Predicting Arabic Dialects

Arabic is spoken across many countries, each with its own unique dialect. This presentation explores building a model to predict the dialect based on the text, shedding light on the rich diversity of the Arabic language.

by ahmed mostafa

Data Fetching

1

Connecting to Database

Used SQLite3 to connect to a database containing Arabic dialect data.

2

Extracting Tables

Fetched a list of all tables in the database and extracted data from each one.

3

Merging Data

Combined the data from all tables into a single DataFrame, removing any duplicate entries.

Data Pre-processing

Regex & Emoji Removal

Removed usernames, URLs, non-Arabic characters, punctuation, and emojis from the text.

Linguistic Cleaning

Removed tashkeel, elongation, repeated letters, and stop words to prepare the text for analysis.

Custom Tokenization

Developed a custom tokenizer to handle the unique characteristics of Arabic text.

Preprocessing Pipeline

Automated the data cleaning process using a reusable preprocessing pipeline.

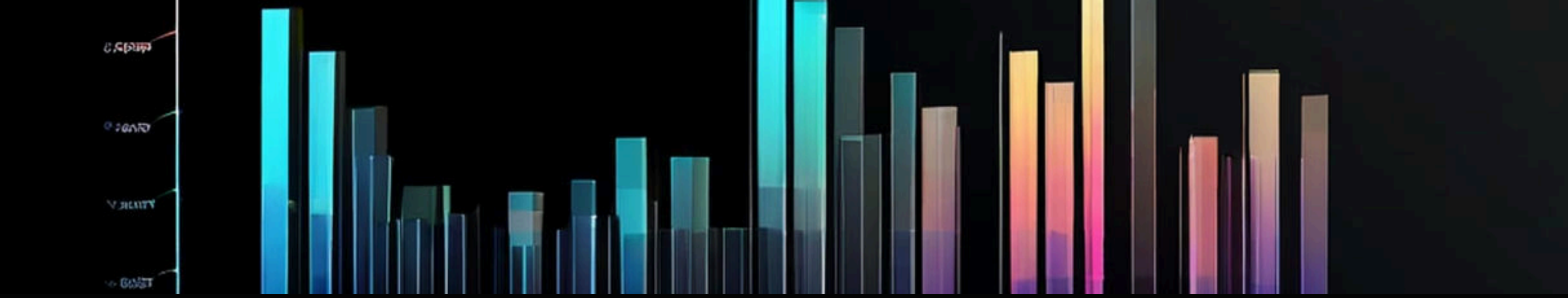
Model Architecture

Deep Learning

Utilized a Sequential model with an Embedding layer, Spatial Dropout, LSTM, and a Dense layer for the final classification.

Machine Learning

Implemented a Pipeline with TF-IDF vectorization and a Logistic Regression classifier for a more traditional approach.



Evaluation Metrics

1 Deep Learning

Used sparse categorical cross-entropy loss, Adam optimizer, and accuracy as the evaluation metric.

2 Machine Learning

Calculated test accuracy and generated a classification report to assess the model's performance.

Results

Deep Learning

Test Accuracy: 81.06%

Machine Learning

Test Accuracy: 81.77%

Deep Learning Testing

```
import numpy as np

arabic_texts = ["بسطا ازيك", "اشطا ياعم ", "أخيرًا كيف حالك"]


sequences = tokenizer.texts_to_sequences(arabic_texts)

padded_sequences = pad_sequences(sequences, maxlen=max_length, padding='post')

predictions = model.predict(padded_sequences)

original_labels = label_encoder.inverse_transform(predicted_classes)

print(original_labels)
predicted_classes = np.argmax(predictions, axis=1)

print(predicted_classes)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 62ms/step
['EG' 'EG' 'LB']
[0 0 1]
```

Machine Learning Testing

```
new_data = [
    "الناس دي بتنفخ قربه مقدوده بالدارجي كده البلد دي ماعندهاش حل",  # Example from SD
    "رانيا عقليتك متخلفه اولا الانسان يلي يحتاج اهل ليشوف مصلحته ضعيف شخصية",  # Example from LV
]
```

```
new_data_transformed = [' '.join(map(str, sentence.split())) for sentence in new_data]

# Make predictions
predictions = modelML.predict(new_data_transformed)

# Print the predictions
for text, prediction in zip(new_data, predictions):
    print(f"Text: {text}\nPredicted Dialect: {prediction}\n")
```

```
Text: الناس دي بتنفخ قربه مقدوده بالدارجي كده البلد دي ماعندهاش حل
Predicted Dialect: EG

Text: رانيا عقليتك متخلفه اولا الانسان يلي يحتاج اهل ليشوف مصلحته ضعيف شخصية
Predicted Dialect: LB
```

Challenges and Limitations

1 Time Constraints

Working on this project alone posed challenges in completing the tasks within the given timeframe.

2 Lone Efforts

Tackling the project without a team limited the ability to leverage diverse perspectives and expertise.

Conclusion

Resilience

Despite the challenges, the project was successfully completed by leveraging individual strengths and determination.

Future Directions

Expand Dataset

Collect more comprehensive data to improve model performance and cover a wider range of Arabic dialects.

Collaborative Effort

Engage with a team of experts to leverage diverse skills and experiences for enhanced model development.

Model Optimization

Explore advanced techniques, such as transfer learning or attention mechanisms, to further improve the model's accuracy.