

---

# Few-Shot Speaker Identification Using Masked Autoencoders and Meta-Learning

---

**John Boccio**

Department of Electrical Engineering  
Stanford University  
jboccio@stanford.edu

## Abstract

Speaker identification refers to the task of identifying which speaker is talking from a given set of speakers. With the rise of deep learning and data driven approaches to audio processing problems, new approaches can be taken on this problem which traditionally was performed by trying to find separability between speakers using hand-engineered feature extractors. The speaker identification problem is a very appropriate problem for meta-learning. Meta learning is a machine learning technique in which a model is trained to learn how to learn, allowing it to adapt to new tasks quickly using relatively little data. By applying meta-learning to the speaker identification problem, a model can be trained to learn how to recognize speakers from a small amount of audio samples from each of the speakers. This could be useful in scenarios where the set of possible speakers is not known in advance, or where the speakers may vary over time.

All of the data that was used for this paper comes from the VoxCeleb dataset [4]. VoxCeleb provides labeled audio data spoken from over 1,000 celebrities. These audio files are then cut into 3 second segments and converted into a spectrogram which will then be fed into a convolutional neural network. The final result is a dataset with over 200,000 spectrograms, each labeled with an id that represents the celebrity that is talking.

The approach that this paper has taken towards applying meta-learning to the speaker identification problem is through the use of a prototypical network (protonet) [5]. The protonet generates an representation of a spectrogram from one speaker which, ideally, is a far distance from the representation of a spectrogram from a different speaker. These representations of each speaker are then used to perform classification. This paper also experiments with the affects of pretraining the network using a masked autoencoder [3]. When applied to speaker identification, a masked auto encoder can be trained to extract characteristics of a speaker's voice from audio samples and use these characteristics to identify the speaker. Combining meta learning with masked auto encoders offers the potential to train a speaker identification model that can adapt to new speakers quickly and accurately using a small amount of training data.

Given 5 examples of 5 different speakers, a model trained from scratch is able to identify who from these 5 speakers is talking with 94% accuracy. When the model is initialized with the weights learned from the masked autoencoder, the best accuracy achieved decreased to 91%.

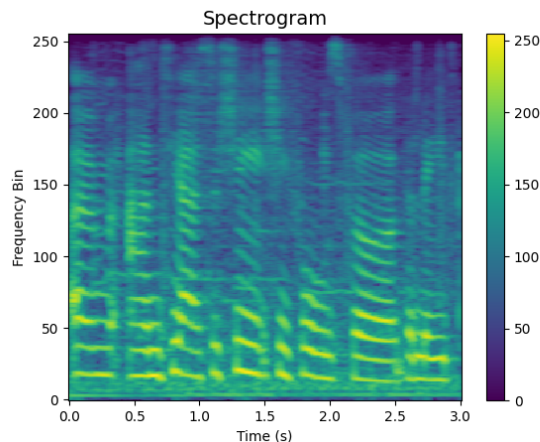


Figure 1: Spectrogram Captured from 3 Seconds of Speech

## 1 Introduction

The problem that this paper is solving is the problem of speaker identification; given a set of speakers, create an algorithm to identify which one of those speakers is speaking in an audio clip. This problem has a wide variety of applications ranging from security, automatic speech recognition, and forensic analysis.

## 2 Related work

There are multiple ways to tackle this problem and how you do so can depend on the data that you have available and the use case. If all of the speakers that need to be identified are known ahead of time and that set of speakers will not change, then a large network could be created to perform the classification. This approach is quite fragile as it depends on knowing all the potential speakers ahead of time, having plenty of data for each, and restricts you to always classifying from your entire speaker set. Another approach is to form i-vectors [2]. An i-vector is a representation of an utterance that is formed by utilizing a statistical model and speech-to-text. Characteristic information about a speaker's voice is contained within these i-vectors which can then be applied to the speaker identification task. This approach works well in some scenarios but still falls short as the i-vector may not be optimal for finding separability between speakers even though it captures the characteristics of a speakers voice.

The shortcomings of classifying all speakers and using i-vectors can be resolved by using meta-learning.

## 3 Dataset and Features

The dataset used for this paper consists entirely of data from VoxCeleb [4]. The VoxCeleb dataset is a collection of over 150,000 utterances from 1,251 celebrities. There is also the VoxCeleb2 [1] dataset, which has even more celebrities and utterances, but VoxCeleb1 provided enough data for the purposes of this paper. The owners and maintainers of VoxCeleb no longer host a download for all of the audio files of the utterances but the links to the youtube videos with labeled timestamps is available. A tool had to be created to scrape the data off of youtube to reconstruct this dataset. The final result is a few thousand audio files, each of which have associated labeled timestamps that specify which celebrity was speaking.

### 3.1 Audio Preprocessing

To preprocess the audio data from the VoxCeleb1 dataset, the raw audio data was converted into a melspectrogram. A spectrogram is a visual representation of the frequency content of a signal over time, and it can be useful for analyzing the characteristics of a speaker’s voice. A melspectrogram, on the other hand, represents the audio data in terms of mel-frequency bins, which are a nonlinear scale of frequency that is closer to the way the human ear perceives sound. One advantage of using a melspectrogram instead of a regular spectrogram is that it can better capture the characteristics of the human voice, such as the pitch and timbre. This is because the mel-frequency scale is more closely aligned with the way the human ear processes sound, so a melspectrogram can better represent the spectral content of speech signals.

Since all of the utterances are of different lengths, a single length had to be decided on. In this paper, the length that was chosen was 3 seconds. If an utterance was greater than 3 seconds, then it was split into multiple 3 second utterances. Any utterance that was less than 3 seconds was thrown out.

The Fast Fourier Transform (FFT) was used to convert the 3 seconds of audio data into the mel-spectrogram. The spectrogram was created with 256 mels, 2048 point FFTs, a hanning windowing function, and a hop length equivalent to 10 milliseconds. All of the audio data was downloaded at a sample rate of 16 kHz, so a 10 millisecond hop length is equivalent to 160 samples. The audio sampling rate being 16 kHz also implies that the maximum frequency of the spectrogram is 8 kHz which is enough to capture most speech sounds.

Once all the utterances are converted into melspectrograms, we arrive at a labeled dataset of over 250,000 images of size 256 by 301. An example spectrogram can be seen in figure 1. This paper uses the same train and test split that is provided by VoxCeleb; 1,211 celebrities are in the training set and 40 celebrities are in the test set. The features in the melspectrograms provide the information needed to create models that can perform speaker identification.

## 4 Methods

There are two methods used in this paper to achieve few-shot learning on the speaker identification problem. The first method is to use protonets trained from scratch on the VoxCeleb dataset. The second method is to perform unsupervised pretraining using a masked autoencoder and then training a protonet for few-shot learning with the weights initialized to those learned by the masked autoencoder.

### 4.1 Prototypical Network

The data being used to identify speakers is a spectrogram of human speech. This makes using a convolutional neural network a very appealing choice. A spectrogram can be treated as an image and then the CNN can be used to find features within the spectrogram to identify the speakers. In order to perform few-shot learning with the prototypical network, the problem needs to be formed in such a way that meta-learning can be applied, as in we want to train the network to learn how to learn to identify speakers from a few given examples.

The convolutional neural network itself can be thought of as an encoder. In this paper, we used a simple 6 layer convlutional neural network which can be seen in table 1. It takes as input a spectrogram and outputs a 1024-dimensional vector. When various spectrograms of a speaker are inputted into this network, the CNN should produce similar encodings. This would imply that the encodings store information about the humans voice such as its timbre and pitch and not information about a particular utterance. The CNN can be driven to create such encodings by utilizing the prototypical loss defined in "Prototypical Networks for Few-shot Learning" [5].

When performing meta-learning, it is common to refer to the number of classes  $N$  and the number of examples of each class given as  $K$ , or N-way K-shot learning. When using the prototypical loss, we take our  $K$  samples of each class, form the encoding using the CNN, and average them to create a prototype for each class  $\mathbf{c}_k$  as seen in equation 1.  $S_k$  refers to all the examples for class  $k$ ,  $\mathbf{x}_i$  is the spectrogram, and  $y_i$  is the speaker label.

$$\mathbf{c}_k = \frac{1}{|S_k|} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_\phi(\mathbf{x}_i) \quad (1)$$

Once the prototypes  $\mathbf{c}_k$  are created, we then classify a new speaker by creating its encoding  $f_\phi(\mathbf{x}_i)$  and finding which prototype  $\mathbf{c}_k$  it is closest to using a distance function  $d$ . This can be done with a softmax function as shown in equation 2. The network can then be optimized by taking negative-log probability of  $p_\phi(y = k|\mathbf{x})$  and minimizing it. The final prototypical loss is shown in equation 3. The number of classes  $N$  and number of samples for each class  $K$  can be adjusted based on the specific problem being solved.

$$p_\phi(y = k|\mathbf{x}) = \frac{\exp(-d(f_\phi(\mathbf{x}), \mathbf{c}_k))}{\sum_{k'} \exp(-d(f_\phi(\mathbf{x}), \mathbf{c}_{k'}))} \quad (2)$$

$$J(\phi) = -\log(p_\phi(y = k|\mathbf{x})) \quad (3)$$

#### 4.2 Unsupervised Pretraining with Masked Autoencoders

### 5 Experiments, Results, & Discussion

### 6 Conclusion and Future Work

### 7 Contributions

John Boccio was the only member of the team and completed all of the work described in this report. The code for this project has been made public and can be found at <https://github.com/John-Boccio/SpeakerIdentificationMetaLearning>.

Layer	Parameters	Activation Function
Input	-	-
Convolutional	16 filters, 3x3 kernel	ReLU
Batch Norm.	16 features	-
Max Pooling	2x2 pool size	-
Convolutional	32 filters, 3x3 kernel	ReLU
Batch Norm.	32 features	-
Max Pooling	2x2 pool size	-
Convolutional	64 filters, 3x3 kernel	ReLU
Batch Norm.	64 features	-
Max Pooling	2x2 pool size	-
Convolutional	64 filters, 3x3 kernel	ReLU
Batch Norm.	64 features	-
Max Pooling	2x2 pool size	-
Convolutional	64 filters, 3x3 kernel	ReLU
Batch Norm.	64 features	-
Max Pooling	2x2 pool size	-
Convolutional	64 filters, 3x3 kernel	ReLU
Batch Norm.	64 features	-
Max Pooling	2x2 pool size	-
Flatten	1024 neurons	-

Table 1: Convolutional neural network used in all experiments

### References

- [1] J. S. Chung, A. Nagrani, and A. Zisserman. Voxceleb2: Deep speaker recognition. In *INTER-SPEECH*, 2018.

- [2] Najim Dehak, Patrick J. Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet. Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4):788–798, 2011.
- [3] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B. Girshick. Masked autoencoders are scalable vision learners. *CoRR*, abs/2111.06377, 2021.
- [4] Arsha Nagrani, Joon Son Chung, and Andrew Zisserman. Voxceleb: a large-scale speaker identification dataset. *CoRR*, abs/1706.08612, 2017.
- [5] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. *CoRR*, abs/1703.05175, 2017.