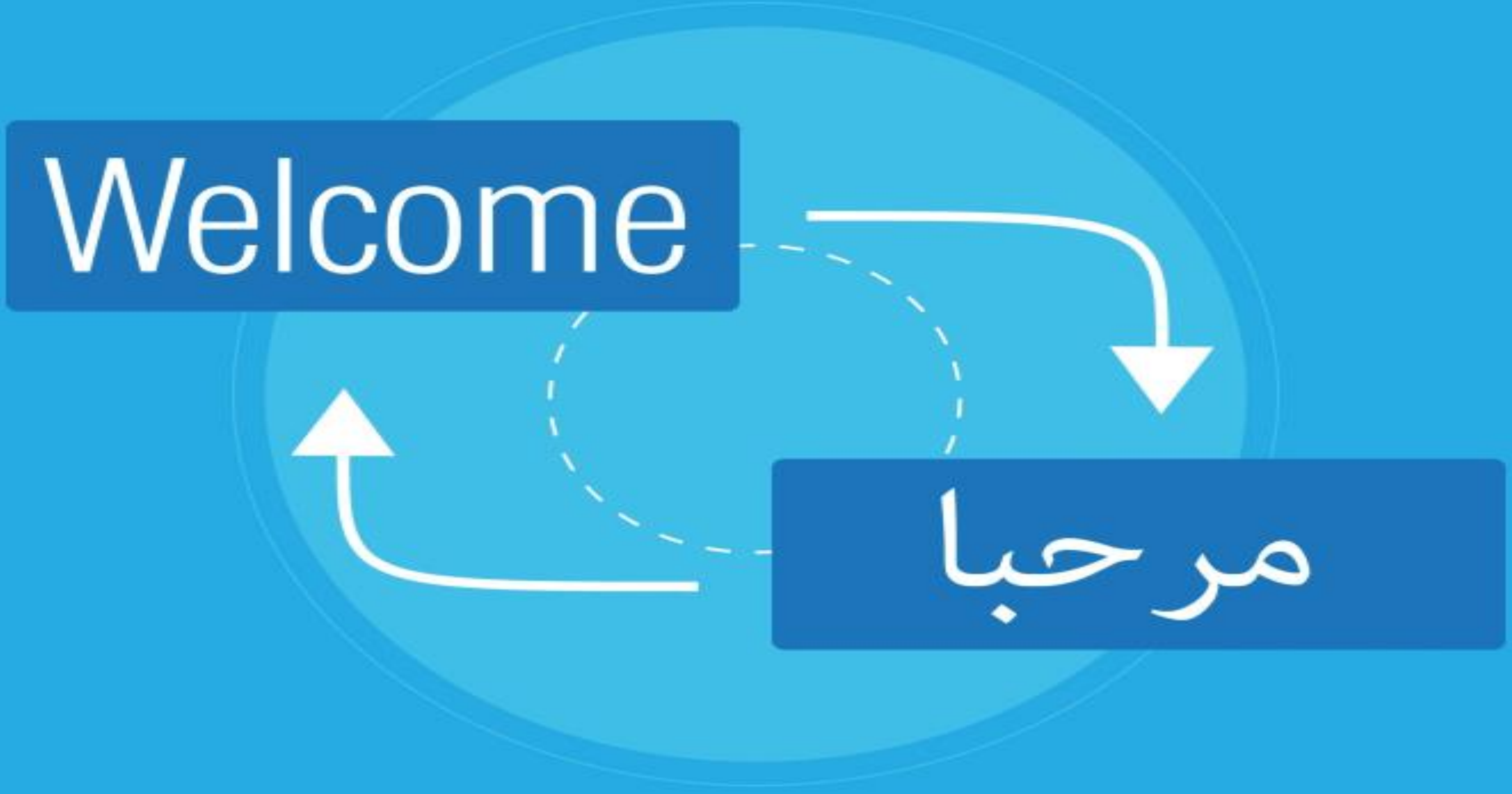
Translation

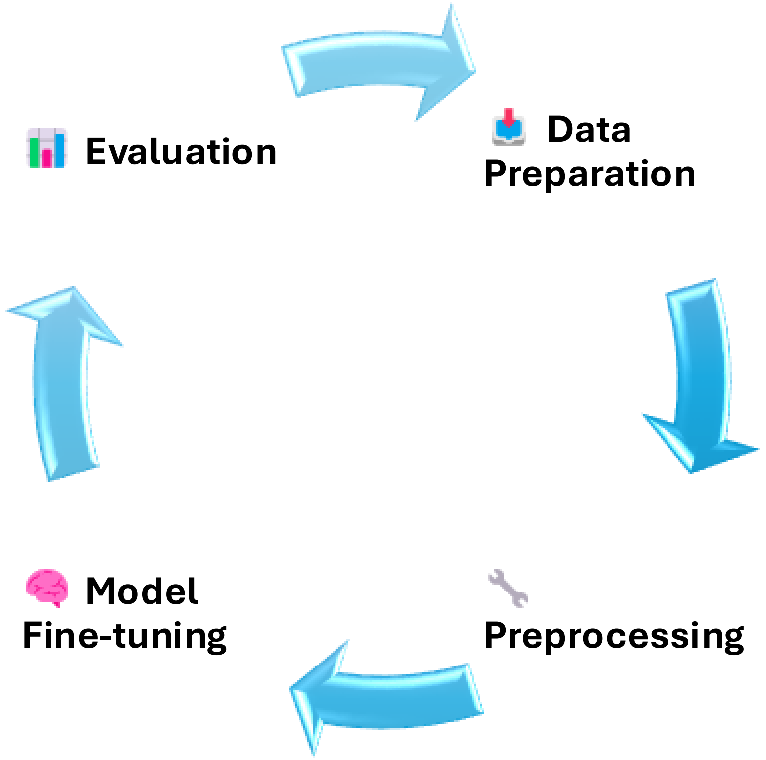


This project is represented by:

ID	NAME
20210196	ايات احمد محمد
20210170	الاء مجدي بيومي
20210168	الاء عاطف مصطفى
20210187	إنجي موسى محمد
20210071	احمد عبد الرشيد حسن
20210333	رمزي اشرف رمزي

ProjectOverview 🔍

- This project focuses on building a **Neural Machine Translation (NMT)** system to translate **Arabic to English** using a pretrained **transformer-based model** from Hugging Face — specifically `Helsinki-NLP/opus-mt-ar-en`. The model was fine-tuned on a parallel Arabic-English dataset using the Hugging Face `Trainer` API.



Dataset Information

- We used a **parallel Arabic-English dataset** from **GitHub** (Arabic-English Parallel Corpus by Samir Moustafa).
- The dataset contains **10,742 sentence pairs** of Arabic and English.
  Each line consists of an Arabic sentence and its corresponding English translation, separated by a tab (`\t`).
  You can find the dataset [here].

Data Preparation

- **First,** we started with preparing our dataset. We used a parallel corpus where each line contains an Arabic sentence and its corresponding English translation, separated by a tab. We cleaned the data to remove any missing or corrupted lines, and then we converted it into a format compatible with the Hugging Face `Dataset` library.

🔧 Preprocessing

- In this step, we used the **tokenizer** from the same pretrained model to convert the Arabic and English sentences into token IDs.
- We applied padding and truncation to make sure all sequences are of **equal length**, which helps the model learn more efficiently during training.

📝 Fine-Tuning Configuration

🔍 Pretrained Model

Helsinki-NLP/opus-mt-ar-en is a pretrained transformer model designed for Arabic-to-English translation. We used it as a base model and fine-tuned it on our specific parallel dataset.

📊 Number of Epochs (3)

The entire dataset is passed through the model 3 times. More epochs may help the model learn better but also increase the risk of overfitting.

📦 Batch Size (16)

The model processes 16 sentence pairs at a time during training. This affects memory usage and training speed.

🔠 Max Sequence Length (128)

All sentences are padded or truncated to a maximum of 128 tokens to maintain uniform input size.

⚙️ Optimizer (AdamW)

The AdamW optimizer is used to update the model's weights. It is commonly used in transformer-based models for stable and efficient training.

📉 Learning Rate Scheduler (Linear Decay)

The learning rate starts at an initial value and decreases linearly throughout training for better convergence.

📈 Learning Rate (5e-5)

By default, the learning rate is set to 5e-5 (0.00005). This controls how much the model weights are updated during training.

🔢 Precision (FP16)

Mixed precision training using half-precision floating point (FP16) is enabled if a GPU is available. This reduces memory usage and speeds up training.

💾 Save Steps (500)

A checkpoint of the model is saved every 500 training steps to allow resuming or analyzing intermediate results.

🧾 Logging Steps (100)

Training progress (like loss) is logged every 100 steps to monitor model performance.

Parameter	Value
Pretrained Model	Helsinki-NLP/opus-mt-ar-en
Number of Epochs	3
Batch Size	16 (per device)
Max Sequence Length	128 tokens
Optimizer	AdamW
Learning Rate Scheduler	Linear decay
Precision	FP16 (if GPU is available)
Save Steps	Every 500 steps
Logging Steps	Every 100 steps

Evaluation

- After training, we evaluated the model using the BLEU score, which compares the model's output to the correct translations. Our model achieved a BLEU score of **74.91**, which shows that the translations were highly accurate and close to human-level quality.

Translation Process Flow

Translation Result:

Input :

"البرمجة مفيده"

Final Output:

"Programming is useful."

("البرمجة مفيدة").

["البرمجة",
"مفيدة"]

[1254, 7890]

"Programming"
→ "is" →
"useful"

"Programming
is useful."

Input → Tokenization → Encoding → Decoding → Detokenization

⚠️ Model Limitations

- **Domain Sensitivity**
 The model is pretrained on general-domain data. Accuracy may decrease slightly with domain-specific or highly technical content.
- **Dialectal Arabic**
 The model handles Modern Standard Arabic well, but performance may vary with dialects or informal language.

Despite these limitations, fine-tuning on our dataset significantly improved the model's performance and made it more suitable for our specific translation task.

Thank you