NUANS Homework 1a: Named Entity Recognition

Leonardo Lavalle
1838492
Sapienza, University of Rome
lavalle.1838492@studenti.uniroma1.it

1 Introduction

Named entity recognition (NER) is a natural language processing technique that automatically identifies named entities in a text and classifies them into predefined categories. Entities can be names of people, organizations, locations, times, quantities, and more. With named entity recognition, you can extract key information to understand what a text is about, and therefore of fundamental importance for Narrative Understanding. Nowadays, all the SOTA NER models are trained on huge amount of data coming from different domains (wikipedia, newspapers, medical data, religious texts and many more). Basically from data relatively easy to access from the Web. This homework had precisely the purpose of addressing the lack of data in the literary domain.

2 Approach

Using a NER system trained on *non-literary* data on our dataset will probably bring poor outcomes. So, the immediate thing that came to my mind was to *finetune* the most popular models on the as similar as possible data to our domain. The best architectures to perform this task are the Transformers and the best dataset choice for finetuning was **LitBank** [**?**]. But while I was thinking about which strategy to use for the task, during my several web searches, I encountered **BookNLP** [**?**], a natural language processing pipeline that scales to books and other long documents. This library performs also entity recognition using a BERT transformer (plus a 3 LSTMs cascade at the end) trained exactly on LitBank and a new dataset of 500 contemporary books.

3 Comparison between SpaCy and BookNLP

A qualitative comparison between the *Spacy NER tagger* and the *BookNLP* one has been made. It was worth making it because the two models were trained on two completely datasets and different domains (Spacy model on *OntoNotes 5*), and a lot of insights emerged. Moreover, two different architectures were used: *roberta-base* for Spacy and *google/bert-uncased* for BookNLP. In order to analyze the performance of the two NER taggers in a qualitatively way, a visualization tool was necessary. The perfect and easier choice was **DisplaCy**, a built-in software of the Spacy library. It's been useful even during all the experiments: thanks to it we could see if the tried NER systems were performing well on our unlabeled data.

3.1 Observations

The BookNLP model was pretrained with the aim of extract entities, not only *named*, therefore it considers also pronouns (him, he, you, etc.) and common nouns as entities (a man, the princess, etc.). It's quite explicit the ability of BookNLP to detect the *agents* of the various stories, the *characters* acting, communicating and thinking throughout the plot of the story. That's because, having been trained on literary books, the model has been able to learn the main features of this kind of narrative patterns. It understands when a particular entity is behaving as a human, a person (PER). Indeed in *"The Frog Prince"* or in *"The Enchanted Snake"* the frog and the snake respectively are considered as people: they express opinions, they chat, they have human-like thoughts and the system recognize it. This happens also in *"The Raven"* with the raven and the giant. This model in general, for what I saw in a qualitative manner over a sample of 8 stories, performs always better with respect to SpaCy.

Even if the latter it's not bad at all, in some occasions it showed its limits; due to the *not-knowledge* of certain text structures that are widely adopted by fiction books. The only case in which SpaCy has performed better, is with the *".007"* short story. That's probably because the story is about a more real-world scenario than the other tales (it deals with transportation vehicles). In particular many organizations are not identified and this is probably due to the fact that LitBank has very few ORG entities annotated.

One thing is sure for the BookNLP model: when a certain entity is involved in direct speeches, even if is an animal or a car, it'll be considered as a person (PER).

4 Solution

The major shortcomings of this model are the erroneous identification of the *plural entities* as a single one and the very frequent inclusion of *adjectives* in the entities span. Probably these epithets are included because they are important in literary texts. To address all these issues and for the superiority of BookNLP with respect to SpaCy in this domain, I used its NER model for the initial predictions and then *post-process* them for obtaining the best results.

4.1 Implementation

Starting from the BookNLP github code, I extracted the components that I needed for our task and a public repository was built for convenience from where it could be retrieved the code anytime. To address the main problems of the extracted entities I had to delete all the *pronouns* entities; manipulate the *nested entities* that are present because of LitBank; merge *LOC* with *FAC* and *GPE*; not consider entities with a *length greater than 8*; delete *adjectives* and *plural* entities. The most delicate situation was when dealing with *noun entities*, which are not proper names and I needed to decide which consider named or not. There were simply kept the entities in which it was present at least a word in capital letter (being careful to check that the word did not come after a point or a quotation mark) and, for discarding the plural entities, it's been checked for the final 's' in each token of the span.

5 Entities clustering

For the optional part, I decided to perform a sort of named entity "*clustering*", providing a **unique string** which identifies all the occurrences of a certain entity in the text. BookNLP library has a sofisticated tool to execute characters clustering, using a lot of additional informations to help the sytem in order to group all the mentions refferring to the same agent. Instead, I wanted to use a very simple approach, without using any machine learning techniques. The extracted named entities will go through:

- a first *preprocessing* step, where all adjectives and prepositions which accompany the entity are removed.
- a second step where all the entities that are **subsets** of other ones are clustered together.
- a third one where all the entities which are **permutations** of other entities are grouped in the same entity mention.

The *start offset* and the *end offset* of each recognized named entity represent the initial and final character of the entity span (both *inclusive*). These are computed considering the whole document and not the single sentences, starting from position 0 and treating also spaces and '\n' as characters.