CS230

Accent Classification and Neural Accent Transfer of English Speech

Lily Chen
Department of Chemistry
Stanford University
chlily@stanford.edu

Laura L. Shen
Department of Electrical Engineering
Stanford University
lashen@stanford.edu

Meng Tang
Department of Energy Resources
Stanford University
mengtang@stanford.edu

Abstract

We aimed to apply neural style transfer to make accented English speech clips sound like native American English. We first trained a convolutional neural network (CNN) on a binary classification task ("native" vs "non-native"), achieving 97.8% accuracy on the test set. The trained CNN was then used in neural style transfer, with a non-native clip as the "content" and a US accent clip as the "style". The CNN classifier was able to classify the generated spectrograms as "native." The generated audio sounds somewhat like human speech, but it was distorted and noisy, and not all words were clearly distinguishable.

1 Introduction

There are approximately 1.5 billion English speakers in the world, of which 75% are English as a Foreign Language (EFL) speakers. [1] Thus, many English speakers, especially those with heavily accented speech, cannot effectively communicate. We set out to build a system to transform accented English speech to sound like native American English. Such a system could enable EFL speakers to communicate more effectively, or even facilitate accent reduction practice.

Neural style transfer [2] is a technique (most often used on images) by which a "content" and a "style" input are combined. Our approach was to apply neural style transfer by using an accented clip as the "content" input and a native accent as the "style" input.

An alternative to accent transformation is to use Automatic Speech Recognition (ASR) to transcribe the accented speech, then use Text-to-Speech (TTS) to resynthesize the transcription, as a form of Speech-to-Speech Machine Translation. This, however, remains challenging due to the heavy accents and faulty grammar/vocabulary of non-native speakers. A successful accent transfer system could improve the quality of ASR by pre-transforming the speech into a native accent.

As an intermediate step, we built a binary classifier to classify speech clips as "native" or "non-native." The inputs were log-amplitude spectrograms of 500 ms speech clips generated from wav files via short-time Fourier transform (STFT). We used a convolutional neural network (CNN) to output predicted labels, either "native" or "non-native."

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LaTeX template borrowed from NIPS 2017.)

The trained CNN from the classifier was then used to carry out the neural style transfer algorithm. The input was the log-amplitude spectrogram of a non-native speech clip combined with Gaussian noise. The output was the result of applying the neural style transfer algorithm on the input, using a native clip as the style input. The generated spectrogram was then converted back into audio via inverse short-time Fourier transform (ISTFT), yielding a wav file.

2 Related work

Verma and Smith have demonstrated a neural style transfer approach on spectrograms of audio and were able to achieve timbral transfer, e.g. from a singing voice to a violin. [3] They trained a modified AlexNet architecture on a classification task (classifying audio based on categories of musical instruments), and used the trained network to optimize a random-noise input signal on a loss function incorporating style and content loss terms. They also used two additional loss terms measuring the deviation from the temporal and frequency energy envelopes of the style input, which they found helped to achieve better quality.

Similarly, Foote et al. attempted to use neural style tranfer to make speech by one speaker sound like speech by another speaker. [4] Using a dataset of music clips labeled by genre, they used various networks such as WaveNet and a VGG-like network to perform neural style transfer, but only obtained results that sounded like noise.

3 Dataset and Features

We used the 18 free publicly released CMU_ARCTIC databases of speech from US English speakers and other accented speakers (e.g. Indian, Scottish), labeled by accent. [5]

Each CMU_ARCTIC datasets consist of approximately 1150 one to five second utterances stored as raw WAV audio files. Since the datasets come with full phonetic labeling that breaks down each utterance by phoneme, we initially extracted the individual phonemes of each of the audio clips. However, we found the individual phonemes did not provide enough meaningful data because they were too short. Thus, we decided instead to break each audio clip into non-overlapping 500 ms segments (audio outside of 500 ms increments were discarded) and performed Short Time Fourier Transform (STFT) on the segments resulting in spectrograms of dimensions (1024, 22), as shown in Figure 1, which were reshaped into (205, 110).

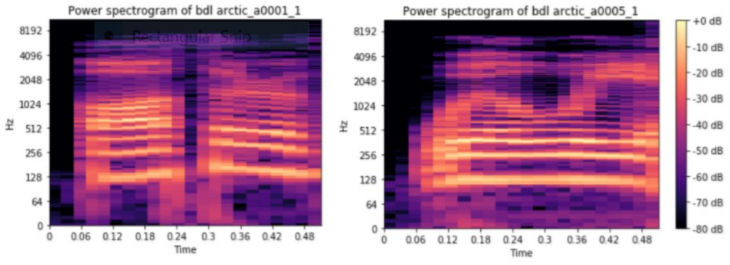

Figure 1: Example plots of speech data spectrograms.

This approach yielded a total of 92k spectrograms, which were divided into training, dev, and test sets. Our goal was to correct Indian male accented speech (non-native) to have US male accent (native). Thus, our dev and test sets consisted of the 3 Indian male and 3 US male data sets. The other 12 datasets, including US and Indian females, Scottish, and Canadian speakers, are included in the training set. We labeled the Canadian examples as "native," because the Canadian and US accents were indistinguishable to the human ear.

We produced additional data using data augmentation (e.g. rescaling audio amplitude, offsetting the clipping window), but did not end up using the augmented data because we did not find variance to be a problem, as our classifier had similar performance on the test set and training set.

4 Methods

4.1 Accent Classification

We used a cross-entropy loss function to minimize the difference between the predicted labels and the actual labels (which were generated from information provided with the dataset). Our training set consisted of 82k examples, and our test and dev sets consisted of 5k examples each.

4.2 Neural Accent Transfer

We applied the neural style transfer algorithm developed by Gatys et al., [2] using the parameters learned during the training of our classifier CNN. The activations of the early layers represent the structure (content) of the spectrograms, while the deeper layers represent the style (accent). To generate a spectrogram G, we minimized: [7]

$$J(G) = \alpha J_{\text{content}}(C, G) + \beta J_{\text{style}}(S, G)$$

where:

$$J_{\text{content}}(C, G) = \frac{1}{4 n_H n_W n_C} \sum_{\text{all entries}} (a^{(C)} - a^{(G)})^2$$

and

$$J_{\text{style}}^{[l]} = \frac{1}{4 n_C^2 (n_H n_W)^2} \sum_{i=1}^{n_C} \sum_{j=1}^{n_c} (G_{ij}^{(S)} - G_{ij}^{(G)})^2,$$

where G_{ij} denotes the (i, j)th entry of the Gram matrix.

An Indian accented clip was used as the content input C, and a native US accented clip was used as the style input S. We initialized G as C with added noise. The generated audio was then reconstructed from the generated spectrogram G by using an inverse short-time Fourier transform (ISTFT).

5 Experiments/Results/Discussion

5.1 Accent Classification

Our initial attempt used a simple CNN with only 2 CONV layers, which gave a test set accuracy of 92.4%. We were encouraged by the high performance of this simple model. We then deepened the CNN to 8 and then 10 layers, because we anticipated needing a deeper neural network for the style transfer step. The architecture of the final 10-layer model is shown in Figure 2.

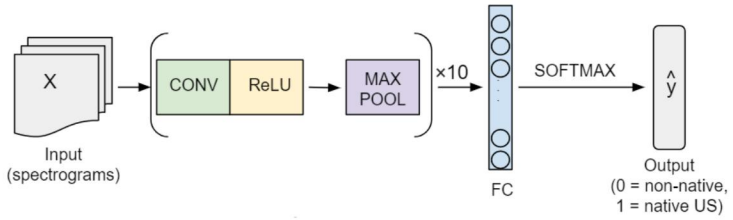

Figure 2: Architecture of CNN classifier.

Our primary metrics for the CNN classifier were accuracy, precision, and recall, shown in Table 1.

Model	Train set accuracy	Test set accuracy	Precision	Recall
2 CONV layers	96.1%	92.4%	90.2%	98.1%
8 CONV layers	97.8%	97.2%	95.9%	99.5%
10 CONV layers	99.2%	97.8%	98.1%	98.3%

Table 1: Accuracy, Precision, and Recall for CNN models.

The confusion matrix on the test set with the 10-layer network is shown in Table 2.

	Predicted:	
True labels:	Non-Native	Native
Non-Native	1945	59
Native	51	2981

Table 2: Confusion matrix on the test set

Classifier performance was very good, consistent with our subjective evaluation that humans can perform nearly perfectly on the classification task. There was little difference between the performance on the training set and the performance on the test set.

5.2 Neural Accent Transfer

Using our trained CNN, initial attempts to perform neural style transfer were unsuccessful when using early layers (e.g. layer 2 or 3) for content and later layers (e.g. layer 6, 7, or 8) for style because the style cost J_{style} was going to infinity. By examining the activations of the later layers, we observed that they were almost entirely zero. Using batchnorm did not alleviate this problem because batchnorm prevents the activations from becoming too large, but our activations were too small.

We changed tactics to take the style from the same earlier layers as the content since for audio distinguishing between content and style is not as easily clear-cut as for images. While this change prevented the style cost from going to infinity, the plotted images of the generated spectrograms were noise with no visible shape or content.

Given the difficulty of making our CNN work with style transfer, we switched to using a pretrained VGG-19 network, generating spectrograms that were successfully classified as "native" by our accent classifier. We tried different weightings of the style and content loss terms, yielding different loss curves (See Figure 3).

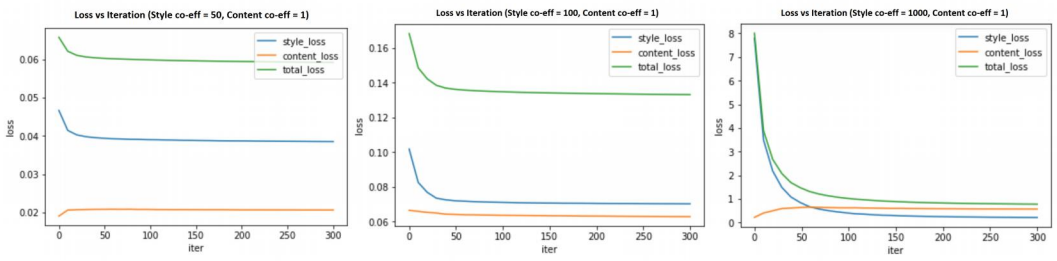

Figure 3: Loss versus number of iterations for total, style, and content using the VGG-19 network. The style to content ratios for plots left to right are 50:1, 100:1, and 1000:1.

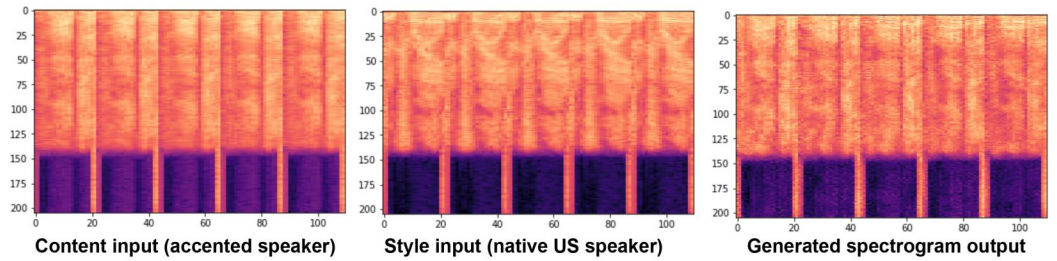

Figure 4: Inputs and generated spectrograms using the VGG-19 network.

The pretrained VGG-19 network takes in an array with values between 0 and 255 and has three channels, so we mapped our spectrogram values to match the expected input values and duplicated the content and style spectrograms across three channels. After feeding the spectrograms through the

VGG-19 network, the spectrogram arrays in each channel had slightly different values, so we took the average of the values and rescaled the values back to the original spectrogram values.

One downside to using the pretrained VGG-19 network for our neural style transfer is that the VGG-19 was trained on the ImageNet database, not spectrograms of audio. However, the earlier layers of the VGG-19 network which we used for our style transfer (layer 2 for content and layers 3-6 for style) should still capture low-level image features such as edges and shapes, which are relevant spectrogram features. Visual inspection of the inputs and outputs shows that the generated spectrogram combines visual features (e.g. edges and contours) of both the content and style input.

The audio was reconstructed by converting the spectrograms back to a time-series array using ISTFT, exporting the time-series arrays to wav files, and concatenating the individual wav clips to produce the original sentence. The generated audio sounds like human speech, but distorted and noisy, and not all words are clearly distinguishable. This could be partially due to the phase information lost when taking the STFT of the audio c ausing the resulting audio to sound tinny and distorted. Several samples of our generated audio have been uploaded to our GitHub in the `generated_audio` directory.

6 Conclusion/Future Work

Classifier performance was very good, consistent with our subjective evaluation that humans can perform nearly perfectly on the classification task. However, neural style transfer was not nearly as straightforward for audio as it is for images. After neural style transfer, the generated spectrograms were classified as "native" by our original classifier, whereas before they were classified as "non-native," indicating that some form of style transfer took place. However, the generated audio sounds garbled with no discernible speech sounds, perhaps partially due to the difficulty of reconstructing audio from spectrograms since phase information was lost.

Our classifier performed nearly as well with significantly fewer layers, so perhaps later layers did not provide additional feature information, thus hampering style transfer. In the future, we could use a more sophisticated CNN architecture, such as AlexNet or VGG-19, and train on more audio spectrograms. We could also try further tuning of the filter size, stride size, etc.

One problem with our approach is that, by treating audio spectrograms like images, we discarded a lot of temporal information. The Gram matrix term we used for the style loss captures only correlations between frequency features, and discards temporal data which is relevant to speech accents. By clipping our audio into short 500 ms segments, we also lose a lot of contextual temporal information that characterizes accents. One possible avenue for improvement in would be to experiment with different loss terms, e.g. to incorporate more temporal information, similar to the approach in [3].

Another optimization might be to use a more sophisticated method of reconstructing audio from the generated spectrogram, e.g. using the Griffin-Lim algorithm to recover the phase information that was lost. This might improve the quality of the generated audio and allow any speech sounds present to be more clearly heard.

Another approach to the problem might be to use a Generative Adversarial Network (GAN) to generate the accent-transferred clips. We would first train a GAN to generate native US English speech clips against a discriminator that classifies the output as US speech or non-US speech (including non-speech sounds). Then, we would freeze the weights of the Discriminator and use accented speech as the input to the Generator. This would have the advantage that the generated clip would be more likely to come from the target distribution (i.e. speech sounds), avoiding the problem we had where the generated clip did not sound like speech.

7 Contributions

- Data preprocessing: Laura, Lily
- Classifier: Meng, Lily
- Style transfer and generated audio reconstruction: Meng, Laura
- AWS setup, efficient large data uploading, generation of plots and result statistics: Meng
- Proposal, milestone, poster, and final report writeups: Laura, Lily

References

[1] Nuno Marques. "How And Why Did English Supplant French As The World's Lingua Franca?" https://www.babbel.com/en/ magazine/how-and-why-did-english-supplant-french-as-the-world-s-lingua-franca

[2] L. A. Gatys, A. S. Ecker, M. Bethge. "A Neural Algorithm of Artistic Style." (2015) https://arxiv.org/abs/1508.06576

[3] P. Verma, J. O. Smith. "Neural Style Transfer for Audio Spectrograms." (2018) https://arxiv.org/abs/1801.01589

[4] D. Foote, D. Yang, M. Rohaninejad. "Do Androids Dream of Electric Beats?" (2016) audiostyletransfer.wordpress.com/

[5] Carnegie Mellon University. "CMU_ARCTIC", http://www.festvox.org/cmu_arctic/

[6] Coursera, Convolutional Neural Networks. "Deep Learning & Art: Neural Style Transfer." www.coursera.org/learn/convolutional-neural-networks

Code

`https://github.com/soloist96/CS230Accent`

Packages used and references to code consulted

- Tensorflow
- PyTorch
- NumPy
- SciPy
- Librosa
- Sample code from jupyter notebooks on Coursera
- PyTorch tutorial on neural style transfer: http://pytorch.org/tutorials/advanced/neural_style_tutorial.html