Emotion Recognition in English and Greek Speech: Deep CNN with 2D Spectral Features and Transfer Learning

Thesis · September 2023

DOI: 10.13140/RG.2.2.21568.51206

1 author:

Georgios Skoulidis
University of Patras
2 PUBLICATIONS **0** CITATIONS

This work is based on the thesis "Speech Emotion Recognition using Deep Learning" [9] and aims to build a robust system for recognizing speech emotions through the utilization of supervised labeled data and advanced deep-learning techniques with an image classification approach. The core objective is to construct a model that can precisely distinguish between 7 emotional states for English speakers. The investigation extends to the analysis of the influence of three specific spectral features, extracted from the audio samples. These features are treated as images, each displayed in four different output sizes resulting from a quadratic transformation achieved through bilinear image interpolation. We also emphasize the evaluation of three custom abstract Convolutional Neural Network (CNN) architectures. Furthermore, we study the impact of the optimized pre-trained English Speech Emotion Recognition model when applied to speech samples from Greek speakers. A limited dataset of Greek speech is employed to train, validate, and test the model's performance, while we assess the knowledge of the model's pre-trained layers.

Introduction

In a time where artificial intelligence applications such as super-intelligent chatbots, advanced home automation systems, autonomous vehicles, and art generation networks have become an integral part of our daily lives, the importance of researching Speech Emotion Recognition (SER) is greater than ever before. While other subdomains of artificial intelligence and machine learning have made significant progress, Speech Emotion Processing (Recognition and Synthesis) remains a crucial and practical field to explore.

Understanding human emotions holds great significance as emotions play a fundamental role in our lives, interactions, and decisions. Speech is one of the main ways through which emotions are expressed and comprehended, while the way of verbally expressing them differs a lot between cultures, languages, and dialects.

This research will mainly explore SER in English speech, with a secondary focus on Greek speech. The effects of different parameter combinations will be studied and various architectures of Convolutional Neural Networks (CNNs) [3] will be investigated. The performance of specific acoustic spectral features extracted from the initial speech data, used to train the models, will be also observed. These features will be treated as images and fed into the CNN, incorporating the domain of image recognition. After training the optimal English SER CNN model, we will explore the capability of adapting the model to Greek speech.

The objective of this research is to uncover valuable insights into the domain of English Speech Emotion Recognition, analyze the impact of different parameter combinations, and investigate the adaptability of an English SER classifier to Greek speech data.

Experimental Setup

The code was executed in **Python 3.10** [8], while the specific framework we used for deep learning was **PyTorch**. Some of the primary modules involved, included Scipy, Scikit-Learn, PIL (image processing), and Librosa (speech and audio processing)

To achieve satisfactory results within reasonable timelines, we utilized an **NVIDIA TESLA V100-PCIE-32GB GPU** for training and testing our models.

English SER Data Collection

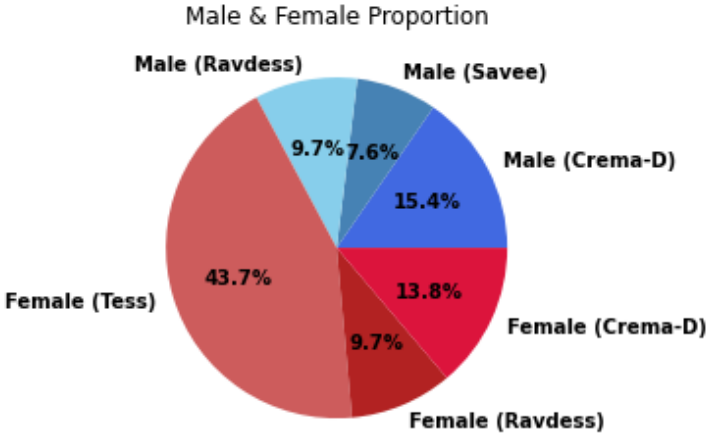
Our classifiers were trained and evaluated using data samples collected from four distinct English SER databases, to distinguish between the seven emotion classes in Ekman's model; $happy$, $angry$, $disgust$, $fearful$, sad, $neutral$ and $surprised$.

Those were RAVDESS (Ryerson Audio-Visual Database of Emotional Speech and Song) [5], TESS (Toronto Emotional Speech Set) [2], SAVEE (Surrey Audio-Visual Expressed Emotion) [7], and CREMA-D (Crowd-Sourced Emotional Multimodal Actors Dataset) [1].

Table 1: Emotion Distribution across the Total Dataset.

Emotion	RAVDESS*	TESS*	SAVEE	CREMA-D*	Total
Happy	192	400	60	273	925
Angry	192	399	60	273	924
Disgusted	192	400	60	273	925
Fearful	192	399	60	273	924
Sad	192	400	60	273	925
Neutral	96	400	120	509	1125
Surprised	192	400	60	0	652
Total	1248	2798	480	1874	**6400**

It is evident that the neutral class comprises more samples than any other class in our dataset. In everyday life, neutral speech is more common than speech expressing specific emotions such as happiness or anger. As a result, neutral samples occupy approximately 17.58% of our final dataset, whereas the remaining classes account for up to 14.45% of the total audio samples. While the classifier demonstrates strong capability in recognizing neutral class speech, it is expected to have lower accuracy when classifying surprised speech due to the scarcity of training examples for that particular class. This class's imbalance can impact the model's performance and may lead to relatively lower accuracy in identifying surprise emotions compared to other emotions. Logically speaking, however, a speaker should feel less frequently surprised than happy, angry, or sad.



It is important to note that a dataset can sometimes exhibit an unnatural distribution regarding the speaker's gender. Sadly, our SER classifier is likely to exhibit some bias towards the emotional context of female speakers, as approximately 67.2% of the total data samples are estimated to be from female speakers. Due to the random selection process of instances from the CREMA-D database, we can only provide an estimation of the percentage. In this database, 1874 samples out of 7442 total samples were randomly selected, and 48 actors out of 91 in the dataset are male. Our estimation is that we have $\frac{48}{91} \times 1874 = 988$ male speaker utterances and $\frac{43}{91} \times 1874 = 886$ female speaker utterances.

Greek SER Data Collection

The Greek database we used was the Acted Emotional Speech Dynamic Database [10]. The speech samples there belong to 5 different emotional classes: $anger$, $disgust$, $fear$, $happiness$, and $sadness$ [11]. The dataset contains a total of 605 speech samples, however by removing a single corrupted audio file from the sad class, we utilized 604; 121 anger files, 122 disgust files, 120 fear files, 119 happiness files, and 122 sadness files. 3 out of 5 speakers were female, making our dataset almost appropriate, as the pre-trained English SER classifier will already have some bias towards female speakers.

Data Pre-Processing

After we have gathered our SER data, the speech waveforms were appropriately manipulated before being fed to our CNN models. The pre-processing steps are listed below:

1. Pre-Emphasis
2. Trim Zeros (both in front and at the end of the signals)
3. Spectral Feature Extraction
4. Convert amplitude to dB scale
5. NxN Resize Transform

The spectral features that have been extracted for classification were **STFT chromagrams**, **Mel spectrograms**, and **MFCCs**. The necessary functions in Python were provided by Librosa [6]. As for the resize transform, we used the bilinear image interpolation technique [4] to convert our rectangular images into **NxN** square images. Moreover, the time-axis of the extracted features was rescaled to match the length of the frequency-axis, the size of which has been predetermined when calling the feature extraction function.

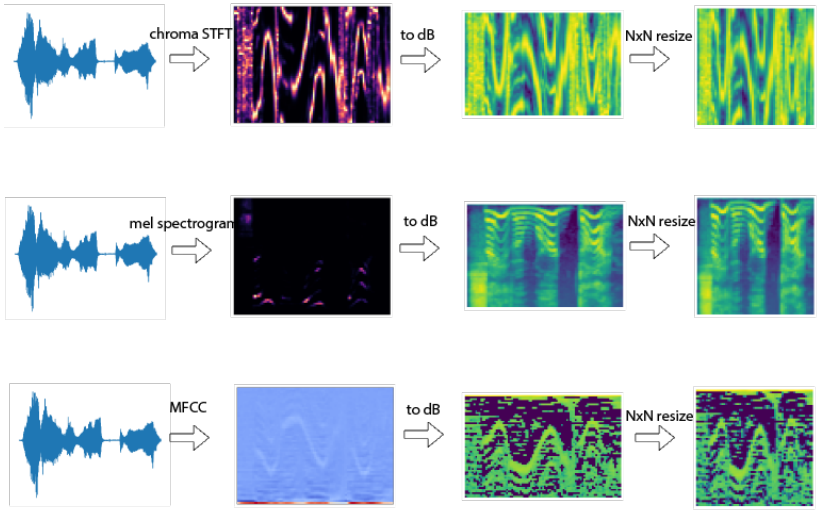


Figure 1: Pre-Processing Steps: Feature Extraction, Convert to dB and NxN Resize Transform.

CNN Architectures

The labeled, square image outputs were then passed as inputs to a CNN for both training and validation.

This study presents three custom, abstract convolutional neural network architectures for 2D acoustic feature speech emotion classification tasks, the names of which were inspired by the popular Pokémon characters. We refer to these networks as **BulbaNET**, **CharmaNET**, and **SquirtleNET**. Each network has three convolutional layers, three fully-connected layers, and a variety of different pooling layers (max or average), activation functions, and regularization layers that can be seen in 2. Several of their internal architecture parameters had initially been undetermined and were optimized through the tuning process.

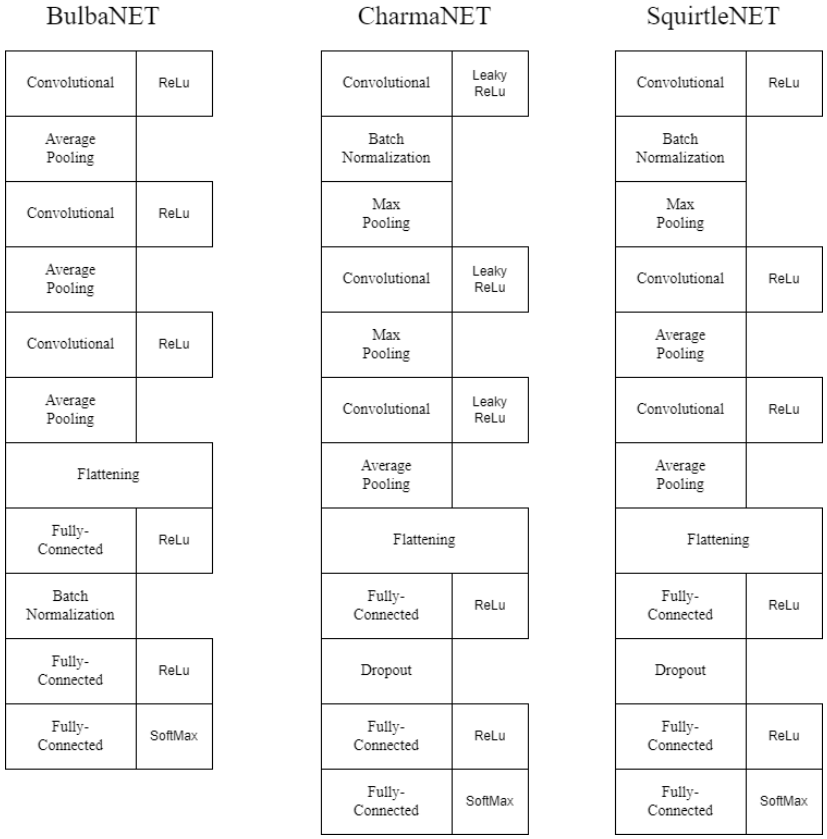


Figure 2: CNN Architectures: BulbaNET, CharmaNET and SquirtleNET.

Parameter Tuning

A variety of parameters were tuned to improve performance. We categorize the parameters which were optimized, into **primary** and **secondary** parameters.

The primary parameters used for optimization included the three following:

- 2D acoustic spectral feature.
- Image size after the interpolation method.
- CNN architecture.

The secondary parameters used for optimization included all of the following:
CNN structural parameters:

- Output channels of each convolutional layer.
- Output features of the first two dense layers.
- Kernel size of each convolutional layer.
- Kernel size of each pooling layer.
- Stride in each pooling layer.
- Probability for an element to be zeroed in the dropout layer.
- Momentum in the batch normalization layer (controls the running mean and running variance).

Hyperparameters:

- Batch size.
- Learning rate.

Additionally, certain parameter values were fixed as **default**, and did not participate in tuning:

- Stride in convolutional layers: 1.
- Stride in SquirtleNET's pooling layers: 2.
- Padding in convolutional layers: 1.
- Negative slope in CharmaNET's Leaky ReLU: 0.010 (Partially Tuned).
- Pre-Processing parameters.
- CrossEntropyLoss() (torch.nn)
- Adam optimizer without learning rate decay.

Enhance Generalization

a) 5-Fold Cross Validation: To ensure a fair comparison of parameter vectors and validate trained models effectively, we employ a distinct set of unseen data for validation, separate from the original test set, which should only be used to evaluate the ultimate performance of the optimal model.

To achieve this, we implement 5-fold cross-validation. This entails dividing the original training set into five equally-sized subsets or folds. In each iteration, one fold is designated as the validation set, while the remaining four serve as the training data for model training. This process is repeated five times, with each fold taking on the role of the validation set once. Then, the average performance metric is computed for every iteration. This approach prevents additional overfitting and provides a more powerful approximation of the model's performance.

b) Speech Manipulation Techniques: To enhance generalization further, we modified a small portion of each cross-validation subset within the 5-fold cross-validation method. Specifically, we subjected each original waveform extracted from the TESS database to a manipulation technique before proceeding with the standard pre-processing steps. These techniques encompassed: i) Pitch-Shifting, ii) Time-Stretching, and iii) Gaussian Noise addition. We selectively applied different pairs of these three audio manipulation methods to introduce greater variation in the signal features, thereby simulating a wider range of real-world speaking characteristics.

To implement these audio processing methods, certain parameters had to be defined. For time-stretching, the stretch rate parameter was determined. For pitch-shifting, the shifting was specified in a semitone scale. For Gaussian noise addition, the signal-to-noise ratio (SNR) in decibels (dB) was defined, while the length of the noise signal was set to be the same as the length of each speech segment, and the mean of the Gaussian noise was set to 0. Each sample was assigned random parameter values selected from specific grids. The grids of values used are listed below:

- Stretch rate values: $0.75, 0.80, 0.85, 0.90, 1.05, 1.10, 1.20, 1.25$
- Semitone values: $-2, -1, +1, +2$
- SNR(dB) values: $7, 8, 11, 12, 14, 15, 17, 18, 20$

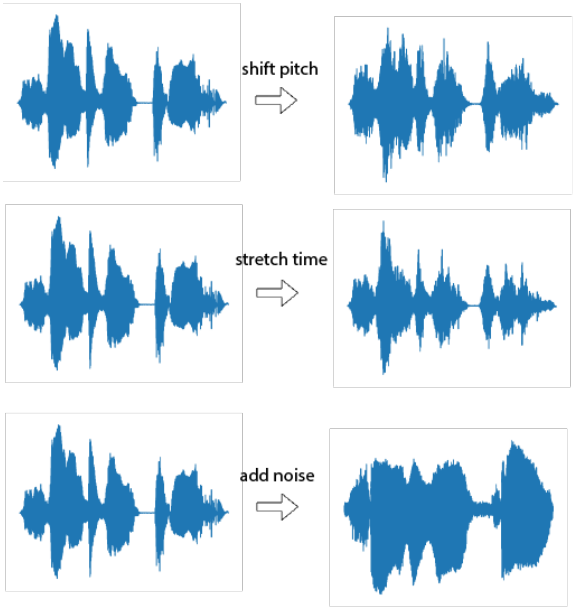


Figure 3: Data Manipulation Techniques applied in the TESS dataset.

c) Early-Stopping Algorithm: Instead of training each model for a specific number of epochs, we chose to define a set of early stopping conditions based on the models' behaviors. For this, we performed an extra step. We trained several models, each containing a set of different random

parameters, for a high number of epochs and carefully observed their convergence patterns and performance trends. We monitored the train loss - test loss curve as well as the testing accuracy and macro F1-score values per training epoch.

After diagnosing the curves and observing the overall behavior, we were able to gain a brief insight into what kind of early stopping conditions we should apply to training. We set the maximum number of training epochs equal to 300, and two sets of conditions were defined, along with a third overall condition.

Class A early stopping conditions were designed to immediately stop the cross-validation process if a model's classification ability was considered poor. Moreover, if the Macro F1 score fell below a predefined threshold in specific epochs, the training and validation processes were abandoned. In such cases, a new set of random parameters was selected for the successive iterations. This class of conditions allowed us to identify and discard poorly performing models quickly, ensuring efficiency in the optimization process.

Class B early stopping conditions were responsible for detecting overfitting and preventing further training of a high-performing model, also reducing the computational cost. For each predefined step of epochs, we examined a set of consecutive test losses and Macro F1 scores. If the values consistently increased for losses or decreased for F1 scores, it indicated the emergence of overfitting. In such cases, we stopped training to avoid the model becoming inordinately specialized to the training data. The maximum number of epochs for the next cross-validation folds, was set based on the point at which convergence began to emerge. Most importantly, this saved us a significant amount of computational time and resources.

In addition to the class A and class B conditions, we also incorporated an overall early stopping condition, referred to as the C condition. This condition aimed to terminate the validation process if either the Accuracy or Macro F1 score values, evaluated after an entire training fold, fell below a predefined threshold. The purpose of this condition was to address scenarios where models might have survived the class A conditions, indicating a certain level of classification ability, but still exhibit very poor overall performance.

Those conditions can be seen below:

- A1: Stop if epoch equals 20 and macro F1 score (f1) is less than Threshold%.
- A2: Stop if epoch equals 50 and macro F1 score (f1) is less than Threshold%.
- A3: Stop if epoch equals 75 and macro F1 score (f1) is less than Threshold%.
- A4: Stop if epoch equals 100 or epoch equals 125 and macro F1 score (f1) is less than Threshold%.
- A5: Stop if epoch equals 150 and macro F1 score (f1) is less than Threshold%.
- A6: Stop if epoch equals 200 and macro F1 score (f1) is less than Threshold%.
- A7: Stop if epoch equals 250 and macro F1 score (f1) is less than Threshold%.

Class B:

- B1: Stop if epoch is greater than 50, divisible by 4, and conditions k1, k2, k3, and k4 are all true, where:
  - k1: Test loss at epoch is greater than test loss at epoch-4.
  - k2: Test loss at epoch-4 is greater than test loss at epoch-8.
  - k3: Test loss at epoch-8 is greater than test loss at epoch-12.
  - k4: Test loss at epoch-12 is greater than test loss at epoch-16.
- B2: Stop if epoch is greater than 50, divisible by 4, and values v1, v2, v3, and v4 are all less than 1.0%, where:

  - v1: Difference between macro F1 score at epoch and macro F1 score at epoch-4.
  - v2: Difference between macro F1 score at epoch-4 and macro F1 score at epoch-8.
  - v3: Difference between macro F1 score at epoch-8 and macro F1 score at epoch-12.
  - v4: Difference between macro F1 score at epoch-12 and macro F1 score at epoch-16.
- B3: Stop if epoch is greater than 110, divisible by 40, and the difference between macro F1 score at epoch and macro F1 score at epoch-40 is less than Threshold%.

Class C:

- C: Stop if fold's macro F1 score is less than Threshold% and fold's accuracy is less than Threshold%.

The E.S. Conditions' Thresholds adopted different values depending on the experimental phase. As the parameter tuning process progressed, these values were set to higher ones, meaning that the conditions were gradually becoming stricter.

English SER Model Optimization

For training the models, we split the total English dataset into training and test sets with an 80%/20% proportion, utilizing the `CrossEntropyLoss()` from the `torch.nn` module, and during training, we employed the `Adam optimizer without any learning rate decay`. We also have already constructed the early-stopping algorithm to minimize computational requirements.

To tune all the parameters, we followed a series of steps that constitute the **general methodology**. Let's take a look at these steps before moving on.

1. Define the spectral feature and the image size.
2. Execute all of the Pre-Processing steps.
3. Split the data into training and testing sets (80%-20%). Use only the training set for tuning.
4. Specify the grid of secondary parameters and the searching method.
5. Define the Early-Stopping algorithm conditions (if any).
6. For every secondary parameter vector, use 5-fold cross-validation and compute the average evaluation metrics for all folds. We computed the following metrics for comparison:
   - Train loss
   - Test loss
   - Accuracy
   - Macro F1 score
7. Analyze learning curves to better understand the problem and identify potential overfitting (only if necessary).
8. Select the parameter combinations with the highest performance based on the average metrics computed in the cross-validation process.

The general methodology described above was implemented in four different parameter qualification stages, during which certain steps were occasionally modified. The objective of these four stages was to explore various sets of parameters, including different CNN architectures and acoustic spectral features, and retain those that yielded the best performance. We examined the various parameters by employing the random search algorithm across different grids, as outlined below:

1. Conduct an initial secondary-parameter random search for every specified combination of primary parameters. For the secondary parameters, the same vectors are used for the first ten iterations of every primary parameter combination. An additional random search was performed for two new Leaky ReLu Slopes in CharmaNET, for 50x50 Mel Spectrograms and STFT Chromagrams. **(First Parameter Qualification Stage)**

2. Perform a second round of random search using only the most promising primary parameters. Select the combinations that display the highest performance and merge them with the previously best combinations. **(Second Parameter Qualification Stage)**
3. Select the best three parameter vectors from the previous list and conduct an additional random search with slight modifications to the existing secondary parameters. Add the combinations that demonstrate the highest performance to the list. **(Third Parameter Qualification Stage)**
4. Select the best models from the list and train them for a greater number of epochs without applying any Early-Stopping condition. Identify the optimal parameter vector. **(Fourth Parameter Qualification Stage)**

Below are the values used to define the initial parameter grid:

Convolutional Output Channels: 100, 116, 132, 148, 164, 180, 196, 212, 228, 244, 260, 276, 292, 308, 324, 340, 356, 372, 388, 404, 420

Convolutional Kernel Sizes: 2, 3

Pooling Kernel Sizes: 2, 3

Pooling Strides: 2, 3

Batch Normalization Momentum: 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9

Dropout Probability: 0.25, 0.30, 0.35, 0.40, 0.45, 0.50

Fully Connected Output Channels: 1024, 1152, 1280, 1408, 1536, 1664, 1792, 1920, 2048, 2176, 2304, 2432, 2560, 2688, 2816, 2944, 3072, 3200, 3328, 3456, 3584, 3712, 3840, 3968, 4096, 4224

Batch Size: 16, 32, 48, 64

Learning Rate: 2e-8, 6e-8, 1e-7, 5e-7, 1e-6, 3e-6, 6e-6, 9e-6, 2e-5, 4e-5, 6e-5, 8e-5, 1e-4, 3e-4, 7e-4

The Thresholds set for the E.S. Algorithm in 1st Qualification Stage are:

- A1: 17%, A2: 20%, A3: 24%, A4: 27%, A5: 36%, A6: 39%, A7: 42%
- C: 45% and 42% **or** 50% and 50%

The Thresholds set for the E.S. Algorithm in the 2nd Qualification Stage are:

- A1: 42%, A2: 45%, A3: 48%, A4: 51%, A5: 54%, A6: 56%, A7: 58%
- C: 60% and 60%

Table 2: Highest-Performance Parameter Vectors after Third Parameter Qualification Stage.

ID	000	001	010	011	100	101	110	111
CNN	Charma Net	Squirtle NET	Squirtle NET	Squirtle NET	Squirtle NET	Squirtle NET	Squirtle NET	Squirtle NET
Feat	Mel Spec	Mel Spec	Mel Spec	Mel Spec	Mel Spec	Mel Spec	Mel Spec	Mel Spec
NxN	80x80	60x60	60x60	60x60	60x60	60x60	60x60	60x60
COC	388, 116 116	220 140 356	404 356 372	212 148 356	236 124 356	204 124 356	188 156 340	212 148 356
CKS	3 2 2	3 3 3	3 2 3	3 3 3	3 3 3	3 3 3	3 3 3	3 3 3
PKS	2 3 2	2 3 3	2 3 2	2 3 3	2 3 3	2 3 3	2 3 3	2 3 3
PS	2 2 2	2 2 2	2 2 2	2 2 2	2 2 2	2 2 2	2 2 2	2 2 2
BNM	0.1	0.3	0.4	0.2	0.3	0.4	0.4	0.3
DP	0.30	0.45	0.25	0.45	0.50	0.45	0.40	0.45
FCOC	1664 2688 7	3610 2910 7	1792 1280 7	3840 2858 7	3610 2816 7	3926 2858 7	3634 2816 7	3840 2944 7
BS	16	32	48	32	32	32	32	32
LR	2e-05	6e-05	8e-05	6e-05	6e-05	6e-05	6e-05	6e-05
ET(min)	22.17	13.67	38.45	29.88	21.40	18.07	17.78	31.14
EP	52	52	64	96	76	64	64	60
Macro F1-Score %	62.43	63.20	63.91	64.03	64.60	65.27	65.82	66.42
Accuracy %	63.07	63.61	64.39	64.80	64.92	65.74	65.29	66.89

Furthermore, an ID tag was assigned to each parameter vector and displayed in the first row in order for us to easily be able to refer to them. We finalized our search process, not just by comparing the Macro F1 scores and Accuracy values, but by also analyzing the learning curves through two types of graphs: the test loss - train loss curves per epoch and the Macro F1 score - Accuracy per epoch. In other words, we have conducted curve diagnosis in order to qualify the best results. All of the graphs are displayed in figures 4 and 5.

While not much can be inferred from the loss curves, it appears that (001) has not converged to the optimal value yet, indicating that the model still has the potential to learn. This lack of convergence is more likely due to a misleading early stopping, activated condition. Considering that this model would require additional training to reach its true performance level, we have chosen not to exclude it. In addition, models (101), (110), and (111) have also demonstrated Macro F1 scores and Accuracies above 65%, all in convergence. As a result, after eliminating the rest of the parameter vectors, the updated list of optimal models is (001), (101), (110), and (111).

An important point to note is that convergence in learning does not necessarily indicate true overfitting. In our case, convergence simply signifies a reduction in the rate at which the model is learning and the area where the model learns less and harder. Overfitting occurs when the performance of the model starts to decline significantly. It is worth mentioning that the E.S. algorithm has performed admirably thus far. However, since the number of epochs required for convergence appears to be similar for all qualifying models, we can extend the training by a few more epochs to measure the progress and learning capacity of these models.

Next, we trained all the qualifying models for 150 epochs without implementing E.S. conditions and then compared their accuracy and macro F1-scores to determine which model performs best.

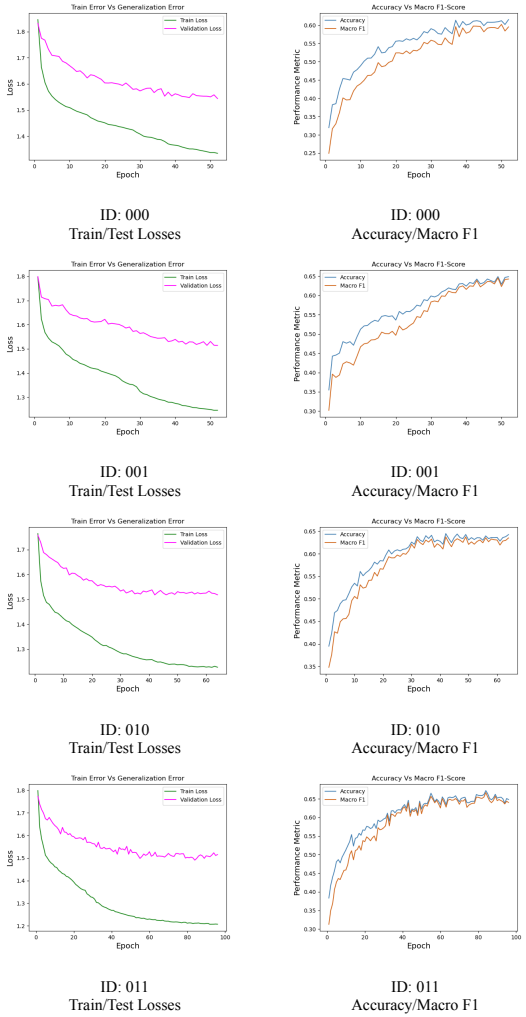
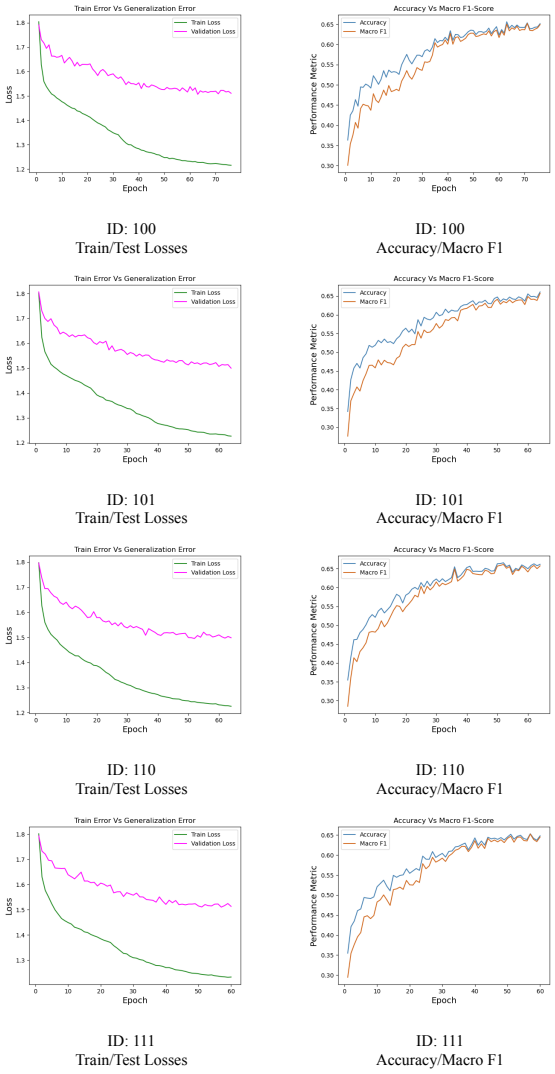


Figure 4: Learning Curves of the Qualifying models: ID:000 to ID:011



ID: 100 Train/Test Losses | ID: 100 Accuracy/Macro F1

ID: 101 Train/Test Losses | ID: 101 Accuracy/Macro F1

ID: 110 Train/Test Losses | ID: 110 Accuracy/Macro F1

ID: 111 Train/Test Losses | ID: 111 Accuracy/Macro F1

Figure 5: Learning Curves of the Qualifying models: ID:100 to ID:111

Hence, Table 3 shows the performance results for training the qualified models for 150 epochs.

Table 3: Best Models trained for 150 Epochs.

ID (Model)	Epochs needed	Macro F1-Score (%)	Accuracy (%)
001	150	63.14	64.02
101	150	65.16	65.53
110	150	65.35	66.05
111	150	64.21	65.00

We could state that all of the above combinations would easily be a good fit for our SER model. Foremost, if the training and validation process were repeated, the results would very possibly be significantly different than before, especially considering that every batch in the training set is randomly re-shuffled each time.

It appears that the parameter vector (110) achieves the maximum macro F1-score and accuracy for 150 epochs of training. This will be our optimal model.

Table 4: Optimal Parameters for English SER Model.

CNN Architecture	Squirtle NET
Spectral Feature	Mel Spectrogram
NxN	60x60
Convolutional Output Channels	188, 156, 340
Convolutional Kernel Sizes	3, 3, 3
Pooling Kernel Sizes	2, 3, 3
Pooling Strides	2, 2, 2
Batch-Normalization Momentum	0.4
Dropout Probability	0.40
Fully-Connected Output Channels	3634, 2816, 7
Batch Size	32
Learning Rate	6e-05

In the final step of our English SER experimental phase, the SquirtleNET model, configured with the optimal model parameters determined earlier, was trained using 60x60 mel spectrograms from the entire training dataset. The model was trained for a large number of 4,200 epochs and evaluated on a separate testing dataset that had not been previously used, without E.S. conditions.

The objective here was to capture the model's behavior at the exact moment when the generalization loss reached its minimum value. While this approach is similar to certain E.S. techniques, the intention was not to prematurely stop the training process but to test the model's performance every 10 epochs instead and only store the optimal model (the one that displays the minimum generalization loss). If the current testing loss was lower than the minimum loss recorded thus far, the model was stored or overwritten, and the current loss value was updated as the new minimum.

One of the reasons behind the decision to continue the training process without stopping it was to showcase the learning curve and potentially identify any signs of overfitting. By allowing the training to proceed for an entire duration of 4,200 epochs, it became possible to visualize the model's learning progress over time and display it in this thesis dissertation for future reference.

Table 5: Every Update in Final English SER model's Training.

Epoch	Test Loss	Accuracy	F1-Score
0	1.61215	0.5453	0.4956
10	1.52352	0.6359	0.5847
20	1.41498	0.7461	0.7459
30	1.38019	0.7828	0.7816
40	1.36727	0.7945	0.7946
50	1.35884	0.8008	0.797
90	1.35772	0.8055	0.8073
150	1.35546	0.8078	0.8089
350	1.35534	0.8094	0.8077
400	1.348	0.8188	0.8183

The total execution time for training and testing the SquirtleNET model for 4,200 epochs was approximately 253.04 minutes, which is equivalent to about 4 hours and 13 minutes. A summary of the epochs when the model was updated during the training process, along with some significant performance metrics is provided in 5.

The model reached its optimal performance after being trained for 400 epochs. During this period, it achieved a minimum generalization loss of 1.348, accompanied by an Accuracy of **81.88%** and a Macro F1-score of **81.83%**. These metrics indicate the model's ability to generalize and accurately classify speech emotions.

In Figure 6, we can see the confusion matrix of the optimal model's classification. In Figure 7 we display two learning curves, one for the train/validation losses and the other for accuracy/macro F1-score, both with respect to training epochs.

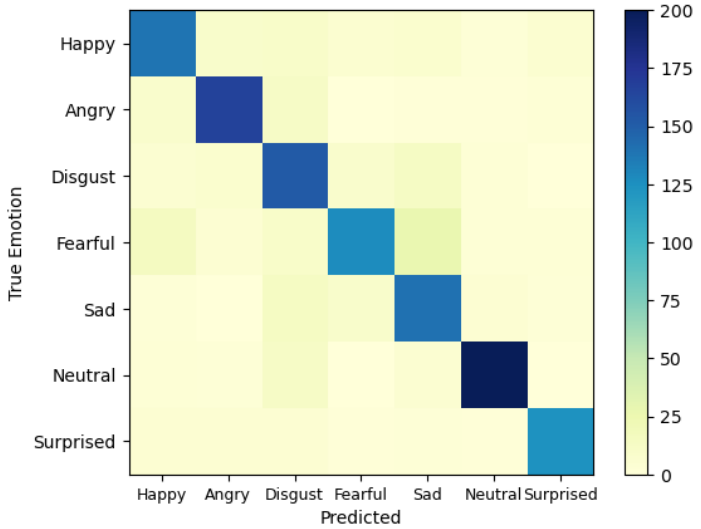


Figure 6: Confusion Matrix.

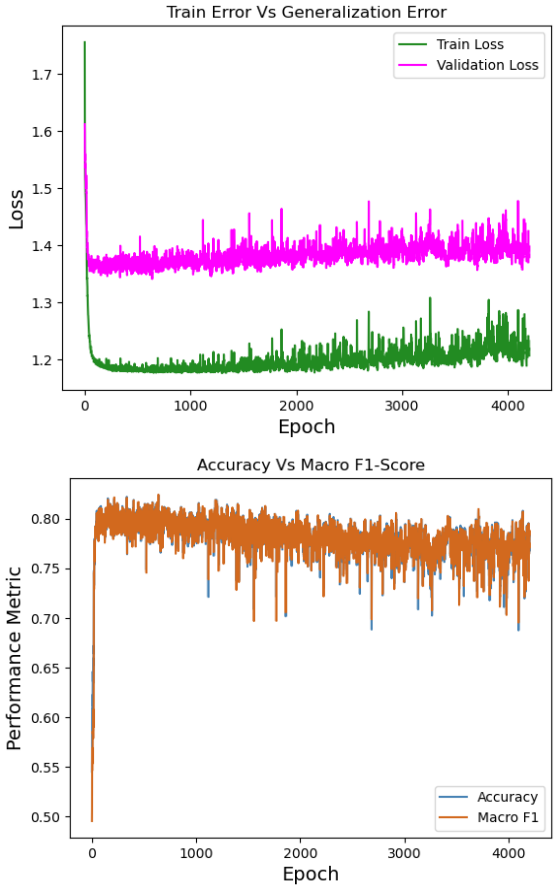


Figure 7: Losses and Performance Metrics.

Overall, the classification results appear promising and satisfactory. The confusion matrix, along with the other performance metrics, demonstrate that the classifier is capable of accurately distinguishing between the seven emotion classes. This indicates that the model has learned to capture and differentiate the unique features associated with each emotion.

However, it is important to acknowledge that the SER problem poses several challenges. The performance of the trained classifier may be influenced by various biases, limitations, and assumptions inherent to the dataset and the model itself. These biases could arise from factors such as data collection methods, class representation imbalances, and potential cultural or contextual differences.

Evaluating the Efficacy of Transfer Learning in Greek SER

One possible question that could be raised is whether our English SER classifier is suitable for recognizing emotions in various types of speech data. It is interesting to explore speech samples generated by speakers of different languages. If the already trained classifier underperforms on different language data, it impels us to consider how much we can utilize its existing training and knowledge.

Thus, we analyzed the applicability of the pre-trained model's knowledge in Greek SER, for 5 emotional classes. We treated this as a different task from the previous one where we predicted emotions in English phrases, for 7 emotional classes. Transfer learning techniques were utilized to gain insights into the similarities and differences between the English and Greek SER tasks. Our approach involved freezing a subset of layers from the pre-trained model and training the remaining layers on the new Greek dataset. The goal was to achieve high performance despite the limited number of samples available. By employing this transfer learning methodology, we aimed to identify which pre-trained layers possess significant knowledge that can contribute to our new SER task in the Greek language.

To simplify referencing the models, we assigned some letters as identity tags to each variation:

	Conv. 1	Batch-Normalization	Conv. 2	Conv. 3	Fully-Connected 1	Dropout
(a)	FROZEN	FROZEN	FROZEN	FROZEN	FROZEN	FROZEN
(b)	FROZEN	FROZEN	FROZEN	FROZEN	FROZEN	
(c)	FROZEN	FROZEN	FROZEN	FROZEN		
(d)	FROZEN	FROZEN	FROZEN			
(e)	FROZEN	FROZEN				
(f)	FROZEN					
(g)						

Each subset of frozen layers represents a distinct model. We can attempt to identify the ideal subset of frozen layers during the tuning process. It is highly important to mention that the third and last fully-connected layer of the pre-trained model was replaced with a new, untrained fully-connected layer featuring 5 output channels.

Except for the subset of freezing layers, the learning rate, and the batch size, all of the other parameters remained the same and were not used for tuning. We now refer to SquirtleNET, as a fully defined CNN architecture, for which we know all of its internal parameters.

Hence, a pre-trained SquirtleNET model, with its first layers frozen (c), was trained for several hyperparameter combinations and later compared with a pre-trained SquirtleNET model which was now re-trained with a variety of learning rates and batch sizes, without any layer frozen (g). Both models were trained for a total of 150 epochs. The performances were validated using the 5-fold cross-validation technique, similar to before.

Despite being trained and validated using fewer hyperparameter combinations, model (g) attained the highest maximum Accuracy%. However, this alone was insufficient to infer that fewer frozen layers yield better results. Therefore we had to evaluate performance across a wider range of distinct frozen layer subsets.

Another valuable insight gained from this experimental stage is a better understanding of the best-performing learning rate range. Consequently, we could now focus on a narrower set of learning rate values moving forward.

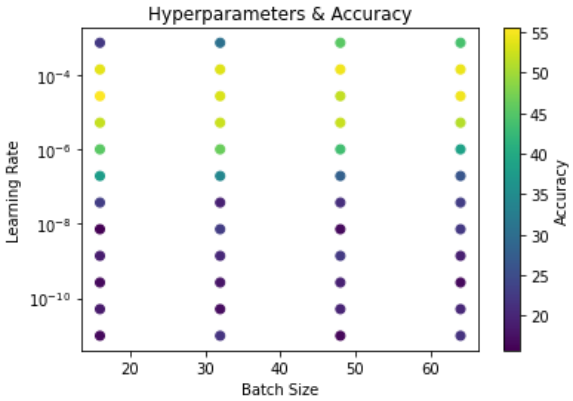


Figure 8: Accuracy% with respect to hyperparameter pairs, for model (c), when training for 150 epochs.

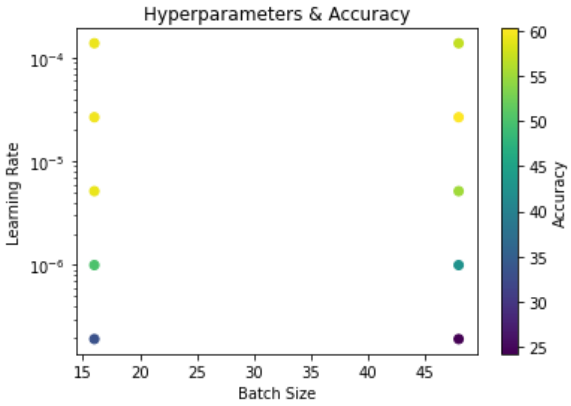


Figure 9: Accuracy% with respect to hyperparameter pairs, for model (g), when training for 150 epochs.

To gain a deeper understanding of which pre-trained layers have a significant im-

pact on the SER task, we conducted a more comprehensive grid search. We restricted our learning rate range to six values between 1e-6 and 1e-4 and explored four different batch sizes: 16, 32, 48 and 64. The aim was to examine additional model variations and identify the optimal combination of frozen layers.

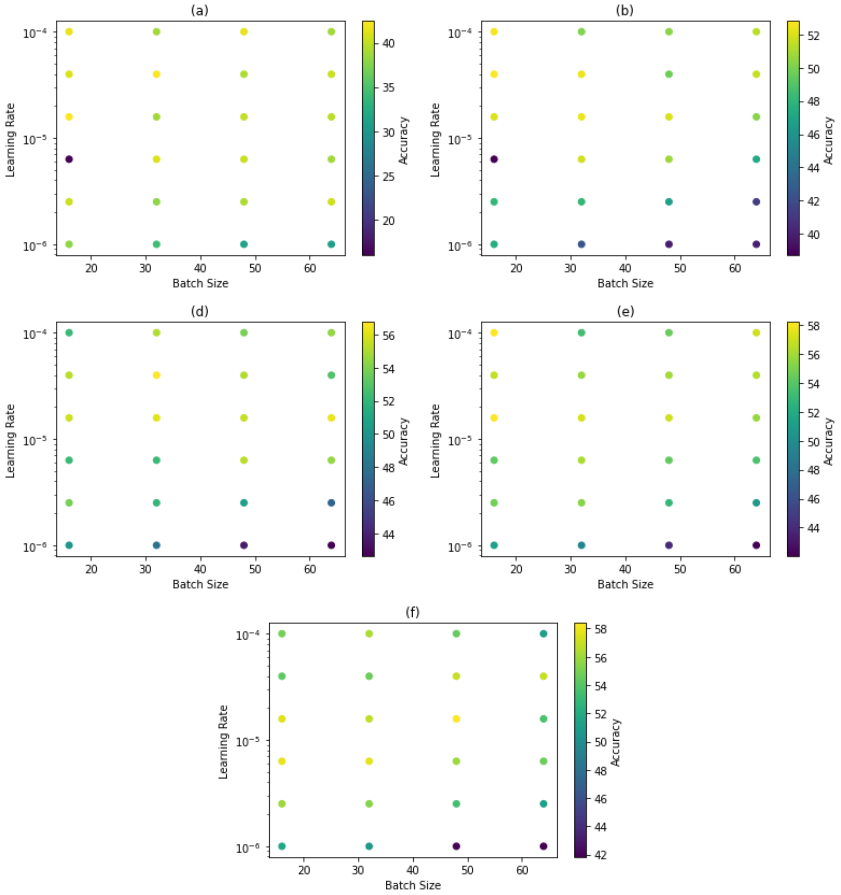


Figure 10: Accuracy% with respect to hyperparameter pairs, for the different subsets of frozen layers, when training for 150 epochs.

In general, we observed an increasing pattern in accuracy scores as we decreased the number of frozen layers in the pre-trained model. However, this observation does not hold across all cases. By calculating the average accuracy for each model across all hyperparameter combinations, we obtained the following results: 38.94% for model (a), 48.88% for model (b), 52.64% for model (d), 54.06% for model (e), and 53.87% for model (f).

These average accuracy values possibly indicate that there is a modest performance improvement when we utilize and thus freeze the pre-trained batch normalization layer in (e), as opposed to retraining it in (f). This suggests that there might be some transferable knowledge from the original English SER task to the new Greek SER task. Yet, the average accuracy likely plays a role of minor importance to the usefulness of the classifier's pre-learned patterns, and we might need more proof to support such an argument. After all, we are searching for the highest performance and the optimal hyperparameter vectors when tuning, and we still have not received significant results.

It's worth noting that conducting training for 150 epochs possibly provides plenty of time for the models to stabilize and reach their peak performance levels. This naturally raises the question of what might occur if we were to reduce the training duration. A slightly different approach that involved only 5 epochs of training was followed. The models trained and tested were (c), (e), and (g).

We aimed to investigate whether any of the pre-trained layers possessed initial transferable knowledge that could contribute to the classification ability of the network and to examine if any sudden changes in the content of the network, such as freezing or unfreezing specific layers, would have an impact on the classification ability.

By computing the average accuracy percentages for models (c), (e), and (g) across every tested pair of learning rate and batch size values, we obtained the following scores: 36.95% for (c), 36.59% for (e), and 29.96% for (g). These results further support the observation that the pre-trained SquirtleNET model possesses some degree of knowledge applicable to the Greek SER task, since the more the frozen layers, the higher the accuracy.

Nevertheless, achieving an average accuracy of 30-35% is far from satisfying.

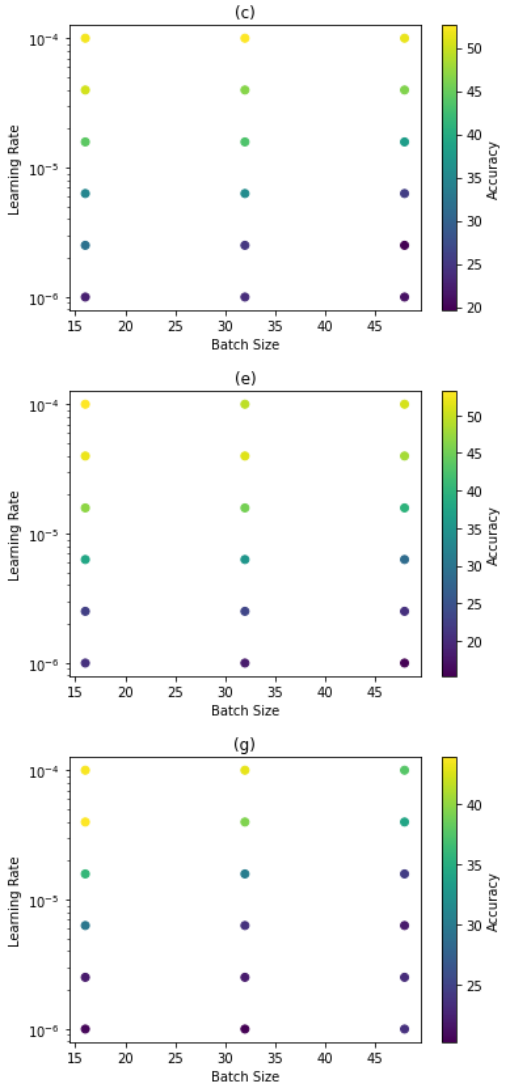


Figure 11: Accuracy% with respect to hyperparameter pairs, for the different subsets of frozen layers, when only training for 5 epochs.

To validate the applicability of transfer learning in this context, **we conducted a final small grid search using a completely untrained SquirtleNET model**. The model was trained for 150 epochs and evaluated using the 5-fold cross-validation method, in three instances, all with a batch size of 32 and learning rates of 1e-6, 1e-5, and 1e-4, respectively. The accuracy scores obtained from the 5-fold cross-validation process were 57.97%, 55.90%, and 59.84%, respectively. If compared with previous results, those scores indicate that training an untrained model yields better overall performance compared to training a pre-trained model. These findings suggest that the pre-trained SquirtleNET model's integrated knowledge is most probably insufficient.

Finally, we took advantage of the remaining testing data to test the pre-trained model with the highest accuracy so far and then evaluated it. The highest accuracy present in our experiments is achieved for (Learning Rate, Batch Size) = (2.68e-5, 48) and for model (g), which has zero frozen layers (every layer is re-trained). We proceeded to train the pre-trained English SER model, on the entire Greek training set for a total of 2,000 epochs, and test it after every epoch, storing the results in order to plot the learning curves. We monitored the performance throughout the training process and updated the best model every 5 epochs. If the testing loss was smaller than the previous minimum testing loss, we set the current loss as the new minimum and stored/overwrote the model accordingly, following the same procedure as before. The specific updates are provided in Table 6.

After training the pre-trained model with no frozen layers for 615 epochs on Greek speech data, we achieved a promising level of Accuracy and Macro F1-score in recognizing emotions. The model achieved a total Accuracy of **66.12%** and a Macro F1-score of **65.80%** when distinguishing between the five different emotion classes in the Greek speech data of the Acted Emotional Speech Dynamic Database.

While the classifier's performance is not bad, considering it has only been trained on a very small new dataset, we were hoping for higher performance results by leveraging the existing knowledge of a pre-trained model on a SER task.

Table 6: Every Update in Final Greek SER model's Training.

Epoch	Test Loss	Accuracy	F1-Score
0	1.60715	0.1736	0.0592
5	1.48266	0.3967	0.3477
10	1.42297	0.4545	0.4313
15	1.42106	0.4711	0.4680
20	1.41261	0.4711	0.4549
25	1.37832	0.5207	0.5182
30	1.37434	0.5372	0.5263
40	1.35561	0.5372	0.5416
55	1.34463	0.5537	0.5570
65	1.32099	0.5785	0.5717
75	1.31576	0.5950	0.5970
90	1.31144	0.5785	0.5797
95	1.29611	0.5950	0.5932
110	1.29428	0.6116	0.6063
125	1.29117	0.5950	0.5918
165	1.28960	0.5950	0.5943
170	1.28484	0.5950	0.5988
350	1.27992	0.6033	0.6036
355	1.27898	0.6198	0.6203
360	1.27496	0.6281	0.6230
375	1.27183	0.6198	0.6200
385	1.27140	0.6198	0.6211
415	1.25427	0.6364	0.6356
605	1.25305	0.6446	0.6465
615	1.23522	0.6612	0.6580

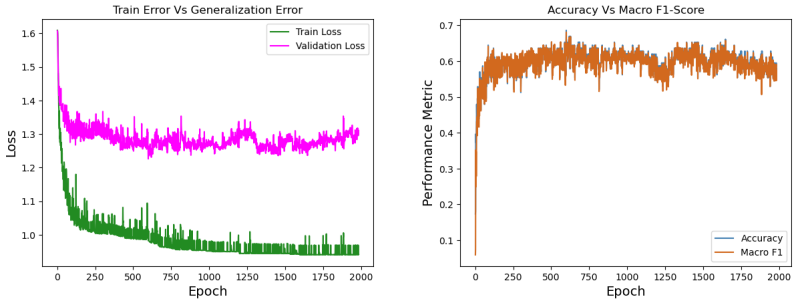


Figure 12: Final Greek SER model's Learning Curves.

Conclusions & Future Work

Mel Spectrograms have shown the highest performance among the English-SER 2D spectral features, while STFT Chromagrams have also demonstrated relatively high scores. On the other hand, the MFCC features have not contributed significantly to building a capable classifier.

Regarding image size, experiments with 50x50, 60x60, and 80x80 images have yielded promising results. The 30x30 images appeared to lack essential information, resulting in reduced classification performance.

The optimal CNN architecture is SquirtleNET, delivering superior results, closely followed by CharmaNET, which also provided satisfying outcomes. Unfortunately, BulbaNET did not meet our expectations. It should be noted that these architectures could be furtherly altered by adding more layers. Consequently, the research could be expanded on wider CNN architectures.

In general, the Speech Emotion Recognition task was shifted into an Image Classification task. We suggest that those interested in replicating our approach should utilize Mel Spectrogram features, specifically of size 60x60. Larger image sizes do not necessarily lead to higher accuracy.

For the application of Transfer Learning to the Greek language, it was discovered that for longer number of training epochs, enough for the model's performance metrics to converge, more freedom is required and consequently fewer frozen layers achieve higher performance. Conversely, when the training duration is too short and the model is far from convergence, better performance is obtained with more frozen layers, possibly suggesting that some degree of knowledge is embedded within the structure of these pre-trained layers.

However, training a fully defined SquirtleNET architecture for 150 epochs revealed that leveraging a pre-trained model is not necessary to achieve high performance. Therefore, the task of English SER and the larger amount of English data did not significantly aid in the new task of Greek SER.

There are several pivotal factors to take into account when explaining the challenges of applying transfer learning:

1. The pre-trained model was already trained on 7 instead of 5 classes which were required for the Greek SER task, with a significant focus on identifying neutral class patterns, which are not present in the AESDD dataset. As

a result, some valuable knowledge may have been wasted.

2. English speakers may express emotions differently from Greek speakers. Due to the vast linguistic differences between the two languages, certain emotions might be conveyed using a range of pitches in one language but not often utilized in the other. This aspect requires further research and a comprehensive study.

3. When applying transfer learning, additional parameter tuning may be necessary. Secondary parameters, such as output channels and kernel sizes, could be adjusted to improve classification results on the new task.

4. The element of acted emotions by actors in the dataset may have distorted the true emotional context of the audio speeches, causing a loss of similarity between the two tasks.

5. The timing and duration of the audio samples in the dataset might affect performance. Emotions can be expressed differently depending on the length of the speech segment and the specific moments within the speech. Nevertheless, duration also affects the information stored inside a Mel Spectrogram image, which is further altered after the bilinear image interpolation.

6. The Mel-Spectrogram feature may not effectively capture the shared information between the two language sets of speeches. The two tasks could exhibit similarities that are not adequately represented by this particular spectral feature.

As for future work, the defined SquirtleNET model can be trained with 60x60 Mel Spectrograms extracted from speech, and be utilized in several datasets that follow the Ekman's model. Experiments could also be made for more than 7 emotional classes, while a regression approach could be adopted for continuous emotion vector-values.

The research could be extended to wider CNN structures. Some other suggestions for future research include; experiments with various pre-processing techniques, executing parameter tuning with more parameters included (eg. activation functions, algorithmic arguments), or applying transfer learning to more languages and testing if we can leverage knowledge.

Keep in mind that supervised learning relies on human labeling, which can be prone to errors, so this approach may encounter limitations. The recognition of emotions in speech is a complex process, and even humans may not always execute it accurately. Alternative methods such as unsupervised learning could be implemented and combined with our existing model, to provide better and safer results.

Abbreviations

SER Speech Emotion Recognition

CNN Convolutional Neural Network

ES Early Stopping

COC Convolutional Output Channels

CKS Convolutional Kernel Sizes

PKS Pooling Kernel Sizes

PS Pooling Strides

BNM Batch-Normalization Momentum

DP Dropout Probability

FCOC Fully-Connected Output Channels

BS Batch-Size

LR Learning Rate

ET Execution Time

EP Epochs (Needed)

Bibliography

[1] Houwei Cao, David G. Cooper, Michael K. Keutmann, Ruben C. Gur, Ani Nenkova, and Ragini Verma. Crema-d: Crowd-sourced emotional multimodal actors dataset. *IEEE Transactions on Affective Computing*, 5(4):377–390, 2014.

[2] Kate Dupuis and M. Kathleen Pichora-Fuller. Tess Dataset, 2010.

[3] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. Adaptive computation and machine learning. The MIT Press.

[4] Earl J. Kirkland. *Advanced Computing in Electron Microscopy*. Springer US.

[5] Steven R. Livingstone and Frank A. Russo. The ryerson audio-visual database of emotional speech and song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in north american english. 13(5):e0196391.

[6] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. Librosa.

[7] Sanaul Haq Peter J. Jackson. The Surrey Audio-Visual Expressed Emotion (SAVEE) Database. URL: http://personal.ee.surrey.ac.uk/Personal/P.Jackson/SAVEE/.

[8] Python 3.10. https://www.python.org/downloads/release/python-310/, 2021. Python Software Foundation.

[9] Georgios Skoulidis. Speech emotion recognition using deep learning, 2023.

[10] N. Vryzas, R. Kotsakis, A. Liatsou, C. A. Dimoulas, and G. Kalliris. Speech emotion recognition for performance interaction. *Journal of the Audio Engineering Society*, 66(6):457–467, 2018.

[11] N. Vryzas, M. Matsiola, R. Kotsakis, C. Dimoulas, and G. Kalliris. Subjective Evaluation of a Speech Emotion Recognition Interaction Framework. In *Proceedings of the Audio Mostly 2018 on Sound in Immersion and Emotion*, page 34. ACM, September 2018.