Computer Engineering Department
Faculty of Engineering
Cairo University

CMP Elective 4
Fall 2023

Natural Language Processing
Project Document

Project Description

Arabic is one of the most spoken languages around the globe. Although the use of Arabic increased on the Internet, the Arabic NLP community is lagging compared to other languages. One of the aspects that differentiate Arabic is diacritics. Diacritics are short vowels with a constant length that are spoken but usually omitted from Arabic text as Arabic speakers usually can infer it easily. The same word in the Arabic language can have different meanings and different pronunciations based on how it is diacritized. Getting back these diacritics in the text is very useful in many NLP systems like Text To Speech (TTS) systems and machine translation as diacritics removes ambiguity in both pronunciation and meaning. Here is an example of Arabic text diacritization:

ذَهَبَ عَلِيٌّ إِلَى الْشَّاطِئِ → ذهب علي إلى الشاطئ

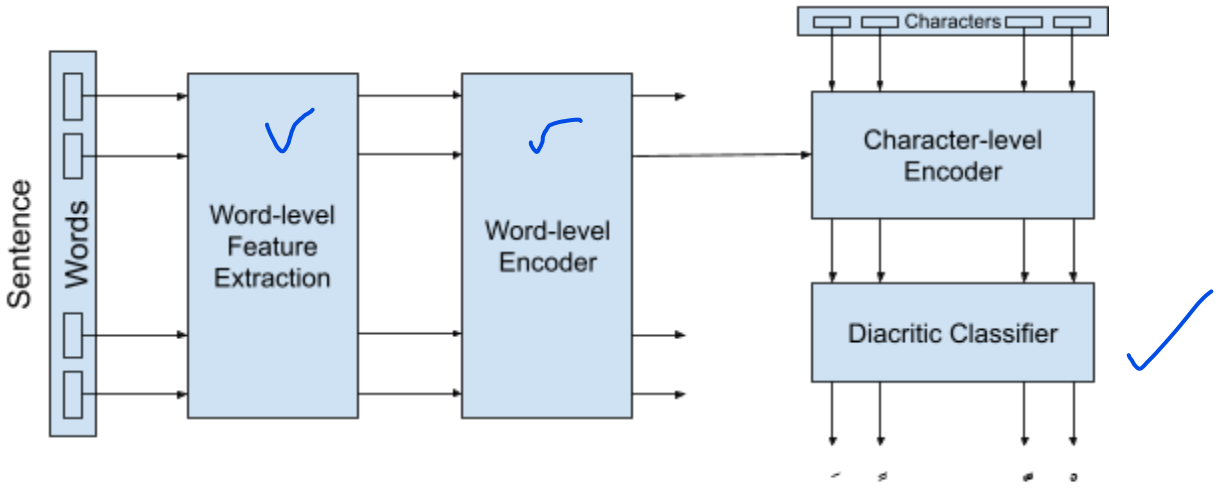
Dataset Description

The dataset contains discretized Arabic sentences. Each sentence occupies a line. The dataset is divided into three different portions (train, dev, and test). Both the train and dev sets will be annotated (All characters are diacritized). The test set contains Arabic text without diacritization and your task is to restore the test set diacritics. The dataset portion sizes are as follows:

1. The training set contains 50k lines.
2. The dev set contains 2.5k lines.
3. The test set contains 2.5k lines.

Project Pipeline

Your task is to build a system that takes a sentence and produces the same sentence after restoring the missing diacritics. There are several approaches to tackle such a problem. You are free to propose your own pipeline based on your understanding and research of the problem. Here is an example pipeline diagram that you may follow:



The above diagram is based on the problem understanding. To restore the diacritics of a word, we need to read several words before it and maybe after it also. So on the word level, you need to have the features of it besides the features of the words preceding and following it if available. To predict the diacritic of each character, this can be viewed as a classification problem per character and the classes are the diacritics. So, for the character level, each character needs to know the features also of the characters preceding and maybe following it in the same word. The features then can be fed to the final classifier. Following are some of the main steps you need to include in your project:

1. **Data Preprocessing:**
    a. Data cleaning: removing undesired words such as HTML tags, English letters, etc.
    b. Tokenization: You are free to determine the best tokenization approach for the problem.
2. **Feature Extraction:** you are required to try **at least three** different features (e.g. Bag of Words, TF-IDF, Word Embeddings, Trainable embeddings etc.) It would also be great if you tried other features (e.g. contextual embeddings). This part can be done in the word and character levels.
3. **Model Building:** In this phase, you are required to build **at least two** machine learning models (e.g. RNN, LSTM, CRF, HMM, …). Optimizing the model weights will be done using the training set. You will need to use the dev set to pick the model that performs the best. You can have multiple models in your system based on your approach. For example, you can have a model for word-level encoding and another for character level.
4. **Model Testing:** In this phase, you will use your best-performing model to produce the stance and category of the given test set.

Grading Criteria

This is a competitive project. The teams will be ranked based on the scores. You will be provided with the test set tweets only **ONE DAY** before the final delivery. We will use the Diacritic Error Rate (DER) as the metric for the ranking process. We will use Kaggle for the ranking process. The overall grading will depend on the following:

1. The team rank
2. The approach you followed. This includes the following:
    a. The preprocessing techniques
    b. The features you used (at least three different features)
    c. The models you trained.
3. The workload division.

Project Schedule

- Project Document Release: Week 6 (Wednesday 8th Nov. 2023)
- Final Delivery: Week 13.

Project Instructions

- You will work in teams of 3 or 4.
- Your final submission only will be considered for the ranking process.
- There is a penalty for late submissions.
- Any sign of cheating or plagiarism will not be tolerated and will be graded **ZERO** in the project.

Final Deliverables

1. **Final Project Document** containing the following:
    a. Project Pipeline
    b. A detailed description of each phase in your pipeline
        i. Data preprocessing
        ii. Feature extraction
        iii. Model training
    c. Evaluation: Report the DER for all trials you did.
    d. Specify what model you used for the test set submission on Kaggle and the reason for choosing it.
2. **Codes**: All scripts you used.
3. **The final Models**: the weights of the model you used for submission. Use the framework you used for training the model default format when saving.
4. **Presentation**: you will use it for the final project discussion.