Language Classification through Audio

Brayden Christensen

April 20, 2024

Abstract

Winter 2024 final project report for CS 474 (Deep Learning) at BYU. The only original part of this project was my idea of treating language detection as an image classification problem by converting the audios to images and using a pre-trained CNN to learn the language classes. Data

1 Data and Approach

Two different datasets were utilized for this task: Spoken Language Identification, and Google's FLEURS. These datasets were chosen due to their diversity in terms of the number of languages, instances, and audio topics covered. While the HuggingFace datasets are more readily available, the Spoken Language dataset required a unique Dataset class to load the data. The data was either provided in waveform format or converted to waveform and then to a spectrogram image representation of the audio. I learned a lot about the torchaudio library and ways to speed up the audio conversion process for faster training. I had a lot of trouble with the FLEURS dataset so the following results are only from Spoken Language. The three classes are English, Spanish and German.

I used a CNN that is a fine-tuned ResNet18 and an AST (Audio Spectrogram Transformer). As I researched and learned about Audio Classification I learned about AST's and ViT's (Vision Transformer) and decided to compare my ResNet to a transformer so I implemented my own AST from [GCG21] that uses parts of a pre-trained ViT to extract embeddings from the audio image that a transformer can encode again with positional embeddings and be softmaxed into classes (Figure 1).

2 Spoken Language Dataset with ResNet18 and AST

I only needed to train for 1 epoch while fine-tuning ResNet18 since I saw quick overfitting of the train data. This dataset has three classes: English, Spanish and German audios. Most of the audios were around 10-12 seconds long. I created two CNN's that had the same parameters and just adjusted the format of the audio spectrograms. In the figure below (1d) the spectrogram doesn't actually look that interesting so I tried a different type of spectrogram and didn't reshape it. This gave me better accuracy both in the train set and the test set.

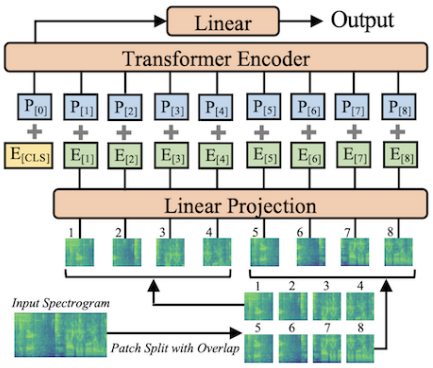
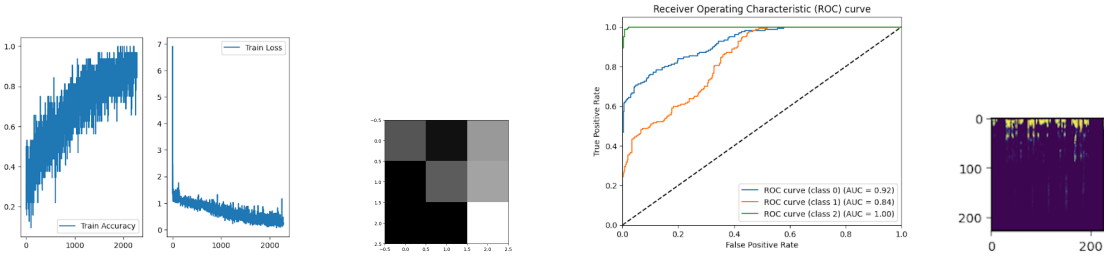


Figure 1: AST architecture

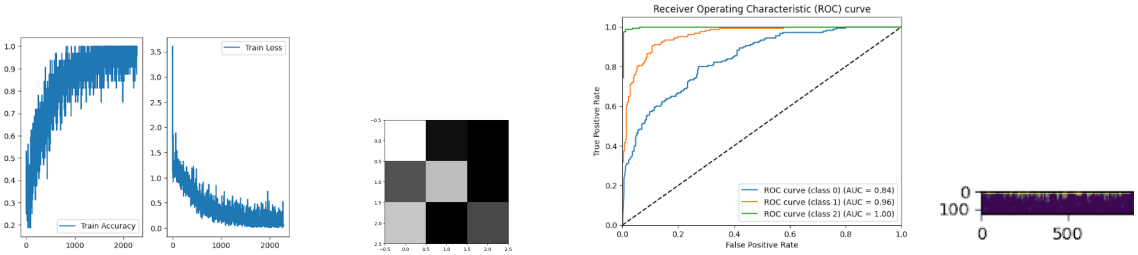
Metric	Train Accuracy	Train Loss	Test Accuracy	Test Loss
Spectrogram	0.8912	0.2842	0.5644	1.5683
MelSpectrogram	**0.9635**	**0.0949**	**0.6373**	2.1767
AST	0.5270	0.9915	0.4851	**1.0284**

Table 1: Results



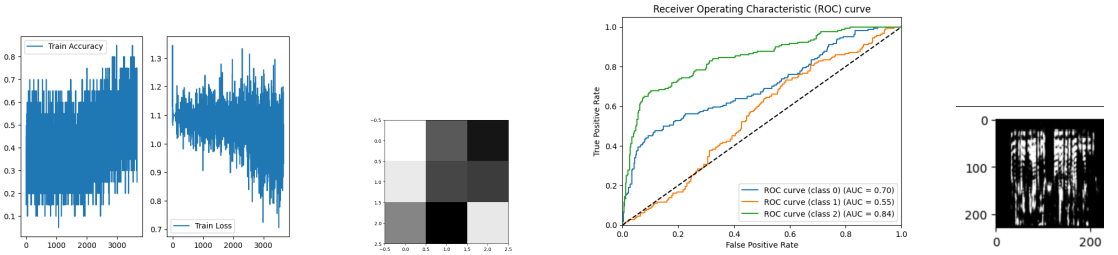
(a) Acc and Loss (b) Confusion Matrix (c) ROC Curve (d) Input Image

Figure 2: CNN Spectrogram Visuals



(a) Acc and Loss (b) Confusion Matrix (c) ROC Curve (d) Input Image

Figure 3: CNN MelSpectrogram Visuals



(a) Acc and Loss (b) Confusion Matrix (c) ROC Curve (d) Input Image

Figure 4: AST Visuals

3 Conclusion

For the dataset that I used it turned out that the best performing model was the ResNet with non-square images. In the future I would like to test these models on larger datasets and modify my AST to be able to take in non-square images, colored images, or just pre-process the waveform directly into an embedding at the start.

References

[GCG21] Yuan Gong, Yu-An Chung, and James R. Glass. AST: audio spectrogram transformer. *CoRR*, abs/2104.01778, 2021.

Date	Time	Summary	Type
2/20	30 min	Found 2 potential datasets for voice recognition with 100+ different languages with men and women speaking	research
3/20	1 hour	Looked at papers and tutorials on how to convert audio to a set of images	research
4/1	3 hours	Tested out downloading my dataset and turning audio files into images. Also learned how to extract the audio labels from the file path.	prep work
4/2	3 hours	Created Dataset class to preprocess the audio files and turn them into images with language labels. Realized I might have to make a new Dataset class if I decide to use some other datasets.	prep work
4/10	2 hours	Learned about modern audio classification models like CTC, Audio Spectrogram Transformers, and Whisper.	research
4/10	2 hours	Built out CNN architecture to take in an image of a certain size and spit out a classification for the type of language.	model
4/12	2 hours	Tested sending 1 batch of images into the CNN and matching output to a class label. Had to adjust code that transforms audio to image.	model
4/12	2 hours	Got the model to train for 1 epoch to test everything. Took just 1% of the train dataset for testing.	model
4/13	2 hours	Optimized audio to image conversion with torchaudio and trained on the entire train set and reported train and test metrics.	model
4/14	1 hour	Created 2nd Dataset class for MINDS14 dataset and tested with CNN	model
4/16	2 hours	Explored torchaudio library for audio to spectrogram functions to get better looking images	model
4/16	1 hour	Debugged training CNN on MINDS14 dataset with adjusted dataset class	model
4/17	1 hour	Read Audio Spectrogram Transformer paper and designed my implementation	model
4/17	3 hours	Implemented Audio Spectrogram Transformer. Tested a few batches w/out training.	model
4/18	1 hour	Created wrapper class for FLEURS dataset to train AST with	model
4/18	2 hours	Debugged training AST on FLEURS	model
4/20	1 hour	Gave up on FLEURS dataset and just used same dataset from CNN	model
4/20	1 hour	Compiled results and compared all of my trained models	model