Task 1: Speaker Verification

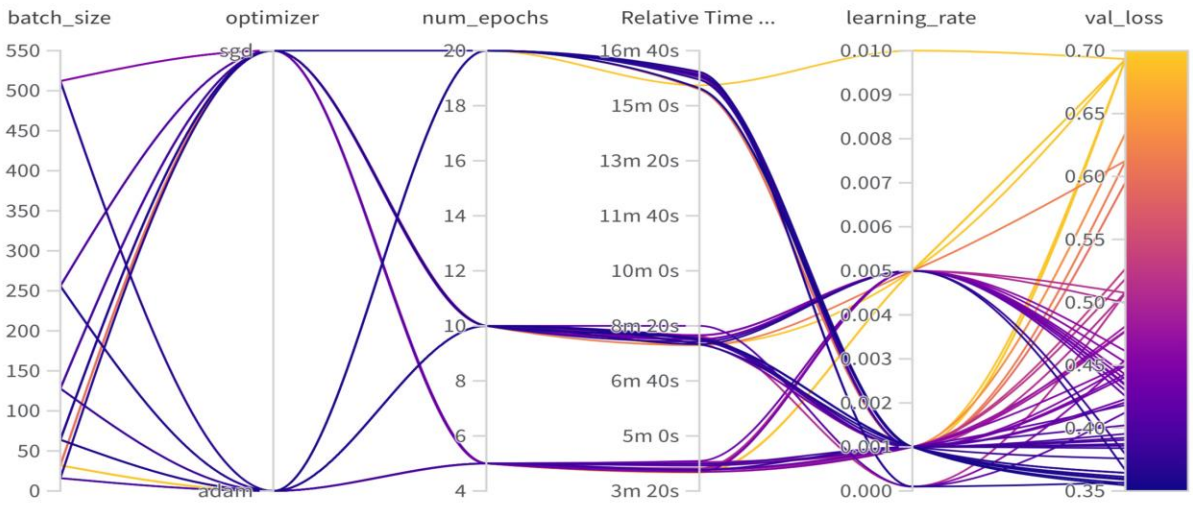
Contributors: Andrew Downing worked primarily on this task, with extra help from group members for condor and data pre-processing methods.

Methods:

Initially, for task 1 we started with some initial data analysis present in the dataset. After running a small script, we found that there were 373 unique speakers present in the train dataset. For each unique speaker, there were 10 audio representations of that class, and that the speaker class numbers didn't follow in a continuous order (eg. Speaker class: 11, 12, 15, 17, 18, 19). There was different speaker classes present in the training and dev datasets as dev showed 89 unique speaker classes, also each with 10 audio representations. When doing some literature review, it became evident that the "no free lunch theorem" was present as there were several models that were claimed to achieve working speaker verification success [1][2][3][4]. Of them, a few were of note. Siamese/Triplet networks were one of early baselines we tested because of their use case in other applications, like signature verification. Alternative methods include the use of x-vectors [2] as well as the ECAPA-TDNN model [4], although we found almost immediate overfitting on the training data occurred when attempting to use the X-vectors in training. Our model includes a 1D Siamese convolutional neural network (CNN) expanding from 64 filters to 256 to process 40 bin mel spectrogram inputs with 25ms frame size and a 10ms stride, extracting hierarchical features through each convolutional layer followed with ReLU activation and max pooling, selectively choosing the most prominent features to train on. We then pass that through a fully connected classification head using sigmoid activation to produce probability outputs for the inputs, where 1 is the true label and 0 is the false label. We implemented data-preprocessing by fixing the length of all the audio files to 5 seconds, by looping back to the start of the audio file if it was too short or clipping the file if it was too long. We then converted the audio files into mel spectrograms to capture relevant acoustic features present in each speaker. For training purposes, we randomly selected two audio files, one where both speakers were the same and a negative where both speakers were different. The output embeddings are then compared by calculating the absolute difference between them. The baseline CNN that we trained was able to achieve 75.62% accuracy on the dataset. We did not do any additional data augmentation or pre-processing.

Results:

Through testing and experimentation, we learned that a Siamese CNN network achieved the best validation accuracy of 85.64% and a loss of 0.3418. Our benchmark was able to get 81.64% validation accuracy with a batch size of 32 a learning rate 0.001. While hyperparameter tuning, several interesting occurrences came up, specifically concerning the Adam optimizer, indicating an issue with training instability. With any other hyperparameter combination, if a learning rate larger than 0.001 was submitted, both the training and validation accuracy would not exceed 50% regardless of however many epochs it would train. Even with this oddity, we tested among different hyperparameters, with a batch size ranging from 8 to 512, learning rate values from 0.1 to 0.0001, training epochs from 5 to 20, exceeding to 50 for the best performing hyperparameters, and an optimizer test between Adam, AdamW, and SGD. AdamW showed volatility in it's ability to generalize well and a learning rate of 0.0001 trained at an extremely slow pace, so we left it between Adam and SGD as they performed well. We also applied dropout to mitigate overfitting, but it had minimal effect on validation performance. Ultimately, the best hyperparameters chosen was a batch size of 64, a learning rate of 0.001, while using the Adam optimizer.



Conclusions:

Again, we used a Siamese convolutional neural network that achieved a validation accuracy of 85.64% and a loss of 0.3418. On train, it was able to achieve 91.32% accuracy indicating a slight overfitting issue but it even with dropout applied, it made little impact on the validation performance. While more complex models may perform better, it's best to note that many of these models perform similarly well and a tradeoff balance should be chosen specifically for the task. As far training goes, we found that increasing batch size beyond 64 yielded negligible results but more volatility in the models ability to generalize well. We also tried to train an LSTM based model, but encountered several CUDA issues that hampered our consideration of using that model. We also tried to train a X-Vector approach but found that the score function failed to generalize well because of the complexity of using cosine similarity in calculating a true/false probability. A slower learning rate does help squeeze additional performance at the cost of running substantially lower, at an additional rate of 60 epochs to reach similar performance as our best. The best model used a batch size of 64, a learning rate of 0.001, Adam as it's optimizer, and 30 epochs for runtime. We chose this model as it was simpler than the alternatives to program and modify and performed well given the initial baseline.

[1] T. Zhou, Y. Zhao, J. Li, Y. Gong and J. Wu, "CNN with Phonetic Attention for Text-Independent Speaker Verification," 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), Singapore, 2019, pp. 718-725, doi: 10.1109/ASRU46091.2019.9003826.

[2] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey and S. Khudanpur, "X-Vectors: Robust DNN Embeddings for Speaker Recognition," 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, Canada, 2018, pp. 5329-5333, doi: 10.1109/ICASSP.2018.8461375.

[3] N. M. Nandhitha, S. E. Roslin and B. Rajasekhar, "LSTM Based Speaker Recognition System with Optimized Features," 2023 Fifth International Conference on Electrical, Computer and Communication Technologies (ICECCT), Erode, India, 2023, pp. 1-4, doi: 10.1109/ICECCT56650.2023.10179810.

[4] Desplanques, B., Thienpondt, J., & Demuynck, K. (2020). Ecapa-tdnn: emphasized channel attention, propagation and aggregation in tdnn based speaker verification. Interspeech 2020. https://doi.org/10.21437/interspeech.2020-2650