Project 3: Named Entity Disambiguation

Introduction

Named Entity Disambiguation (NED) is the task of mapping entities, such as persons, locations, or companies, from a given text document to corresponding unique entities in a target Knowledge Base (KB).

In this project, you are required to develop a NED model which is able to map the entities to their true corresponding identities determined by a Wikipedia URL.

Challenges:

- Name Variations: Entities can have different representations. For example, abbreviations like "NY" for "New York," nicknames such as "Big Apple" for New York, or variations and typos like "New yokr" are common.
- Ambiguity: A single term can refer to multiple entities, its meaning influenced by context. This polysemy (having multiple meanings) is evident in names like "The Wall," which can have different interpretations.
- Incomplete Information: Effective NED systems must handle situations with sparse data or context, such as short documents with few entity references.
- Scalability and Efficiency: For practical applications like search engines or chatbots, NED systems must be fast, often delivering results in real-time. This is challenging with large knowledge bases (KBs), like the English Wikipedia with its 9 million entities and 170 million inter-entity relationships.

Dataset

The AIDA-CoNLL is a widely-used benchmark dataset in NED. The training dataset contains 946 documents, each annotated with entities and their corresponding true identities in YAGO2 KB by a Wikipedia URL (e.g. http://en.wikipedia.org/wiki/Germany). The dataset, along with a lite Wikidata KB (https://www.wikidata.org/wiki/Wikidata:Main_Page), is accessible in Kaggle. The dataset includes the following files:

- train.csv: The train dataset contains a corpus of 946 documents, each text token presented in one row. The table contains:
  + id: The id of the token
  + token: The text token
  + entity_label: The label of the entity token, B for begin and I for continuation of an entity
  + full_mention: The full mention of the entity, which is used to find entity candidates
  + wiki_url: The true identity of the entity, determined by a Wikipedia URL

Each document starts with a text token: -DOCSTART- (); and each following line represents a single token, sentences are separated by an empty line. There are two type of tokens: normal token and entity token. Only the row with entity token has information of entity_label, full_mention and wiki_url; otherwise they are empty.

- test.csv: The test dataset file has a format similar to the train dataset file. For evaluation purposes, the ground truth is not provided in the “wiki_url” column, replaced by a "?" in the wiki_url column of the entities. Again, a token is considered an entity if the information of entity_label and full_mention is available. For each entity token, the task is to generate the correct Wikipedia URL that identifies its entity. Note that the entity token which has "wiki_url" as '--NME--' does not count, you only have to predict the entity which has "wiki_url" as "?" in the test file.

The following files are part of the lite Wikidata KB, which was constructed using the English Wikipedia snapshot taken on December 1, 2019. This database includes only common entities to minimize its size.

- wiki_items.csv: This table contains 5,216,236 entities with the following information:
  + item_id: The entity id in wikidata.
  + en_label: The entity label in wikidata.
  + en_description: The entity description in English.
  + wikipedia_title: The corresponding title of the entity in the English wikipedia. Normally, the Wikipedia URL can be retrieved from the title using the format: "http://en.wikipedia.org/wiki/[TRANSFORMED_TITLE]", where '[TRANSFORMED_TITLE]' is the title with spaces replaced by underscores. For example, the title 'European Commission' would correspond to the URL 'http://en.wikipedia.org/wiki/European_Commission'. However, some titles will be redirected to generate the true link. For example, the title 'Third Planet' would be redirected to 'Earth', resulting in the URL 'http://en.wikipedia.org/wiki/Earth'. These redirect cases are listed in the table enwiki_redirects.tsv.

- enwiki_redirects.tsv: This table includes the mentioned title redirect cases. Each case is on a separate line, with each line containing a source title and a target title, separated by a tab character.

- item_aliases.csv: This table contains the possible English aliases for entities. For instance, the entity 'Universe' may be referred to as 'Our Universe', 'The Universe', 'The Cosmos', or 'cosmos'. In the table, the 'item_id' column specifies the ID of the entities, and the 'en_alias' column lists the aliases.

- statements.csv: This table contains the relations between the entities. The "source_item_id" and "target_item_id" columns specify the id of the source entity and the target entity, respectively. The "edge_property_id" column specifies the id of the relation type (property), mapped to the property table property.csv.

- property.csv: This table contains information about relation properties, including their ID ('property_id'), label ('en_label'), and description ('en_description') in English.

Besides the given data, you are free to use/crawl extra data, provided you can explain how you use them in your model. Make sure that any extra step in your data gathering **does not** use the content of the **test** dataset. (it should be considered unseen for your pipeline and accessed only during the final inference notebook).

Evaluation

- 40%: Quantitative Results: Metric (**F1-score**) & runtime (**T**)
- 30%: Code
    - Working code (20%)
    - Code quality, conciseness and documentation (10%)
- 30%: 2-page Report
    - Originality of approach (10%)
    - Interpretation of results (10%)
    - Report presentation & clarity (10%)

Metrics: The performance of your model is evaluated using the following metrics: **F1-score** (as detailed here), and processing time **T** measured in seconds. F1-score is calculated by: $\frac{2 * Precision * Recall}{Precision + Recall}$, where:

+ Precision: calculated by $\frac{TP}{TP + FP}$, where TP represents the total number of true positive cases, and FP denotes the total number of false positive cases.
+ Recall: calculated by $\frac{TP}{TP + FN}$, where FN represents the total number of false negative cases.

For example, if there are total 300 entities, your model can give a valid prediction for 200 entities and 150 among them are correct, then Precision = 150/200 = 0.75; Recall = 150/300 = 0.5 and thus F1-score = 0.6.

The performance of your implementation will be evaluated using two criteria:

1. **Computation time** (T): One evaluation criterion is the time it takes for your code to *train and evaluate* a rating prediction system. If this time is less than a **T0 = 1200 seconds**, you get the full points of this criteria, and for computation time more than T0, you get penalized using this formula **max(0, 1 - T0/T)**.

This criterion counts for **1/4** of your grade for the Results part.

2. **F1-score**: This accuracy metric will be calculated on your submitted prediction for the entities in the test set. This criterion counts for **3/4** of your grade for the Results part.

   From this portion:
   - A simple baseline using string matching should achieve a F1-score of 0.54 on the test set. The teams that **beat the F1-score of 0.54** get a minimal grade of **5.5** ; among those teams: the top 10% of the teams in the Kaggle competition get **6.0**
   - The teams that **do not beat the performance of the simple baseline using string matching (F1-score of 0.54)** get a **4.0**

Submission: The submission file should contain the NED prediction for the entities in the test file. You have to generate a submission file named "submission.csv" having the following two columns:

+ id: the id of the entity token in the test file
+ wiki_url: the wiki_url predicted by your model

The entity tokens for which your model does not provide a prediction (i.e., empty predictions) should be labeled as NOT_FOUND. Please have a look at the sample_submission.csv file in Kaggle for reference of the expected format.

Material Scope:

- Libraries: The following libraries are recommended to be used in the project: scikit-learn, spacy, numpy, torch, pandas, matplotlib, plotly, tqdm, transformers, scipy, click. **Note that for spacy library, the usage of its entity linker (https://spacy.io/api/entitylinker) is not allowed.** Other libraries for standard data manipulation and analysis libraries and machine learning / deep learning frameworks can be used with the exception of existing implementations of NED. You can use them however for the purpose of comparing your own models.

Reference:

+https://towardsdatascience.com/named-entity-disambiguation-boosted-with-knowledge-graphs-4a93a94381ef
+ B. Yang, W. Yih, X. He, J. Gao, and L. Deng, [Embedding Entities and Relations for Learning and Inference in Knowledge Bases](https://arxiv.org/pdf/1412.6575.pdf), ICLR 2015
+ https://ad-blog.cs.uni-freiburg.de/post/named-entity-disambiguation-with-bert/#bert-ned