The Language of Sounds: Spoken Language Identification via Deep Neural Networks

Nurullah Eren Acar
Artificial Intelligence and Data Engineering
Istanbul Technical University
acarn22@itu.edu.tr
150220310

Ömer Faruk Satık
Artificial Intelligence and Data Engineering
Istanbul Technical University
satik21@itu.edu.tr
150210330

Abstract—**Spoken Language Identification (SLI) is essential for determining the language of a given audio segment. Deep learning methods, including Convolutional Neural Networks, Long Short-Term Memory Networks, and Transformers, have significantly advanced this task. CNNs extract local acoustic features, LSTMs model temporal dependencies, and Transformers capture global patterns with self-attention. This paper examines these architectures' roles in SLI, focusing on their effectiveness and contributions to improving language identification systems.**

Index Terms—**spoken language identification, convolutional neural networks, long short-term memory, transformers, deep learning, acoustic features, temporal dependencies, self-attention mechanism.**

I. Introduction

Spoken Language Identification (SLI) is a classification problem that aims to distinguish the languages from human speech. Language identification plays an important role in a lot of areas such as multilingual support systems, automatic speech recognition systems and real-time translation. This study aims to compare the performance of deep learning architectures such as Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM) and Transformer Models in order to identify selected four languages (Arabic, French, Spanish and Turkish). Each model is used to differentiate between these languages by learning from the spectral features of the speech data.

II. Problem Description

The problem of Spoken Language Identification requires the analysis of sound features from different languages. Although there are significant differences between the tonal phonetic and acoustic features of languages, these differences may not be easy to distinguish in languages from similar language families. Moreover, factors such as accent, noise and speaker dependent features can reduce the performance of the classifier.

In this study, to address these challenges, voice data was converted into mel-spectrograms and used as input to feed deep learning models. This method aims to increase the language identification performance of the models by providing a two-dimensional data that can show frequency, time and amplitude information at the same time.

III. Real World Application

Spoken Language Identification can be seen being used in following fields:

- **Customer Support Services**: Directing incoming calls to the correct assistants based on the language being used.
- **Real-Time Translation Systems**: Automatic language identification and translation in multilingual assistants and meeting systems.
- **Forensic Computer Science**: Language detection by analyzing voice recordings and using them as evidence.
- **Content Categorization**: Language-based labeling and recommendation of content on video and audio platforms.

These applications increase the importance of deep learning based language identification systems in real life scenarios.

IV. Related Work

Spoken Language Identification has improved significantly with advancements in deep learning field. Early methods relied heavily on handcrafted features and statistical models like Gaussian Mixture Models (GMMs) and Hidden Markov Models (HMMs), which often struggled with scalability [1].

Deep learning approaches, particularly Convolutional Neural Networks (CNNs), have shown proper success in processing spectrogram representations of speech data [4]. Aldeneh et al. [5] improved performance by combining CNNs with Long Short-Term Memory (LSTM) networks in a architecture which runs them in parallel, effectively capturing both spatial and temporal features.

Transformer models have recently showed up as a powerful tool for these tasks. Nie et al. [6] showed a BERT-based model (BERT-LID) to improve language identification performance, especially for short speech samples. To add more, Bartley et al. [7] showed that the self-supervised learning models could learn language-specific features while completing pretraining.

Finally, Eyceoz et al. [8] introduced an open set SLI framework using a Time-Delay Neural Network (TDNN) and a novel multilingual dataset, enabling robust language classification of both known and unknown ones.

In summary, improvements from CNNs to Transformers and self-supervised models have significantly improved SLI systems, making them robust and easy to implement in real world scenarios.

V. Data

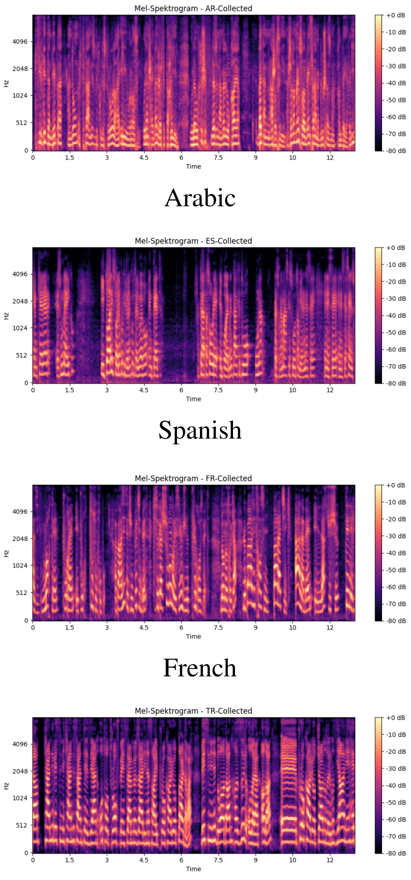
A. Dataset

In order for deep learning models to perform well, the data used in training is of great importance. The quality, quantity and diversity of the data are the main metrics that are taken into consideration. In addition, not violating the Personal Data Protection Law (KVKK) and the copyrights of individuals while collecting data are issues that need to be taken into consideration in such projects. For this reason, researching and collecting data is one of the most important steps of this project and the work was carried out with great care at this stage.

The data set was obtained by collecting and combining 2 different sources. As a result of long research, the Media Speech data set, which constitutes the majority of the data, was discovered on the OpenSLR website [1]. MediaSpeech is a dataset of French, Arabic, Turkish and Spanish media speech built with the purpose of testing Automated Speech Recognition systems performance. The dataset contains 10 hours of speech for each language provided. The dataset consists of short speech segments automatically extracted from media videos available on YouTube. The dataset consists of short speech recordings in the form of FLAC files, which are automatically extracted from media videos available on YouTube. Other data contributing to the project was obtained by the project team after watching hours of videos in each of the 4 languages on YouTube. All videos identified for processing are licensed under the Creative Commons (CC BY) license. This license enables reusers to distribute, remix, adapt, and build upon the material in any medium or format, so long as attribution is given to the creator [2]. These videos present the conversations of people in a natural flow as content. A python script was written to download the relevant videos in wav format. This data collection includes a total of approximately 6 hours of recorded speech. Overall, a dataset consisting of 46 hours of recorded speech before pre-processing was created.

B. Pre-processing

First, we converted the audio files into one-dimensional arrays with the librosa library to make them processable in the Python environment. While doing this, it was necessary to assign the sample rate (number of samples per second) to a common value in order to ensure dimensional consistency because the recording frequency of each audio file was different. In order to feed the audio data to deep learning models, it was first necessary to convert it into representations in the frequency, time and amplitude domains called spectrograms. The maximum frequency that these spectrograms can exhibit is half the sample rate value. Since the frequency of human sounds is low, it was decided to assign the sample rate value to a relatively small value such as 16 kHz. In other words, one-dimensional numpy arrays representing the audio files in the amplitude domain with 16000 samples per second were obtained. Afterwards, it was necessary to clean up the recordings by deleting the silent or non-speech parts because there was a silent part at the beginning and end of all the sounds in the Media Speech dataset. In addition, there were parts in the recordings scraped from YouTube that did not contain speech at the beginning and end of the videos, such as intro music. We did data cleaning to avoid using misleading data. To ensure a common dimension in all data to be given to the model, clips were created by cutting 13-second segments from the sounds. Finally, the arrays in the amplitude domain were converted to mel-spectrograms with the librosa library and converted to 2-dimensional numpy arrays that provide information about frequency, time and amplitude properties. The reason why mel spectrograms are preferred is that the mel scale provides higher resolution at low frequencies, allowing people to better distinguish differences at low frequencies. Since people perceive differences in high frequency ranges with less sensitivity, the mel scale provides less resolution at high frequencies [3]. Thus, data representing the features that the model needs to learn were obtained. 1 spectrogram example for each language can be seen in Figure 1.



Arabic

Spanish

French

Turkish

Fig. 1. Sample Spectrograms

In the spectrograms, the vertical axis shows the frequency, the horizontal axis shows the time and the color shows the amplitude at that time step for each frequency value. The deep learning models to be designed in this project are intended to learn the intonation of each language by learning the time and frequency dependent fluctuations in spectrograms.

C. Data Augmentation

Three different data augmentation techniques were applied to increase the diversity and quantity of data.

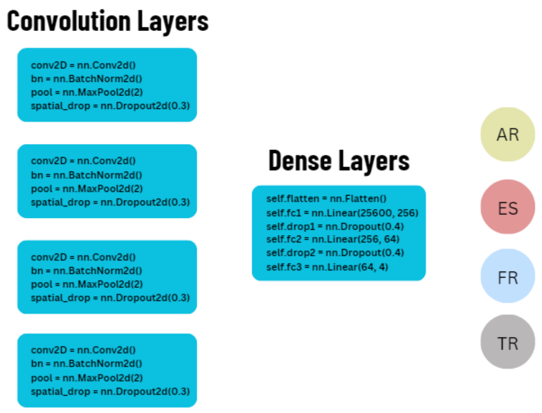
- Time Masking: Causes random time intervals in the spectrogram to be silenced.
- Pitch Shifting: It is the process of changing the frequency of the sound. The sound can be made higher (higher pitch) or lower (lower pitch).
- Adding Noise: It is the process of adding random noise to the audio. It is ideal for training a model that is more robust to background speech or environmental noise.

VI. Methodology

A solution was produced for Spoken Language Identification with 3 different deep learning architectures: Convolutional Neural Networks (CNN), Long Short Term Memory (LSTM) and Transformer.

A. Convolutional Neural Networks

In the CNN model, spectrograms showing frequency and amplitude features for 13-second time periods were considered as a whole. In order to reach the best performing model with this method, 3 different network structures were tried. The structure of the final CNN model is as seen in Figure 2.



```
conv2D = nn.Conv2d()
bn = nn.BatchNorm2d()
pool = nn.MaxPool2d(2)
spatial_drop = nn.Dropout2d(0.3)
```

```
conv2D = nn.Conv2d()
bn = nn.BatchNorm2d()
pool = nn.MaxPool2d(2)
spatial_drop = nn.Dropout2d(0.3)
```

```
conv2D = nn.Conv2d()
bn = nn.BatchNorm2d()
pool = nn.MaxPool2d(2)
spatial_drop = nn.Dropout2d(0.3)
```

```
conv2D = nn.Conv2d()
bn = nn.BatchNorm2d()
pool = nn.MaxPool2d(2)
spatial_drop = nn.Dropout2d(0.3)
```

```
self.flatten = nn.Flatten()
self.fc1 = nn.Linear(25600, 256)
self.drop1 = nn.Dropout(0.4)
self.fc2 = nn.Linear(256, 64)
self.drop2 = nn.Dropout(0.4)
self.fc3 = nn.Linear(64, 4)
```

Fig. 2. CNN Network

In feature extraction from audio spectrograms, convolution layers were used to understand frequency-time features. Batch Normalization was included in the network in order to ensure that the model learns faster and more stably by normalizing the activations. Maximum Pooling was used to reduce the computational load and preserve important information. Dropout layers were also used to prevent overfitting. In the output layer, Softmax activation function helped for the classification task. Cross Entropy, which works ideally with Softmax, was preferred as the loss function and the loss was optimized with Adam Optimizer. The advantages of this optimization method in avoiding local minimums and adaptive learning were utilized.

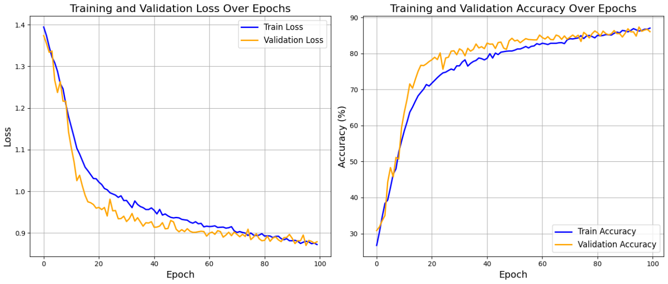
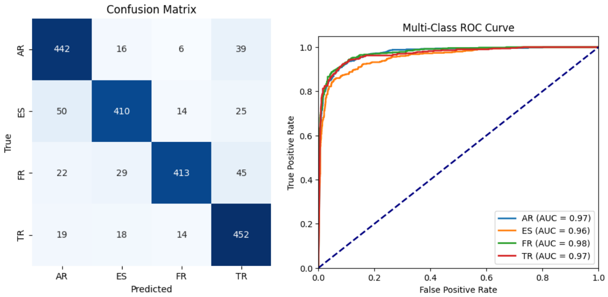


Fig. 3. CNN loss-accuracy plot

When the plot in Figure 3 is examined, it is seen that the train loss and validation loss decrease regularly. Both loss curves reach a fixed point at the end, which shows that the model has completed learning and overfitting has been avoided. The parallel course of the accuracy curves and the fact that they end at a very close point supports this thesis.



	AR	ES	FR	TR
AR	442	16	6	39
ES	50	410	14	25
FR	22	29	413	45
TR	19	18	14	452

Confusion matrix

ROC curve

Fig. 4. CNN test results

The diagonal elements in the confusion matrix in Figure 4 represent correctly classified examples and have quite high values. Misclassifications are few, but some errors are particularly striking. Spanish is confused with Arabic many times (50 errors). Also, Arabic is confused 39 times, and French is confused 45 times with Turkish. On the other hand, to explain the other evaluation metric plot, the ROC (Receiver Operating Characteristic) curve visualizes how well the model can distinguish between positive and negative classes, while the AUC (Area Under the Curve) score summarizes this performance in a single number.In the plot, the AUC values for all classes are quite high and the ROC curves are quite close to the curve of an ideal model (upper left corner). The AUC values between 0.96 and 0.98 show that the model exhibits a discriminatory performance for each class. The curve for the Spanish language shows that it is relatively more difficult for this model to recognize this language. As a result, the 85% accuracy value in the test data shows that the model works quite successfully.

*1) **5x2 Cross Validation**:* This method tests the performance of a model by randomly splitting the dataset into two equal parts (e.g. 50% training, 50% testing) and repeating this process twice in each split (roles reversed) for 5 different random splits. A total of 10 test results are obtained and these results are analyzed with metrics such as mean and variance to evaluate the generalization performance of the model. When the robustness of the CNN model was tested with this method, the outputs in Table I were obtained:

TABLE I
5X2 CROSS-VALIDATION RESULTS

Round	Fold	Accuracy
1	1	0.8331
	2	0.8518
2	1	0.8439
	2	0.8552
3	1	0.8413
	2	0.8496
4	1	0.8381
	2	0.8478
5	1	0.8367
	2	0.8367

These results showed that the model exhibited robust performance even when trained with different parts of the data.

*2) **Optimizers**:* In order to evaluate the model's dependency on optimization methods, the same model was trained with 2 other optimizers. Training with Stochastic Gradient Descent (SGD) and Root Mean Squared Propagation (RMSprop) approaches was tried many times with different parameters, but the level reached by the Adam optimizer could not be reached. In the outputs obtained with the two optimization methods in Figure 5 and Figure 6, although there is a general approach to the minimum, oscillations are noticeable. In addition, the test accuracy was at most 82% with both methods.

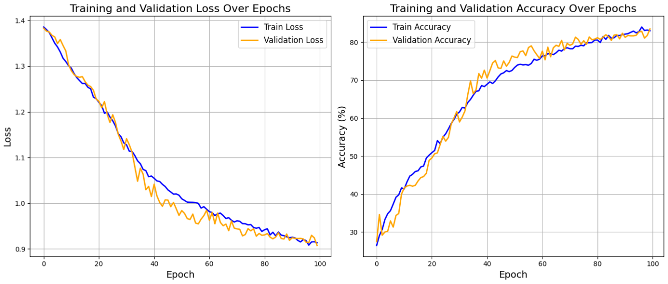


Fig. 5. CNN with SGD optimizer

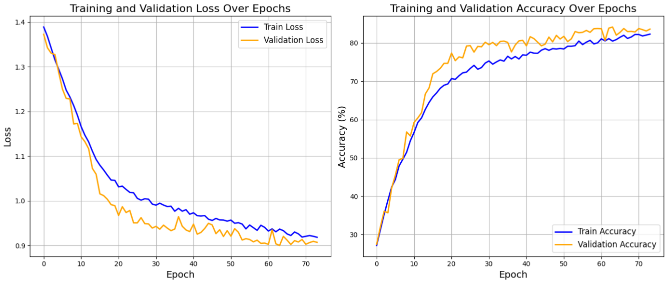
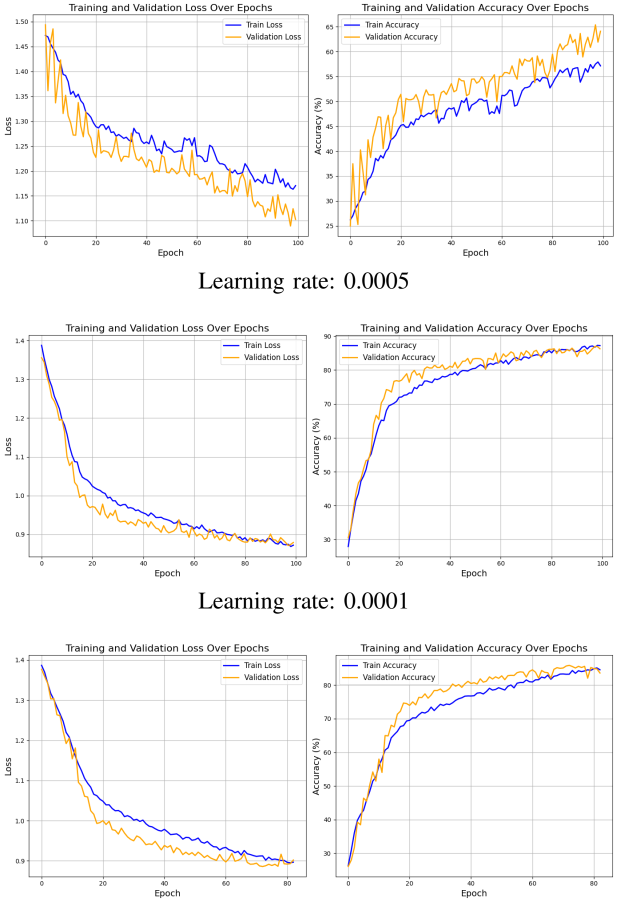


Fig. 6. CNN with RMSprop optimizer

*3) **Learning Rates**:* In order to determine which learning rate value the optimization method works best at, 5 different learning rate values were tested. 2 of them were not included in the report because they performed very poorly. Figure 7 shows the performance of the model at different learning rate values.



Learning rate: 0.0005

Learning rate: 0.0001

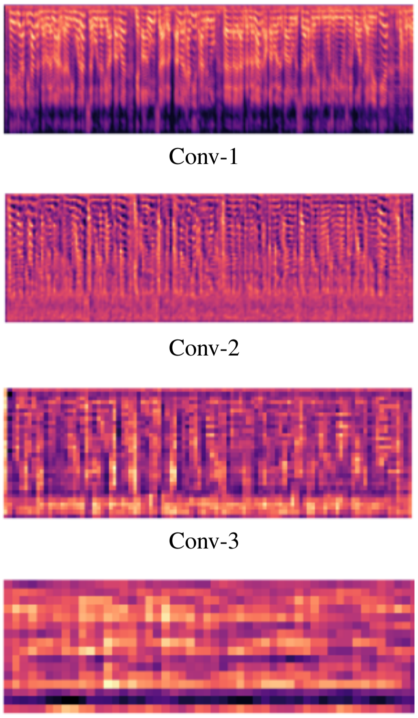
Learning rate: 0.00005

Fig. 7. Performance with different learning rates

Looking at both the structure of the curve and the maximum performance value it reaches, it is obvious that 0.0001 is the most suitable learning rate value for the Adam optimizer in training this model.

*4) **Feature Representation**:* In order to understand which features the convolutional layers in the CNN architecture have learned, the filters of each layer were visualized. In Figure 8, one of the filters belonging to each convolutional layer is shown.

The first layer detects basic patterns and low-level features (edges, frequency textures) in the input data, while the middle layers combine these features to extract more complex and correlated patterns. The last layer learns high-level representations that are abstract. In the images, it is seen that as the layer gets deeper, the learned features become more complex and the frequency-based patterns reach a more abstract level over time.


Conv-1

Conv-2

Conv-3

Conv-4

Fig. 8. Feature representation of CNN

*5) **Ablation Study**:* Three different ablation techniques were applied to measure the contribution of the choices made in the design and training of the model to the performance. The first of these was to measure the contribution of the amount of data to the performance by training the model with one fourth of the data. The model was not affected by this change at all and maintained its accuracy of 85%. In another approach, the performance was tested by removing the last convolution layer of the model. Here, the performance of the model showed a 4% decrease. Finally, when 2 of the 3 fully connected layers of the model were disabled, almost no decrease in performance was observed.

B. Long Short-Term Memory Networks (LSTM)

In this project, the LSTM model was used to analyze time-dependent features. Data consisting of 407 time steps and 128 features were used as input to the model. The hidden dimension was determined as 64. In order to obtain the best performance, single-layer and two-layer LSTM architectures were tested. The structure of the final model is shown in Figure 9. In this model, as in the CNN model, Softmax Classifier, Cross Entropy Loss and Adam Optimizer were used.

Training and validation losses decreased steadily and no signs of overfitting were observed. The accuracy value of the model on the test set was recorded as 92%. Figure 10 shows the loss and accuracy curves of the training and validation processes.

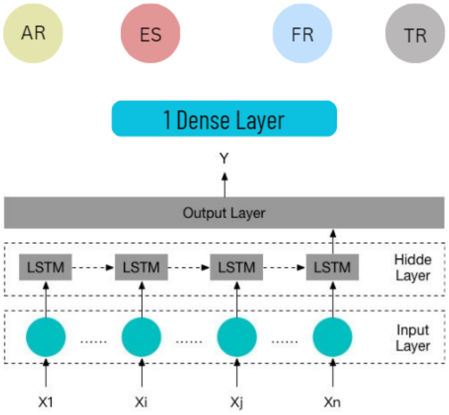


Fig. 9. LSTM network

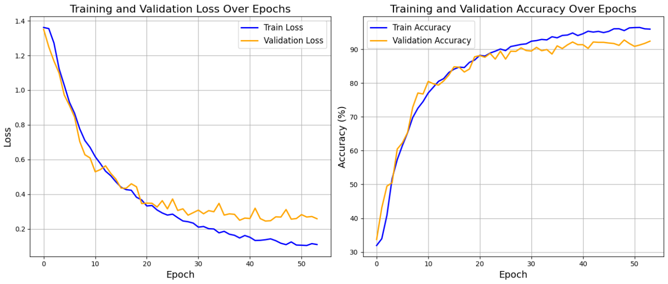
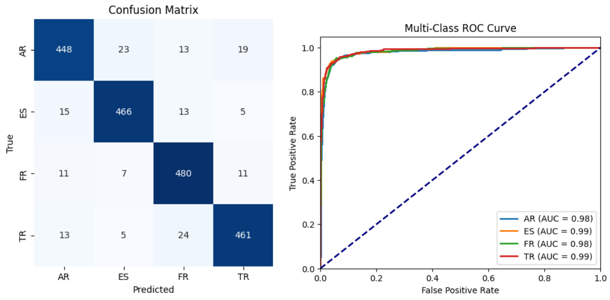


Fig. 10. LSTM loss-accuracy plot

When the test results in Figure 11 are examined, it is seen that the model has shown good performance in terms of both correct classification rates and AUC values. The errors encountered in the results of the CNN model seem to have decreased.



Confusion matrix

ROC curve

Fig. 11. LSTM test results

*1) **5x2 Cross Validation**:* When Table II is examined, although the model performance does not change much between the iterations of the cross validation algorithm, it is seen that the accuracy decreases from 92% to an average value of 88%. The reason for this is that while normally more than half of a data set is used for training, half of the data is used as training in the 5x2 cross validation method. The data loss here may have caused this performance decrease. However, the close performance between the folds still proves the generalization ability of the model.

TABLE II
5x2 Cross-Validation Results

Round	Fold	Accuracy
1	1	0.8699
	2	0.8953
2	1	0.8963
	2	0.8582
3	1	0.8788
	2	0.8725
4	1	0.8822
	2	0.8917
5	1	0.8840
	2	0.8737

2) ***Optimizers:*** Model training was also tried with SGD and RMSprop optimization methods and the results shown in Figure 12 and Figure 13 were obtained. While the model could not approach the minimum with the SGD method, the test accuracy was recorded as 90%, although there were fluctuations in the RMSprop method.

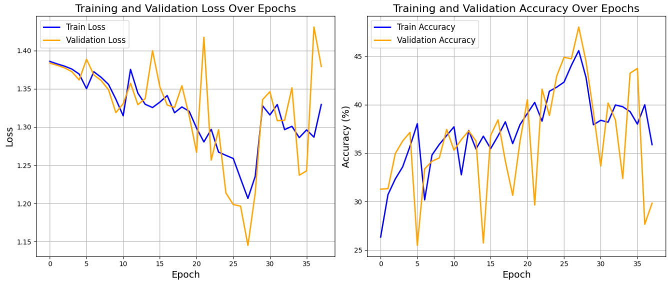


Fig. 12. LSTM with SGD optimizer

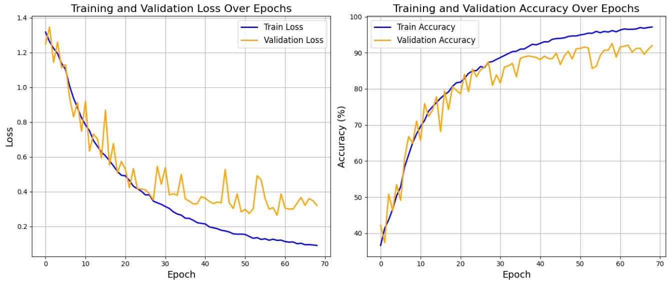
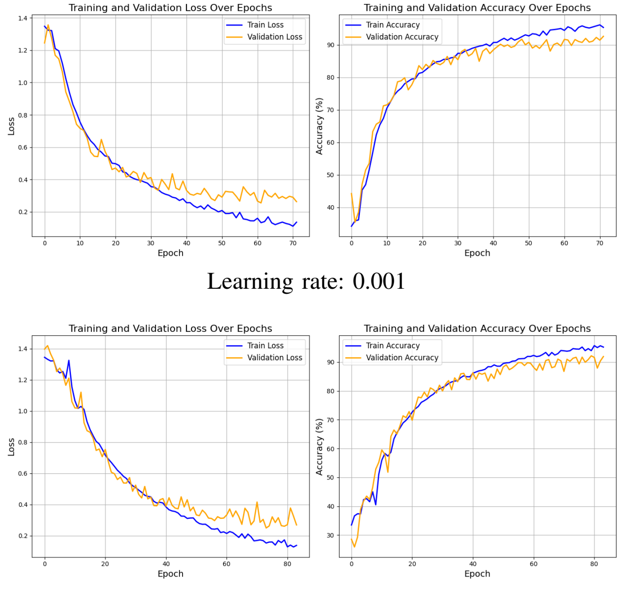


Fig. 13. LSTM with RMSprop optimizer

3) ***Learning Rates:*** The model was tested with 5 different learning rates and 2 of them showed the best performance with 90% accuracy as seen in Figure 14.

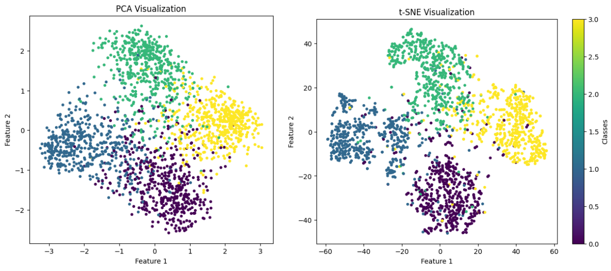


Learning rate: 0.001

Learning rate: 0.0005

Fig. 14. Performance with different learning rates

4) ***Feature Representation:*** The hidden state of the last cell of the LSTM model was taken as a feature. Since the size of this feature was too large to be visualized, Principal Component Analysis (PCA) and T-Distributed Stochastic Neighbor Embedding (t-SNE) methods were used to reduce the vector to 2 dimensions. In Figure 15 it is seen that lstm successfully separates classes in 2-dimensional space.



PCA Visualization

t-SNE Visualization

Fig. 15. Feature representation of LSTM

5) ***Ablation Study:*** In the ablation study of the LSTM model, training was done using one fourth of the data and the accuracy value of the model decreased to 70%. This shows that the size of the data is very important in this LSTM model.

C. Transformer Model

The Transformer model, known for its strength in dealing with sequential data, was employed to be used while classifying spectrogram representations of speech data. The architecture consists of encoder layers utilizing multi-head self-attention and feed-forward networks. Positional encodings were added to preserve the input sequence structure. The final outputs from the encoder were passed through a classification head using softmax activation to predict language labels.

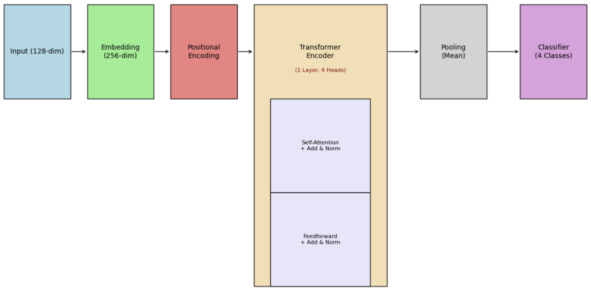


Fig. 16. Transformer model structure

The training process showed consistent learning, as shown in the loss and accuracy plots. The validation loss followed the training loss closely, indicating slight overfitting only.

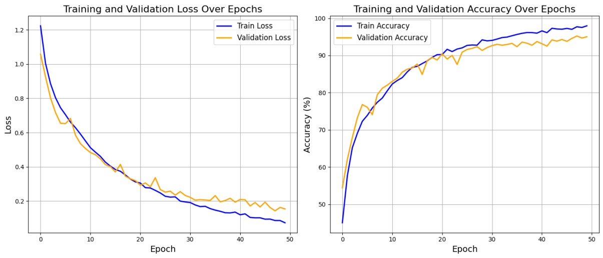


Fig. 17. Training and validation loss and accuracy over epochs

The evaluation results highlighted strong classification performance. The confusion matrix indicates effective language classification, though minor wrong classification situations were observed, particularly between Spanish and Arabic. These errors are likely to occur due to overlapping features between these two languages.

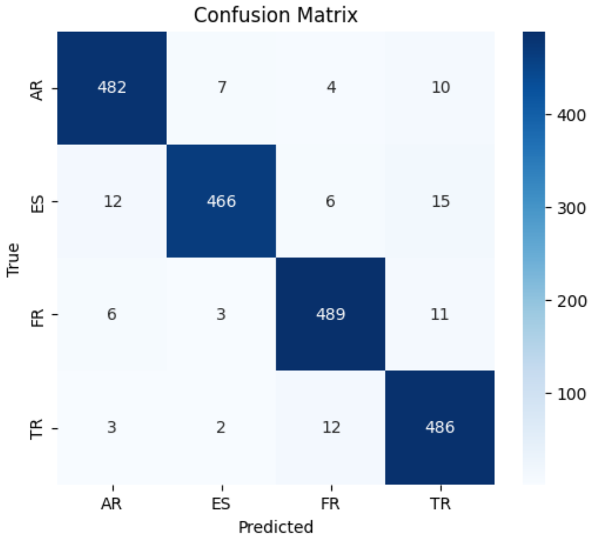


Fig. 18. Confusion matrix of the Transformer model

Overall, the Transformer model outperformed CNN and LSTM models in terms of accuracy and generalization, demonstrating its ability to learn both local and global dependencies effectively. However, its performance was more dependent on the quality and diversity of the dataset, showing the importance of well created training data.

D. Additional Evaluations

After the project was presented in this form, the performance of the model was found to be very high by the instructor. He suspected that the model might have learned the speech of the people rather than the characteristics of the languages, since the voices of the same people were found in the train and test sections of the model. For this reason, studies were conducted to test this situation in the short period of time after the presentation. Indeed, it was understood that there was a very high probability that samples very close to the train samples would be found in the test set due to the random splitting of the data set in the data pre-processing section. Since there were thousands of audio files in the Media Speech data set, which constituted 85% of the data, it was not possible to identify samples belonging to the same people. As a result, we completed this experiment using the data we collected from YouTube, which we could manipulate as we wanted. However, the problem encountered with this data was that the amount of data was insufficient.

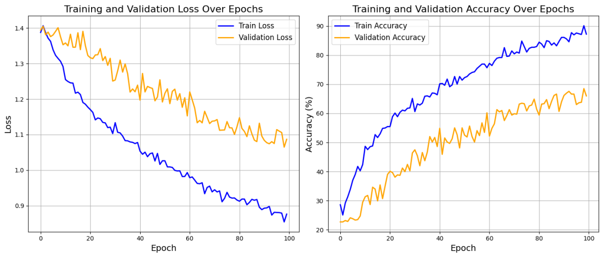


Fig. 19. CNN with correctly split YouTube data

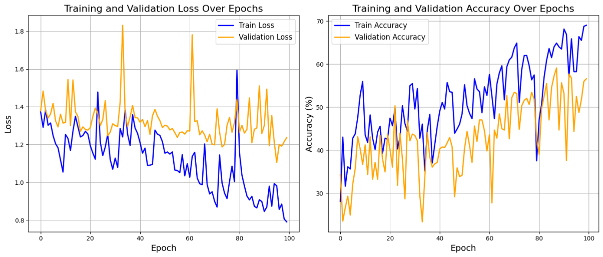


Fig. 20. LSTM with correctly split YouTube data

When Figure 19, Figure 20 and Figure 21 are examined, it is clearly seen that the learning process of the LSTM and Transformer models is not in the desired state. Although the CNN model does not provide very good performance, when looking at the big picture, it is seen that the train and validation curves are in a parallel improving trend. The 30% gap between train and validation accuracy reveals that the model cannot comprehensively learn the intonations of the languages. These results may mean that the time and frequency

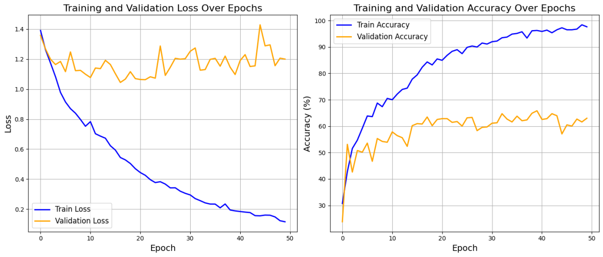


Fig. 21. Transformer with correctly split YouTube data

characteristics of the voice do not provide enough information to distinguish between languages, or this may simply be due to the inadequacy of the available data. In order to understand this, it is necessary to create a large-scale high-quality dataset consisting of independent examples in the train and inference sets.

VII. Conclusion

SLI is a challenging significant task in various domains, including multilingual systems, automatic speech recognition, and real-time translation. This study demonstrated a comprehensive comparison of three state-of-the-art deep learning architectures—CNN, LSTM, and Transformer models—for identifying four different languages: Arabic, French, Spanish, and Turkish. The models leveraged mel-spectrograms as input features, effectively capturing the temporal, frequency, and amplitude characteristics of the speech signals. Among the models, the Transformer model achieved the highest test accuracy of 95%, benefiting from its ability to learn temporal dependencies in sequential data. LSTM, another model that aims to learn temporal dependencies, showed 92% accuracy. The CNN model, with its ability to extract spatial features from spectrograms, also performed admirably with an accuracy of 85

However, after the project was presented, it was suspected that these models showed very successful performance and when the train and test distribution of the data set was examined, it was determined that there could be very similar examples in the two data sets. In order to eliminate this suspicion, the performance of the models decreased significantly in the study conducted with a small data set. In order to measure the real performance of the models, large data sets are needed where the train and inference data sets are completely independent from each other. The project can continue with a similar pipeline after the quality data collection step. After the model successfully performs the classification task using intonations on 4 languages, the language package can be expanded and a global solution can be offered. These efforts aim to bring SLI systems closer to real-world applicability, enabling robust and reliable language recognition.

References

[1] R. Kolobov et al., ”MediaSpeech: Multilanguage ASR Benchmark and Dataset,” arXiv preprint arXiv:2103.16193, 2021. [Online]. Available: https://www.openslr.org/108/.

[2] About CC Licenses, https://creativecommons.org/share-your-work/cclicenses/#:~:text=This%20license%20enables%20reusers%20to,be%20given%20to%20the%20creator.

[3] What Parameters Should We Pay Attention to When Processing Audio Data?, https://ece-akdagli.medium.com/ses-verisi-i%CC%87%C5%9Flerken-hangi-parametrelere-dikkat-etmeliyiz-c906801b58d3

[4] A. Liu et al., ”Language Identification with CNN,” GitHub Repository, 2019. [Online]. Available: https://github.com/alvayliu/Language-Identification-with-CNN

[5] M. A. Aldeneh et al., ”Spoken Language Identification Using Convolutional Recurrent Neural Networks,” Applied Sciences, vol. 12, no. 18, p. 9181, 2022. [Online]. Available: https://www.mdpi.com/2076-3417/12/18/9181

[6] Y. Nie et al., ”BERT-LID: Leveraging BERT to Improve Spoken Language Identification,” arXiv preprint arXiv:2203.00328, 2022. [Online]. Available: https://arxiv.org/abs/2203.00328

[7] T. M. Bartley et al., ”Accidental Learners: Spoken Language Identification in Multilingual Self-Supervised Models,” arXiv preprint arXiv:2211.05103, 2022. [Online]. Available: https://arxiv.org/abs/2211.05103

[8] M. Eyceoz et al., ”Robust Open-Set Spoken Language Identification and the CU MultiLang Dataset,” arXiv preprint arXiv:2308.14951, 2023. [Online]. Available: https://arxiv.org/abs/2308.14951