Project Documentation: Arabic-English Translation using MarianMT

Project Title

Arabic-English Neural Machine Translation using MarianMT

Team Member Names:

- Yassmin ahmed ibrahim
- Maram essam ahmed
- Youssef tarek aboelfotouh
- Warda Khaled dahy
- Mariam ahmed Mostafa

Youssef Mohamed fathy

Overview

This project focuses on building a neural machine translation (NMT) system that translates English text into Arabic using the pre-trained MarianMT model from Hugging Face. The system is trained and evaluated using a parallel corpus of English-Arabic sentence pairs. The implementation is based on the `transformers` library, and the training pipeline leverages Hugging Face's `Seq2SeqTrainer`.

---

1. Dataset Information

Dataset Source: Kaggle

- **Name:** Arabic to English Translation Sentences

- **File Used:** `ara_eng.txt`
- **Columns:** `english`, `arabic`
- **Format:** Tab-separated values (TSV)

Dataset Preprocessing

- Removed punctuation and unnecessary characters.
- Lowercased English text.
- Cleaned Arabic text using Unicode ranges for Arabic.
- Filtered out sentence pairs with less than 3 words or less than 5 characters.
- Removed duplicate English entries.

Data Split

- **Training Set:** 80%
- **Testing Set:** 20%

Preprocessing Steps

.English Text:

Converted text to lowercase.

Removed brackets and their content: (...), [...].

Removed unnecessary characters while keeping useful punctuation: ., ,, !, ?.

Normalized characters using unicodedata.normalize() to standardize accented letters.

.Arabic Text:

Removed all non-Arabic characters using Unicode range: \u0600-\u06FF.

Removed diacritics and decorative marks, such as: ـًٌٍَُِّْ.

Removed digits and non-alphabetic symbols.

---

2. Model Information

Pre-trained Model

- **Model Name:** `Helsinki-NLP/opus-mt-tc-big-en-ar`
- **Library:** `transformers` by Hugging Face

- **Architecture:** MarianMT (based on Transformer architecture)

Tokenizer

- **Type:** SentencePiece
- **Tokenizer:** `MarianTokenizer` from Hugging Face

Input Length

- **Maximum Sequence Length:** 128 tokens

---

3. Training Details

Preprocessing

- Tokenization applied to both English input and Arabic target.
- Padding and truncation used for fixed-length inputs (max length = 128).

Training Configuration

- **Epochs:** 10
- **Training Batch Size:** 16
- **Evaluation Batch Size:** 16
- **Output Directory:** `./results`
- **Logging Directory:** `./logs`

- **Logging Steps:** 10
- **Save Steps:** 1000
- **Evaluation Strategy:** Custom bertscore callback
- **Prediction:** Enabled with `predict_with_generate=True`

Training Loss (last few steps):

- Decreased from ~6.5 to ~0.2 over 10 epochs
- Indicates effective learning and convergence

---

4. Evaluation Metrics

Metric Used: BERTScore

- **Library:** `bert_score` (from HuggingFace)
- **Language Models Used:** Pre-trained multilingual BERT models (`bert-base-multilingual-cased`)
- **Method:**
  - Computes similarity between reference and generated translation using contextual embeddings
  - Captures semantic meaning better than lexical match-based metrics like BLEU
- **Evaluation Frequency:**
  - Evaluated on 100 random test samples at the end of each epoch
  - Provides F1 score for precision-recall balance
- **Why BERTScore?**
  - More reliable for low-resource language pairs
  - Better reflects human judgment of translation quality

---

5. Model Limitations

- **Limited Domain Generalization:**
  - Trained on a small parallel corpus; may not generalize to other domains (e.g., medical, legal).
- **Token Truncation:**
  - Long sentences may be truncated, potentially affecting translation accuracy.
- **Vocabulary Limitation:**
  - Pre-trained tokenizer might not handle rare or domain-specific tokens well.
- **Cultural and Contextual Nuances:**
  - May miss idiomatic expressions or cultural references that require contextual understanding.
- **BERTScore Reliability:**
  - While more semantically aware than BLEU, still limited by the capabilities of the underlying BERT model
  - Sensitive to language-specific pretraining and sentence structure nuances.

---

6. Future Enhancements

- **Data Augmentation:**

    - Use back-translation or synonym replacement to enrich dataset.
- **Fine-Tuning:**
    - Use a larger or more domain-specific dataset.
- **UI Integration:**
    - Integrate with a GUI (e.g., Gradio or Streamlit) for easy testing and deployment.
- **Deploy API:**
    - Wrap the model in a Flask or FastAPI service and deploy to Vercel or Hugging Face Spaces.

---

7. Dependencies

- `transformers`
- `datasets`
- `sentencepiece`
- `scikit-learn`
- `nltk`
- `torch`
- bert_score

---

8. Potential Improvements

- Incorporate domain-specific data for fine-tuning
- Use larger context-aware models (e.g., mBART or T5 multilingual)
- Introduce post-processing grammar correction or reranking
- Utilize semantic-aware metrics (e.g., BERTScore) alongside BLEU
-

9.RUN

10. Conclusion

This project successfully demonstrates the application of a pre-trained MarianMT model for Arabic-English translation. Through fine-tuning and bertscore evaluation, the model achieved good performance on a general dataset, with opportunities for further enhancement in specialized applications.

-