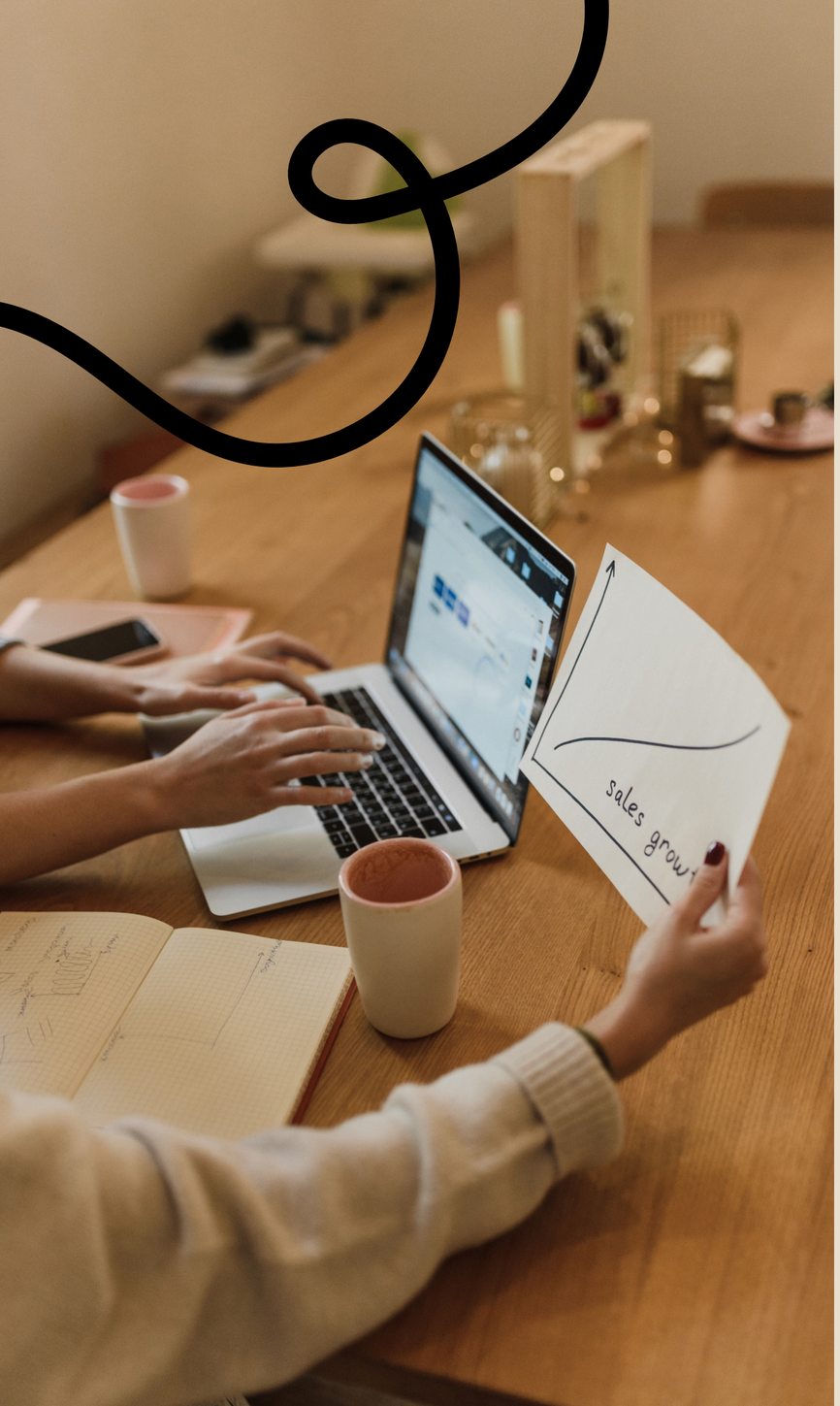
CODE-SWITCHING: DETECTION AND ANALYSIS

TABLE OF CONTENT

INTRODUCTION

CODE-SWITCHING REFERS TO THE LINGUISTIC PHENOMENON WHERE A SPEAKER ALTERNATES BETWEEN TWO OR MORE LANGUAGES WITHIN THE SAME CONVERSATION, SENTENCE, OR EVEN A SINGLE WORD. THIS PRACTICE IS COMMON IN MULTILINGUAL COMMUNITIES, WHERE SPEAKERS SWITCH BETWEEN LANGUAGES BASED ON SOCIAL, CULTURAL, OR EMOTIONAL CONTEXTS.

IN THE MOROCCAN CONTEXT, DARIJA (THE MOROCCAN ARABIC DIALECT) IS OFTEN INTERWOVEN WITH ENGLISH AND FRENCH, REFLECTING THE INFLUENCE OF GLOBALIZATION, MEDIA, AND SOCIAL NETWORKS. THE INTEGRATION OF ENGLISH, IN PARTICULAR, HAS GROWN SIGNIFICANTLY DUE TO ITS ROLE AS A GLOBAL LANGUAGE. AS A RESULT, CODE-SWITCHING BETWEEN DARIJA AND ENGLISH HAS BECOME A DYNAMIC AND INCREASINGLY OBSERVABLE PATTERN IN BOTH SPOKEN AND WRITTEN COMMUNICATION.



THIS PROJECT AIMS TO ANALYZE AND DETECT CODE-SWITCHING BETWEEN DARIJA AND ENGLISH IN TEXT DATA. BY LEVERAGING TECHNIQUES IN NATURAL LANGUAGE PROCESSING (NLP) AND LINGUISTIC ANALYSIS

DATASET PREPARATION

TO DEVELOP OUR MODELS, WE NEEDED QUALITY DATA. WE GATHERED DARIJA AND ENGLISH SAMPLES PRIMARILY FROM THE HUGGING FACE DATASET, AMOUNTING TO 87,000 SAMPLES. THE CODE-SWITCHING SAMPLES WERE MANUALLY COMPILED FROM EXCEL FILES, YIELDING 1,868 SAMPLES. THE DATASET WAS CLEANED THOROUGHLY—REMOVING PUNCTUATION, SPECIAL CHARACTERS, AND NUMBERS—TO ENSURE CONSISTENCY AND RELIABILITY. WE STANDARDIZED OUR DATA BY UNIFYING TEXT COLUMNS UNDER A SINGLE ‘TEXT’ COLUMN AND THEN LABELED THE DATA: 0 FOR DARIJA, 1 FOR ENGLISH, AND 2 FOR CODE-SWITCHING. AFTER BALANCING, THE DATASET CONSISTS OF 2,644 DARIJA SAMPLES, 2,645 ENGLISH SAMPLES, AND 1,868 CODE-SWITCHING SAMPLES. THIS BALANCED DISTRIBUTION HELPS IN TRAINING FAIR AND UNBIASED MODELS.

DATASET CODE-SWITCHING

```
                                                text
0            The file rah missing mn the shared drive
1  Khasni analyze kifash kan the performance results
2  Enta lets consider using an inmemory cache for...
3  Rah the cron job might be failing due to a con...
4  Rah lets do some load testing using a tool lik...
```

DATASET ENGLAIS

```
                                                 eng
0  Have you been in a moroccan 7ammam before
1                          I cant see the light
2                             A detective novel
3          i believe it should be on discount
4               He is on his last legs you see
(2645, 1)
```

DATASET DARIJA

```
                                          darija
0  chnou hia a7ssn blassa saferti liha ?
1              sir bniya ou rged m3a l7eya
2       jarabt dart chbkat dlbasla obtata
3                           rah ba9i l7al
4            raki sme3ti tbib achnou galik
(2644, 1)
```

DATA PROCESSING

“ONCE THE DATASET WAS PREPARED, THE NEXT STEP WAS PROCESSING THE DATA FOR MODEL TRAINING. WE USED THE BERT TOKENIZER TO CONVERT SENTENCES INTO TOKENS, SPECIFICALLY GENERATING ‘INPUT_IDS’ AND ‘ATTENTION_MASK’ VALUES. EACH SEQUENCE WAS FIXED AT 100 TOKENS TO MAINTAIN UNIFORMITY. WE THEN SPLIT THE DATA INTO TRAINING AND TESTING SETS, USING AN 80/20 SPLIT, WHICH ENSURES A GOOD AMOUNT OF DATA FOR TRAINING WHILE RESERVING A PORTION TO EVALUATE MODEL PERFORMANCE. FINALLY, WE CREATED PYTORCH DATA LOADERS TO HANDLE DATA IN BATCHES EFFICIENTLY DURING TRAINING. THIS PROCESS ENSURES OUR MODELS RECEIVE WELL-STRUCTURED, CONSISTENT INPUT DATA.”

```python
def tokenize_data(texts, tokenizer, max_len=100):
    return tokenizer(
        texts,
        padding='max_length',
        truncation=True,
        max_length=max_len,
        return_tensors="pt"
    )
```

MODEL 1 — MBERT

“HERE, WE INTRODUCE OUR FIRST MODEL, MBERT, OR MULTILINGUAL BERT. MBERT IS A POWERFUL TRANSFORMER MODEL PRE-TRAINED ON 104 LANGUAGES, MAKING IT IDEAL FOR HANDLING MULTILINGUAL TASKS. WE CUSTOMIZED MBERT BY ADDING A CLASSIFICATION HEAD TAILORED TO OUTPUT THREE CLASSES CORRESPONDING TO OUR CATEGORIES. DURING TRAINING, WE RAN THE MODEL FOR 3 EPOCHS WITH A LEARNING RATE OF 5E-5. THE LOSS CONSISTENTLY DECREASED FROM 0.0937 IN THE FIRST EPOCH TO 0.0221 BY THE THIRD. FOR FURTHER IMPROVEMENT, WE FINE-TUNED THE MODEL BY FREEZING ALL BUT THE LAST TWO ENCODER LAYERS AND TRAINING FOR AN ADDITIONAL 5 EPOCHS, REDUCING THE FINAL LOSS TO 0.0068. THE EVALUATION RESULTS WERE OUTSTANDING, WITH AN ACCURACY OF 99% AND SIMILARLY HIGH PRECISION, RECALL, AND F1 SCORES. THIS SHOWS MBERT’S EFFECTIVENESS IN ACCURATELY CLASSIFYING MULTILINGUAL AND CODE-SWITCHED TEXT.”

preparation for pytorch

```
print("Préparation des données pour PyTorch...")
class CodeSwitchingDataset(Dataset):
    def __init__(self, encodings, labels):
        self.encodings = encodings
        self.labels = labels

    def __len__(self):
        return len(self.labels)

    def __getitem__(self, idx):
        return {
            'input_ids': self.encodings['input_ids'][idx],
            'attention_mask': self.encodings['attention_mask'][idx],
            'labels': torch.tensor(self.labels.iloc[idx], dtype=torch.long)
        }

train_dataset = CodeSwitchingDataset(train_encodings, y_train.reset_index(drop=True))
test_dataset = CodeSwitchingDataset(test_encodings, y_test.reset_index(drop=True))

train_loader = DataLoader(train_dataset, batch_size=16, shuffle=True)
test_loader = DataLoader(test_dataset, batch_size=16)
```

Model 1 – mBERT

```
print("Construction du modèle...")
class CodeSwitchingClassifier(nn.Module):
    def __init__(self):
        super(CodeSwitchingClassifier, self).__init__()
        self.bert = BertModel.from_pretrained('bert-base-multilingual-cased')
        self.dropout = nn.Dropout(0.3)
        self.classifier = nn.Linear(self.bert.config.hidden_size, 3)  # 3 classes

    def forward(self, input_ids, attention_mask):
        outputs = self.bert(input_ids=input_ids, attention_mask=attention_mask)
        pooled_output = outputs.pooler_output
        dropout_output = self.dropout(pooled_output)
        logits = self.classifier(dropout_output)
        return logits
```

MODEL 2 — BILSTM

OUR SECOND MODEL IS A BIDIRECTIONAL LSTM (BILSTM). UNLIKE TRANSFORMERS, BILSTMS ARE A TYPE OF RECURRENT NEURAL NETWORK THAT CAN CAPTURE SEQUENCE INFORMATION IN BOTH FORWARD AND BACKWARD DIRECTIONS. WE USED PRE-TRAINED GLOVE EMBEDDINGS WITH 100 DIMENSIONS TO REPRESENT OUR WORDS, ENSURING RICH SEMANTIC REPRESENTATION. THE MODEL ARCHITECTURE INCLUDES TWO BILSTM LAYERS, FOLLOWED BY A DROPOUT LAYER TO PREVENT OVERFITTING, AND FINALLY A DENSE LAYER FOR CLASSIFICATION. TRAINING RAN FOR 10 EPOCHS WITH A BATCH SIZE OF 32, AND THE VALIDATION ACCURACY STABILIZED AT 99.3%. THE EVALUATION METRICS SHOWED AN ACCURACY OF 99.3% ALONG WITH HIGH PRECISION, RECALL, AND F1 SCORES, INDICATING THAT BILSTM IS ALSO HIGHLY EFFECTIVE FOR OUR CLASSIFICATION TASK.

embendding glove

```
the -0.038194 -0.24487 0.72812 -0.39961 0.083172 0.043953 -0.39141 0.3344 -0.57545 0.087459 0.28787 -0.0
, -0.10767 0.11053 0.59812 -0.54361 0.67396 0.10663 0.038867 0.35481 0.06351 -0.094189 0.15786 -0.81665
. -0.33979 0.20941 0.46348 -0.64792 -0.38377 0.038034 0.17127 0.15978 0.46619 -0.019169 0.41479 -0.34349
of -0.1529 -0.24279 0.89837 0.16996 0.53516 0.48784 -0.58826 -0.17982 -1.3581 0.42541 0.15377 0.24215 0.
to -0.1897 0.050024 0.19084 -0.049184 -0.089737 0.21006 -0.54952 0.098377 -0.20135 0.34241 -0.092677 0.1
and -0.071953 0.23127 0.023731 -0.50638 0.33923 0.1959 -0.32943 0.18364 -0.18057 0.28963 0.20448 -0.5496
```

Model 2 – bilSTM

```python
model = Sequential([
    Embedding(input_dim=max_vocab_size,
              output_dim=embedding_dim,
              weights=[embedding_matrix],
              input_length=max_sequence_length,
              trainable=False),  # Embeddings non entraînables
    Bidirectional(LSTM(units=128, return_sequences=True, dropout=0.3, recurrent_dropout=0.3
    Bidirectional(LSTM(units=64, dropout=0.3, recurrent_dropout=0.3)),
    Dense(128, activation='relu'),
    Dropout(0.4),
    Dense(3, activation='softmax')  # 3 classes : Darija, English, Code-Switching
])
```

Comparison of mBERT vs. BiLSTM

In this slide, we compare the two models across several dimensions. Both models achieved high accuracy—99% for mBERT and 99.3% for BiLSTM. However, they differ in training time, inference speed, and model size. mBERT takes about 3 minutes per epoch, has slower inference speed, and requires around 400MB of storage. In contrast, BiLSTM trains faster at about 75 seconds per epoch, has faster inference, and only needs about 3.8MB, making it more lightweight. These factors mean that while mBERT handles complex multilingual patterns very well, BiLSTM might be more suitable for environments where computational resources are limited due to its speed and smaller footprint.

mBERT

```
Évaluation du modèle...
Classification Report:
                precision    recall  f1-score   support

         Darija       0.99      1.00      1.00       538
        English       0.99      0.99      0.99       525
Code-Switching        0.99      0.99      0.99       369

       accuracy                           0.99      1432
      macro avg       0.99      0.99      0.99      1432
   weighted avg       0.99      0.99      0.99      1432
```

BiLSTM

```
Évaluation du modèle BiLSTM...
45/45 ━━━━━━━━━━━━━━━━━━━━ 4s 85ms/step - accuracy: 0.9955 - loss: 0.0217
Perte : 0.0285, Précision : 0.9930
Analyse des performances...
45/45 ━━━━━━━━━━━━━━━━━━━━ 5s 100ms/step
Classification Report:
                precision    recall  f1-score   support

         Darija       1.00      1.00      1.00       538
        English       0.99      0.99      0.99       525
Code-Switching        1.00      0.98      0.99       369

       accuracy                           0.99      1432
      macro avg       0.99      0.99      0.99      1432
   weighted avg       0.99      0.99      0.99      1432
```

Predictions

biISTM

```
Prédictions sur de nouvelles phrases...
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 107ms/step
Texte : kifach kan your day?
Classe prédite : Code-Switching

1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 98ms/step
Texte : I need to go home now.
Classe prédite : English

1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 97ms/step
Texte : chnou hia a7ssn blassa saferti liha
Classe prédite : Darija

1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 99ms/step
Texte : Let's go whit the friends!
Classe prédite : Code-Switching

1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 98ms/step
Texte : give me lwe9t
Classe prédite : English

1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 99ms/step
Texte : how was nhar dialek
Classe prédite : English

1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 101ms/step
Texte :  give me the makla
Classe prédite : English
```

Let’s look at some examples to see how our models perform. For instance, the sentence ‘chnou hia a7ssn blassa saferti liha’ should be classified as Darija. Similarly, ‘I need to go home now.’ is clearly English, and ‘kifach kan your day?’ is a mix of Darija and English, hence code-switching. Both models handle straightforward cases effectively, but mBERT tends to perform better on more ambiguous or complex sentences, thanks to its deep understanding of multilingual contexts. This shows that while both models are strong, mBERT’s nuanced language understanding gives it an edge in certain scenarios.

BERT

```
Initialisation des prédictions...
Exemple de prédiction...
Texte : chnou hia a7ssn blassa saferti liha
Classe prédite : Darija

Texte : I need to go home
Classe prédite : English

Texte : It's been 15 minutes since I've seen her!
Classe prédite : English

Texte : raki sme3ti tbib achnou galik
Classe prédite : Darija

Texte : kifach kan your day
Classe prédite : Code-Switching

Texte : can i have zit m3ak
Classe prédite : Code-Switching

Texte : can you give me the phone
Classe prédite : English

Texte : wach nta cv pas
Classe prédite : Darija

Texte : forgive me ana m3ak
Classe prédite : Code-Switching

Texte : Please 3tini l password dyal Wi-Fi rah service kherj
Classe prédite : Code-Switching
```

Conclusion

“To summarize, we successfully trained and evaluated two high-performing models—mBERT and BiLSTM—for the task of multilingual classification involving code-switching. Both models achieved over 99% accuracy on test data, with mBERT showing particularly strong capabilities in handling code-switching. Looking forward, there are several areas for future work: expanding the dataset to include a wider variety of examples, optimizing mBERT for faster inference without sacrificing accuracy, and testing our models on real-world applications like analyzing social media text or powering chatbots. These improvements could enhance both the robustness and applicability of our models in real-world multilingual environments.”