Multi-Lingual Audio Classification

Cynthia Hua
cynthiax

Hiroshi Mendoza
hmendoza

December 16, 2018

We attempt to develop an LID system that classifies very short audio segments into their language.

1 Background

Language identification (LID) has the potential to greatly enhance multi-lingual ASR programs. Current ASR systems such as Siri or Google Assistant rely on a user to manually input their spoken language. However, this is less applicable to multi-lingual households, which is the case for 1 out of 5 residents in the U.S., or similar settings [12]. Improved LID programs can form the basis for enhanced ASR systems that might be used assist multi-lingual families, transcribe in international settings or route calls of various languages to correct operators for example.

One challenge for LID systems is that they must be able to recognize any speech segment from a language rather than learning to recognize specific commands like in a speech recognition task. The network must therefore learn to distinguish a language based on the structure of non-specific sounds, rather than by matching audio up to a text transcription. Another challenge for LID is that it needs to be able to classify extremely short utterances in order to be useful, which differentiates this task from other classification tasks such as song recognition [5].

Literature Review LID is a less well-researched task than ASR. Network-based LID approaches often used variations on LSTMs, such as the models from Zazo et al [4] and Gelly et al [2]. Newer LID approaches have also begun utilizing image representations, which is what we attempt in this paper. CNN's were successfully used for language classification by Lozano-Diez et al [3], and have the added benefit of utilizing relatively fewer parameters than other network models. Bartz et al. achieved 98 percent accuracy on 10-second long samples from four languages with a mixed CRNN (convolutional-recurrent) model [1] and Zhao et. al applied a similar RCNN model for speech recognition [17]. We attempt to classify significantly shorter segments (3-second) on a wider set of languages (6-class) with multiple speakers.

Approach We approach the problem using convolutional networks because we expect LID on short sequences to depend on the structure of individual sounds rather than simply time-based patterns. For image-based processing, we start with spectrograms which have shown greater success in prior approaches than other visual representations such as mel-frequency cepstral coefficients[1].

Our initial approach is novel in that our sound segments are particularly brief, only 3-seconds long, and we train on 6-classes compared to 10 seconds and 4-classes in Bartz et. al [1]. Our dataset also comes from actual recorded conversations, so is complicated by naturalistic pauses and inflections, exchanges between multiple speakers and background noise.

We train both traditional CNN's and a Wavenet-based model with dilated causal convolutions. The Wavenet model, originally a generator, has not been previously applied to classification [9]. Code is at https://github.com/hiromendo/cs230

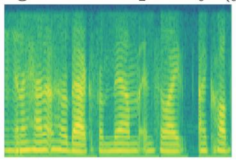
Input-Output The input to the trained neural network will be a 3-second audio sample, and the output will be the predicted language.

2 CNN Models

Data We use the CALLHOME dataset of recorded phone conversations in six languages: English, German, Mandarin, Spanish, Japanese and Arabic, which contains 60 hours of data (in 120 files of 30-minute calls) per language [11].

The data is converted from stereo to mono and split into 3-second segments. We convert the audio wav files into spectrogram arrays 2, which are visualizations of audio frequencies over time. We perform a log-based spectrogram conversion by sliding a window (size = 20 x 20) over the audio file (step size = 10). At each step, a Fourier transformation is used to convert sound signals into sine-based frequencies, and we use a Hanning window which smooths the edges around each window, based on the process described in [10]. In total, we convert 3-second audio segments into [299,161] sized numerical arrays which we pad with zeros into a [300, 300] array.

Figure 1: Spectrogram: Frequency (y) over Time (x)

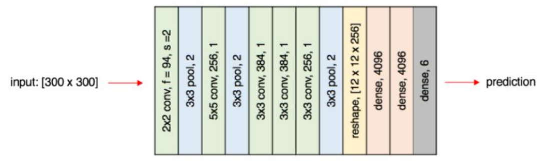


Our data included 25K samples for each of 6 languages. This was split into 105K samples for training, 22.5K for evaluation and finally 22.5K for the test set.

Network Models Our initial networks are CNNs with softmax functions for multi-label prediction, and cross-entropy loss. These models were written from scratch using Tensorflow.

(A) 2-Layer CNN: We train a basic CNN with two sets of convolution and pooling layers. We use kernel size [5x5], stride 2, and valid padding. This simple model, based on the basic CNN used for noise classification by Aykanat et. al [18], allows us a point of comparison for networks with deeper layers. We implement relu activations for all three networks, based on their prior success in image learning.

Figure 2: 7-Layer CNN Diagram



(B) 7-Layer CNN: We train a deeper, 7-layer CNN as detailed in Figure 2. The network has five convolutional layers, with pooling, followed by two fully-connected layers, a structure we based on the AlexNet which was shown to be effective for long-form audio classification [14]. The original AlexNet was designed for [224 x 224 x 3] inputs, compared to our [300 x 300] inputs, so we decrease the size of the initial filters and increase the size of the initial stride; we reduced the number of dense layers and used valid padding throughout.

(C) 16-Layer CNN (VGG-Based): We implement a 16-Layer network with a VGG architecture as detailed in Figure 3, based its success in acoustic scene classification by Zheng et. al [19]. The network has two sets of two convolutional layers followed by pooling layers, then two sets of three convolutional layers followed by pooling layers. We begin with 64 filters and double the number of filters after each pooling layer. Again, we use [3x3] kernels with stride 1 for convolutions and [2x2] kernels for pooling with stride 2. We end with three fully connected layers.

Loss The final layer in all three networks is a softmax layer which is a dense layer with c nodes corresponding to the number of classes that are fed into softmax activation function of the form

$$softmax_i(a) = \frac{\exp a_i}{\sum \exp a_i}.$$

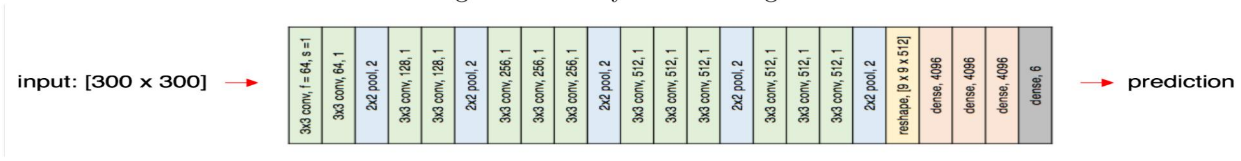
Our loss function is a cross-entropy loss producing a cost of the form

$$Loss = -(1/n)\sum[ylna + (1-y)ln(1-a)]$$

where the summation is over the samples in a mini-batch, y is the true label and a is the predicted label.

Hyperparameter Tuning In our initial experiments, we ran into challenges with over-fitting. Subsequently, we applied dropout to all fully-connected layers, settling on a keep probability of 80 percent after tuning. We also switched from a gradient descent optimizer to an Adam optimizer for the two deeper models because we observed the loss fluctuating up and down during our initial experiments. During hyperparameter tuning, we found that learning rate had the biggest effect on decreasing the loss quickly and found 0.01 to work the best.

Figure 3: 16-Layer CNN Diagram



Training We trained with a Gradient Descent Optimizer with a learning rate of 0.01 for 20000 steps with mini-batch size 100.

3 Wavenet Model

We also implement a Wavenet model in order to improve on the standard convolutions by considering the temporal element of audio.

Data We again use the CALLHOME dataset. We now convert raw 3-second 16kHz audio files into arrays of size 48000 each (16000 slices per second). We then apply a μ-law companding transformation, which aims to improve the signal-to-noise ratio (SNR) by compressing the array value non-linearly to between 0 and 1. This transformation stretches the distance between numbers that are close to zero and squeezes the distance between numbers that are close to 1, with the idea that changes in small sounds are more important than equivalent changes in large sounds for language understanding:

$$f(x_t) = sign(x_t)\frac{ln(1+\mu|x_t|}{ln(1+\mu)}$$

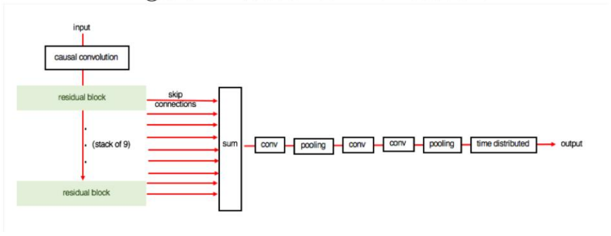
where μ is set to 255 indicating that the audio is encoded into 256 quantization channels.

We used a lower-memory GPU when training the Wavenet, so trained on 4 classes rather than 6. We used 10K samples per class in the training set and 1K per class in the eval and test sets.

Network Model We implement a network model based on Wavenet, but modified to be classifier rather than a generator as shown in Fig. 4.

A stack of residual blocks makes up the core of the network. These blocks contain dilated causal convolutional layers. Causality ensures our network considers the temporal nature of audio data by applying a masking to inputs that ensures that a node can only consider learnings from nodes that evaluated temporally prior windows of inputs. Dilation indicates the convolution is applied with gaps, and skipping inputs in this way allows nodes to consider a larger area of the input (a larger receptive field). In our model, we stack dilated layers with doubling dilation rates in order to improve our network's ability to handle large input arrays.

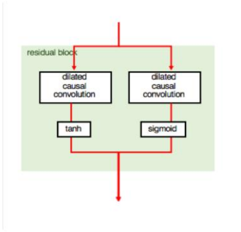
Figure 4: Wavenet Architecture



Inside each residual block are two dilated convolution layers, as depicted in Fig. 5. The convolution with the tanh activation acts a filter, which is responsible for feature learning. The convolution with the sigmoid activation acts as a gate. The outputs from these two activations are multiplied together elementwise to form a final output z. This output z feeds into: (1) the next residual block and (2) a residual connection that passes deeper into the network.

The residual outputs from all the residual blocks are summed together and fed into a relu activation. This result is passed to a series of three convolutional and two pooling layers that reshapes the output in preparation for the final layer. The final layer is a time-distributed layer with a softmax activation. The purpose of using a time-distributed layer is to apply

Figure 5: Wavenet: Residual Block



the same dense function to many outputs across time.

Hyperparameter Tuning Our final architecture implemented 9 residual blocks, which we found to be the best layer number that balanced decreasing loss and memory usage. Our final model had a learning rate set at 0.0005 because we found that the model loss did not decrease when learning rate was any higher. We also lowered Adam parameters beta1 (decay rate for first-moment estimates) to 0.5 and beta2 (decay rate for second-moment estimates) to 0.75.

Training We trained with the Adam Optimizer and mini-batch size 32 for 36 epochs.

4 Results

We trained four models: a basic 2 layer CNN, a 7-layer CNN (based on AlexNet), a 16-layer CNN (based on VGG) and a Wavenet model. The Wavenet model gave us 86% accuracy on 4 classes and the 16-layer CNN gave us 80% accuracy on 6 classes.

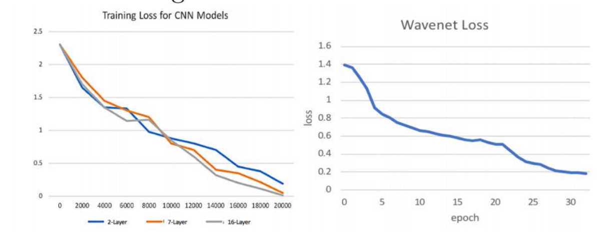
CNN Results Out of the CNN Models, we were able to achieve an accuracy of 80 percent with the 16-layer CNN, 75 percent with the 7-layer CNN, and 53 percent with the basic 2-layer CNN as seen in Figure 6, with losses shown in Figure 7.

We were able to achieve a training accuracy of 99 percent or higher on all our models. The discrepancy between training and test accuracy suggests a variance/overfitting problem. For next steps, we believe that increasing the training data sizes would be most effective, because based on manual analysis, the audio data from phone conversations can vary widely in terms of speaker voice, words being used and ambient noise here we used only 25K samples per speaker. We would also recommend exploring higher rates of dropout or deeper networks.

Figure 6: Model Results

	Training Set Accuracy	Test Set Accuracy
2-Layer CNN	98.5%	53%
7-Layer CNN	99.0%	75%
13-Layer CNN	99.0%	80%
Wavenet	94.8%	86%

Figure 7: Model Loss Plots

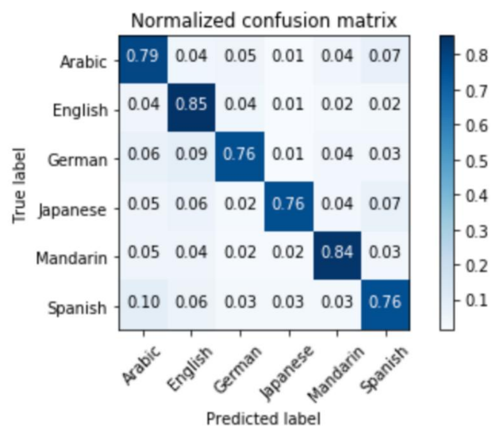


Wavenet Results The Wavenet reached 86% accuracy on four languages. We implemented early stopping here because we noticed that after 36 epochs, the training accuracy continued to increase but the eval set accuracy began to decrease. We also noticed that the loss decline slowed around 20 epochs, after which we performed additional hyperparameter tuning including lower the learning rate to 0.00005 and achieved another 10% increase in accuracy.

Error analysis The confusion matrix for the 16-layer model on the test set is shown in Figure 5. Mandarin and English were the most identifiable languages, where the net achieved 84% accuracy. Notably, Spanish was misclassified as Arabic 10% of the time which may be because Spanish and Arabic share roots and still have a large body of overlapping words today. We also note that German was misclassified as English 9% of the time, which may be because both are Germanic languages and retain similar letter pronunciations even today.

We also performed error analysis on our trained models and found that 2 percent of our data-set could have been discarded because these samples had no sound or had unintelligible speech or ambient noise.

Figure 8: 16-Layer CNN Confusion Matrix



5 Next Steps

Dataset Our audio dataset is challenging because recorded phone conversations contain unpredictable pauses, non-word utterances and multiple, sometimes overlapping speakers. Additionally, our audio sample segments are only 3-seconds long. Based on a hand-classification of these samples, we were only able accurately classify 4 out of 10 samples for languages we were each fluent in. This suggests 3-seconds is often not long enough to allow clear distinctions between languages, particularly when the segments are sliced from natural conversations rather than professionally recorded narration. However, our results suggest a relatively high accuracy rate can be achieved even without cleaner data, so these models could be applied to single-speaker audio for higher efficacy.

Data Model A key issue with image-based audio processing is the large sizes of the data files. This prevents us from training larger datasets and also makes iterating slower due to the time it takes to process and upload data (uploading data to our cloud engine took up to several hours). Due to memory constraints, we reduced the datasets from 'float32' types to 'float16', but we believe that user a higher order data type could allow more accurate learning.

For our CNN's, we only took into account spectrogram features of sound, which only captures tempo and pitch implicitly. Training on chromagrams of the sounds, which explicitly capture harmonic and melodic characteristics of the speech, may have added features that could improve our classification accuracy.

Network Architecture Our findings suggest that deep networks are necessary to learn subtle variations in short audio sequences. Our deeper networks performed better by 30% or more than our basic 2-layer CNN and we saw accuracy increase as we increased the number of layers. For next steps, we would experiment with deeper networks, such as Googlenet which has 22 layers.

Our findings also provide results from a Wavenet classifier, which has not been done before, and suggests the Wavenet is effective as a classifier as well as a generator. The Wavenet classifier performed well on a four-class dataset and had the benefit of requiring slightly less memory for its data inputs (size 48000 samples instead of size 90000 spectrograms). We achieved significant improvements on the wavenet accuracy tuning the learning rate, so expect an even high accuracy could be achieved through additional experimentation with the optimizer parameters.

Contributions Hiroshi worked mainly on finding a data set, data processing, training, tuning, error analysis and the poster. Cynthia worked mainly on data processing, the network models, wavenet training, the diagrams and conclusions.

References

[1] Barts, Herold, Yang Meinel, 2017
`https://arxiv.org/pdf/1708.04811.pdf`

[2] Gelly, Gauvain, Le Messaoudi, 2016
ftp://tlp.limsi.fr/public/IS160180.PDF

[3] Lozano-Dez, Zazo Candil, Gonzlez Domnguez, Toledano, Gonzalez-Rodriguez, 2015
http://cslt.riit.tsinghua.edu.cn/mediawiki/images/a/a9/An_End-to-end_Approach_to_Language_Identification_in_Short_Utterances_using.pdf

[4] Zazo, Lozano-Diez, Gonzalez-Dominguez, Toledano, Gonzalez-Rodriguez: 2015
https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0146917

[5] Taejin Park, Taejin Lee, 2015.
https://arxiv.org/abs/1512.07370

[6] Mohamed, Dahl, Hinton, 2012
https://ieeexplore.ieee.org/document/5704567

[7] Hamooni, Mueen, Neel, 2014
http://www.hosseinhamooni.com/papers/Dual_2014/phoneme_j.pdf

[8] Mareuil, Corredor-Ardoy, Adda-Decker, 2004
https://perso.limsi.fr/madda/publications/PDF/ICPhS99_0726.pdf

[9] Oord, Aaron van den; Dieleman, Sander; Zen, Heiga; Simonyan, Karen; Vinyals, Oriol; Graves, Alex; Kalchbrenner, Nal; Senior, Andrew; Kavukcuoglu, Koray, 2016
https://arxiv.org/pdf/1609.03499.pdf

[10] Muller, Meinard, 2015 https://www.researchgate.net/publication/290440858_The_Fourier_Transform_in_a_Nutshell

[11] Call Home Dataset
https://catalog.ldc.upenn.edu/LDC96S34

[12] Camarota, Steven A.; Zeigler, Karen, 2016 https://cis.org/One-Five-US-Residents-Speaks-Foreign-Language-Home-Record-618-million

[13] Torres-Carrasquillo, Pedro A.; Singer, Elliot; Kohler, Mary A.; Greene, Richard J.; Reynolds, Douglas A.; Deller, John R., 2015
https://www.semanticscholar.org/paper/Approaches-to-language-identification-using-mixture-Torres-Carrasquillo-Singer/31e2fd5e515e4ab855c82af6266fc7e63fd0a1a6

[14] Hershey, Chaudhuri, Ellis, Gemmeke, Jansen, Moore, Plakal, Platt, Saurous, Seybold, Slaney, Weiss, Wilson, 2017
https://arxiv.org/pdf/1609.09430.pdf

[15] Ghahabi, Omid; Bonafonte, Antonio; Hernando; Asuncion Moreno, Javier, 2016
https://pdfs.semanticscholar.org/d937/83c488df0b72806aa1bf7b377ffabae2116d.pdf

[16] Gonzalez-Dominguez, Javier; Lopez-Moreno, Ignacio; Franco-Pedroso, Javier; Ramos, Daniel, 2016
https://ieeexplore.ieee.org/document/5575380

[17] Zhao, 2017
http://www.xlhu.cn/papers/Zhao17.pdf

[18] Murat Aykanat, zkan Kl, Bahar Kurt, Sevgi Saryal, 2017 https://link.springer.com/article/10.1186/s13640-017-0213-2

[19] Weiping Zheng , Zhenyao Mo , Xiaotao Xing, and Gansen Zhao, 2015
https://arxiv.org/pdf/1809.01543.pdf