Temporal Modeling Matters: A Novel Temporal Emotional Modeling Approach for Speech Emotion Recognition (TIM-Net)

Description

Platform

I have trained and tested the code on an *OSX-ARM* Architecture in a Python 3.8 *miniforge*-based environment.

Datasets

- Datasets used for testing my improvements in the paper: SAVEE, EMODB
- Raw Dataset and pre-processed MFCCs were used as is from Phase 1, available here.

Code

This Github Repository contains my **improved** code for both the paper with my proposed changes. The README.md has instructions to run the code.

Improvement Specifics

- **Temporal Dropout:** A 1DSpatialDropout layer was added immediately after the input layer to enhance generalisability and reduce overfitting. However, this led to greatly increased variability in the validation accuracy. To counteract this effect, instead of using a fixed dropout rate, it was gradually increased down the pipeline inside the TIMNET architecture.
- **Classifier Improvement:** A hidden layer of dimension 128 was added after the 39 dimensional output of the dynamic fusion model. This allowed the classifier more freedom in choosing the basis functions for the different emotions. This change was however accompanied by a steep increase in overfitting tendencies of the model and training time for each split also had to be increased (number of epochs). Hence, Focal Loss combined with label smoothening of the output 1-hot vectors, was used as a loss function for improving the robustness of the architecture.

  Note: Since forest-based classifiers don't train using the gradient-based methodology, it would require extra training time by extracting the trained weights of the current architecture. Due to lack of time, this was not attempted.
- **Shuffle Transformation:** The shuffle transformation was added parallel to the backward and forward transformations of the processed MFCCs across the Temporally aware blocks. This allowed the model to better capture the long term dependencies of the emotions from the various parts of the audio signal. The randomness however affects the stability of the model training strategy, and multiple training runs were conducted to obtain a model with the best performance.
- **Static Attention Layer:** The dynamic fusion module in the paper was replaced with a static attention mechanism. This allows for increased flexibility in choosing relevant information while pooling from the outputs of the TABs while retaining the simplicity of the model.

  Note: I experimented with both the popular variants of attention (self and cross) using the backwards and forward passes as input sequences. However, this did not prove to be very useful due to the small number of TABs. However the inflexibility of fixed weights was alleviated by the static attention, which uses a learned but fixed query vector for the data (key and value), assimilating all the relevant information from the different stages into a single output.

Results

Following validation accuracies are based on the hyperparameters chosen to optimise local training on my laptop. The WAR was calculated separately based on class distribution weightages in the dataset.

Table 1: UAR / WAR of Modified TIMNET

Improvement	Improved Accuracy
SAVEE - Phase 1 Accuracy: 0.7415 / 0.7667 **SAVEE** - Github Repository: 0.7726 / 0.7936	
Dropout	0.7630 / 0.7958
Dropout + Attention	0.7376 / 0.7770
Dropout + Attention + Classification	0.7369 / 0.7667
All Improvements	0.7884 / 0.8020
EMODB - Phase 1 Accuracy: 0.8377 / 0.8523 **EMODB** - Github Repository: 0.8919 / 0.9028	
Dropout	0.8606 / 0.8787
Dropout + Shuffle	0.8659 / 0.8746
Dropout + Classification	0.8686 / 0.8823
All Improvements	0.8481 / 0.8598

Note: The improvements were rigorously tested and tuned for SAVEE. To test the generalisation ability of the model, similar runs were conducted for EMODB without any hyperparameter tuning, owing to the lack of time.

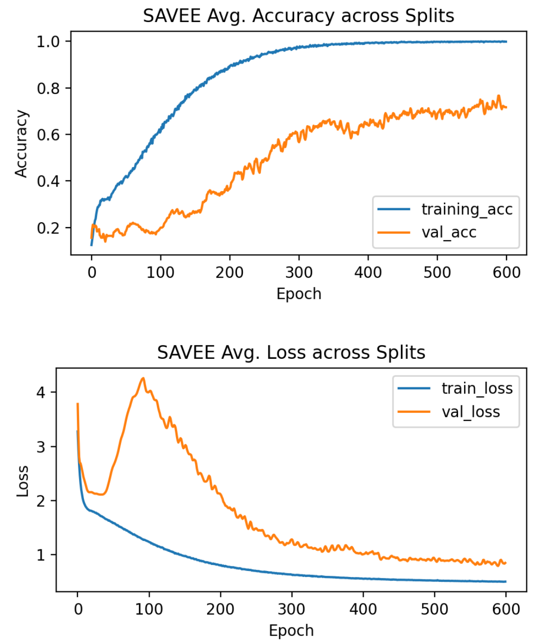
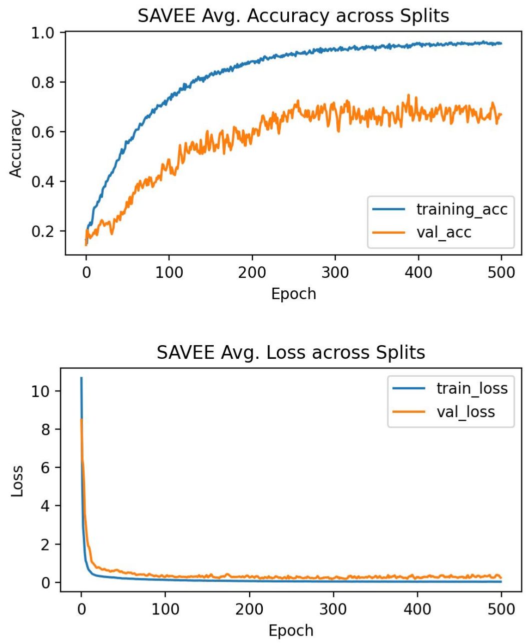
Observations and Conclusions

- An improvement of almost 4% in both UAR and WAR was observed on the SAVEE dataset when compared to the model I had previously trained in Phase 1 (see table 1). The model also beats the updated results provided by the original authors in their Github repository on SAVEE by almost 1%.
- As compared to Phase-1's outputs, the variations in the loss curve were also smoothened out by the improvements, as evident in table 2.
- The original model saturates to a full 100% training accuracy very early on and easily overfitted the dataset. With the changes, the accuracy increases in tandem with the validation accuracy, and the model's overfitting tendencies are greatly minimized.
- As seen in table 1, the results on EMODB have also improved without any hyperparameter tuning for the dataset.
- The introduction of the static attention mechanism with the controlled parameters for Adam's algorithm led to an improved training speed as evident by the lesser epochs and quicker convergence of the loss curves in table 2.

Future Work

TIMNET is a small yet powerful architecture for SER, when compared to its alternatives and is very suitable for edge devices. Here, I have improved the performance on the SAVEE dataset. Further testing, specially on datasets with larger audio files like IEMOCAP is required to validate our improvements. We could also experiment with better transformations for achieving a stable training of the model.

Table 2: Visualised Results of the proposed TIM-Net on SAVEE

Phase 1	**Phase 2**
 SAVEE Avg. Accuracy across Splits Accuracy 1.0 0.8 0.6 0.4 0.2 0 100 200 300 400 500 600 Epoch training_acc val_acc SAVEE Avg. Loss across Splits Loss 4 3 2 1 0 100 200 300 400 500 600 Epoch train_loss val_loss	 SAVEE Avg. Accuracy across Splits Accuracy 1.0 0.8 0.6 0.4 0.2 0 100 200 300 400 500 Epoch training_acc val_acc SAVEE Avg. Loss across Splits Loss 10 8 6 4 2 0 0 100 200 300 400 500 Epoch train_loss val_loss