke. Multiclass language identification using deep learning on spectral images of audio signals. *arXiv preprint arXiv:1905.04348*, 2019.

[15] Kuldip K Paliwal, James G Lyons, and Kamil K Wójcicki. Preference for 20-40 ms window duration in speech analysis. In *2010 4th International Conference on Signal Processing and Communication Systems*, pages 1–4. IEEE, 2010.

[16] Hynek Hermansky. Perceptual linear predictive (plp) analysis of speech. *The Journal of the Acoustical Society of America*, 87 4:1738–52, 1990.

[17] Leland Roberts. Understanding the mel spectrogram, Mar 2020.

[18] Xiaofeng Yuan, Lin Li, and Yalin Wang. Nonlinear dynamic soft sensor modeling with supervised long short-term memory network. *IEEE Transactions on Industrial Informatics*, PP:1–1, 02 2019.

[19] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[20] Michele Valenti, Stefano Squartini, Aleksandr Diment, Giambattista Parascandolo, and Tuomas Virtanen. A convolutional neural network approach for acoustic scene classification. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 1547–1554. IEEE, 2017.

[21] Li Deng, Jinyu Li, Jui-Ting Huang, Kaisheng Yao, Dong Yu, Frank Seide, Mike Seltzer, Geoff Zweig, Xiaodong He, Jason Williams, Yifan Gong, and Alex Acero. Recent advances in deep learning for speech research at microsoft. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), May 2013.

[22] Suraj Tripathi, Abhay Kumar, Abhiram Ramesh, Chirag Singh, and Promod Yenigalla. Deep learning based emotion recognition system using speech features and transcriptions. *arXiv preprint arXiv:1906.05681*, 2019.

[23] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[24] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[25] Precision-recall. `https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html`. Accessed: 2020-06-01.