Arabic Dialect Aware TTS Model

Saif Eldin Mohamed 221001554

Mustafa Sultan 221000432

Seif Kassab 221002051

Ahmed Kamal 221001524

Omar Shalan 221000147

# Introduction

This project aims to create a model that can classify Arabic dialects from texts and deploy related TTS to the detected dialect to provide the user with more familiarity.

## Contribution

In this project, we created different models and compared them to find the most preferred approach. Some of our models are better than other models that use the same approach (such as Marbert V2 by 5%). We gained many skills such as web scraping, data imbalance handling, and Model Validation.

## Methodology

### Preprocessing

We first faced the problem of data imbalance as some of the classes were inequivalent. We tried to under-sample, but it backfired, so we decided to omit the low amount class as the model started to have a bias towards it. That was after integrating multiple datasets into one big dataset. After we did the following processes to ensure the data is clean: Cleaning, Remove emojis, Remove numbers, Remove any non-Arabic characters, Normalize Arabic words, Remove mentions, Remove Links. Remove duplicates, and then we used the tokenizer of the pretrained model; sentences less than 5 words were removed. In the end, each sentence was 14 words max, as this was the third quantile of the number of words per sentence

### Models

We used 4 models to compare and see which approach is more efficient: Marbert V2, Arabert V2, LSTM, and NN. The aim was to see how complicated models compare to more simplistic ones. In regards of the TTS, we could only deploy ready-made APIs as we lacked the information and computational power. We also used LoRA, LoRA (Low-Rank Adaptation) is a parameter-efficient fine-tuning technique that reduces the number of trainable parameters in large language models (LLMs) by only training a small number of new weights, while keeping the original model's weights frozen. This approach significantly reduces memory usage and computational costs compared to traditional full fine-tuning. Accuracies weren’t the best at first, but after enough fine-tuning, we managed to get them to a presentable manner.

## Results

The model results were the following: Marbert V2 (79%), Arabert V2 (62.7%), LSTM (66%), NN (66%)

A graph of different colored lines

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

## Challenges Faced

We couldn’t create a proper TTS as we lacked a lot of things, such as memory, computational power, time, and knowledge. Improving the LLMs was also difficult, as the runtimes were absurdly low.

## Conclusion

This model explored various approaches. And we tried to do our best in regards to TTS but could only go so far. With more time and resources this project could be proved on even further.