Write Me (P8)

Natural Language Processing Project - A.Y. 2023/2024

Nicola Rinaldi 20161A

1 Introduction

The goal of this project is to propose a strategy to extract information from semi-structured documents like emails messages or chats.

The first part of this work shows a way to extract structural elements such as greetings, signatures and the message body. This will be made in two steps. The first is inspired by [1] and uses a lightweight technique to classify the email messages at a sentence level exploiting unique characteristics of each type of entity building a big labeled dataset. This is important because many big email datasets are available but not labelled data for this specific task. The second, takes the result of the previous step to fine-tune a pre-trained model like Bert. Large Language Models are already trained to understand the language and the fine tuning can help to capture characteristics that the previous classifiers can not and then improve the classification process.

The second part is not a proper argument extraction task but uses the same techniques for the extraction of the theme of the messages. The idea behind is retrieve all the email messages that exploit an informative subject-message pair. If we are able to do so, the subject is a good descriptor for the theme of the message. Then we can fine-tune a (generative) pre-trained model to generate for a message the correspondent argument.

2 Methodology

The extraction of the structure of the email messages is treated as a classification problem at sentence level. If we had a large labeled dataset the problem would be much easier because we could skip the first part and use it simply to fine tune a LLM. The idea is to build this type of dataset using lightweight techniques that still guaranties a small error probability.

What is needed is a small amount of examples for each type of entity (let's say a couple of hundreds) and use them to classify the sentences of the messages using a measure of similarity. This measure can change depending on the type of task or on the computational power available. For example the first technique proposed is based

on string matching that doesn't require massive computation, the second based on sentence embeddings similarity that is way more expensive. A sentence is classified with a specific label only if it has a high value of similarity with at least one example in the reference dataset.

The idea is to be very strict with the comparison so two sentences are defined similar only if we are pretty sure. Clearly using this strategy we will exclude a lot of sentences from the classification but if we have very large dataset the extracted labelled sentences will be big enough.

2.1 Lexical Similarity Techniques

The first two strategies used are based on classical concepts for natural language processing: n-grams and Bag of Words. For our task we consider n-grams of words that means sequences of n words.

Both try to extract sentences that share a part of their tokens (the first in a different way than the second) with the reference sentences for each entity to be recognized.

The first strategy is based on **partial string matching**. Two strings have a partial match if a subsequence of n tokens (words in our case) of the first sentence appears in the second one. In this work the comparison is made between the list of given entities and the sentences extracted from the email messages. If a sentence share a subsequence (of length n) with at least one of the entities then it is classified with the label of the entity. As said previously we want to be very strict with the comparison to be enough sure that the classification is correct. This means that we cannot choose too low values for n. On the other side we have to limit n to guarantee to find a big enough number of examples to train the model in the second phase.

Another option is use the concept of **bag of words** to find similar sentences. This time we don't consider the order of words to find a match but only the number of words the two sentences share. We consider two sentence similar if they share a big enough set of meaningful words. We say meaningful because the stopwords could introduce an error in our classification. The solution is simple: we can either consider higher values for the thresholds or we can remove stopwords before the comparison. In this way the risk to consider two sentences with different meaning similar just because they share very common tokens is reduced. Also in this case we want to keep this value high enough to guarantee a decent classification.

In both cases lemmatization can be a good idea to maximize the matching performance.

What we hope is that a sentence that is classified with a specific entity label does not contain only the tokens or sequences matched with the reference sentences, but also other words and sequences specifics for that type of entity. This is the reason because we look just for a partial match, otherwise with so limited reference datasets would be impossible to extract enough data for the next step.

The previous techniques has the advantage to be not too expensive in terms of computational power, but also have some drawbacks. For example, a lexical comparison in some cases is not sufficient to capture the similarity of the meaning of two sentences.

2.2 Word Embedding techniques

The last strategy is based on sentence embedding. Both the previous examples are based on the lexical structure of the sentences but what we want to compare is the meaning of the sentences. The same words can have multiple meanings and consider just the common words can lead to a wrong classification.

We use Universal Sentence Encoder to represent our sentences as vectors and use a similarity measure to compare the given entities with the email phrases.A possible value of similarity can be the cosine similarity and this time we have to choose a threshold in [-1,1] to define when we have a match.

2.3 Classification

The classification for each entity is done as described above except for the body entities. We could simply collect sentences form the email messages, maybe keeping just the central ones to avoid the pleasantries or the signatures. Anyway this doesn't guarantee not to extract also other types of entity. Other options have been considered for the extraction of the bodies. The easier is to keep all the sentences that are not classified as one of the others entities. But this is not correct because of the strategy we adopted for the classification: if we are very strict with the thresholds for the comparisons, many sentences that actually are a pleasantry or a signature have been ignored and would be classified as body.

To avoid this problem two alternatives are proposed.

The first is to classify as body only the sentences that have very low scores in the comparison with the other entities. This is reasonable but not in all the cases is possible. For the word embeddings we have a measure of similarity in the range [-1,1] but in the lexical techniques we have only a boolean value that tells us if there is a match and not how many tokens match. This could be added in the algorithm but also a third strategy is possible.

The idea is to use the same strategy used for the others sentences, but this time as reference dataset for each email messages a particular value is used that comes with (almost) every email: the subject. If the subject is informative probably there will be a reference in the message body and if there is we will extract that specific sentence and it will be classified as body. Also that sentences will have a different structure, length and meaning compared to other entities. This could be sound weird and very limiting because many sentences are being thrown away but is just a strategy to be sure not to keep sentences that should be classified with other labels. The reasoning is the same for the other entities: if we have a very big dataset the amount of extracted data will be enough. This idea inspired also the second part of the project to extract the argument of the messages.

3 Argument Extraction

The argument extraction is treated as a generative problem. If we think about it, a collection of pairs email-subject can be seen as a dataset where the emails are the content and the subject are the topic or the argument of the email. Clearly this is not

true for all the emails because most of the times who write an email doesn't put much effort to produce a meaningful subject. But if we use one of the technique proposed before for the extraction of the entities to check if the whole message has a match with the subject, we can filter out all the not useful records.

For this specific task the objective to extract arguments that describes the massage content, but not to short (like tags) but also not too long (like a summary). So we filter out also the subjects that not follows these characteristics.

Once obtained the dataset it is processed to fine tune a generative model. T5 is used that is a encoder-decoder model and converts all NLP problems into a text-to-text format, so is a good choice for our problem.

4 Data

4.1 Data Sources

The email messages are taken from the Enron Dataset. The messages are in MIME format so it is quite simple to extract only the necessary information, in our case the subject and the body is sufficient.

For each entity to be extracted from the messages a dataset of some hundreds of examples has been built.

The dataset for the signatures is taken from a public github repository that provides a library to extract the signature from the emails alongside a dataset annotated with a signature label. The signatures has been extracted with a simple parsing algorithm.

For the pleasantries, the dataset has been built collecting manually a various sample of entities from some big email dataset.

4.2 Data Cleaning

Some data manipulation is necessary to achieve a decent result.

First of all we have limited the size of the messages: sometimes the dataset has some very long messages with weird structures that we don't want to consider.

We filtered out all the messages with non-informative subjects like empty strings, `no subject` placeholders, too short sequences and just numeric subjects. Email threads terms are also removed like `Re:`, `Fw:`,...

Fortunately the dataset of the signatures was a subset of the Enron Dataset. This was an advantage for extract some recurring patterns like the company name and department in the email signatures. Those kind of things help the matching and the classification. An important thing is to remove this specific lines in the second step if the LLM is fine tuned with emails of just one source because it is supposed to work with all type of email messages.

5 Results

It is difficult to provide an evaluation metric for this kind of experiments because we don't have a labeled dataset available. Clearly it can be built manually and extract some common metrics like precision, recall, f1-score,...

What we are interest in is that both step have reason to exist and whole idea is at least applicable with decent results. This means that the first step should classify with low error probability the different sentences and the second step should be capable of learning more deeply each type of entity to be able to recognize unseen examples.

In the attached notebook we show some result obtained. The experiment is executed on reduced dataset and low training time because of the limited resources available but can still show some good results.

For the first part we show that we are able to classify correctly some unseen examples that have no match with the give reference datasets. This is true for all the three strategies adopted for the first step classification.

For the second part the constraints on the structure of the subjects were too limiting to extract enough examples from just the enron dataset. Reducing these constraints (for examples lower the thresholds of the matches) and training with a larger set of emails showed an appreciable results and in many cases the outputs described correctly the arguments of the emails. Using t5 we have the opportunity to specify a prompt for the generation. In this case, probably for the size of the dataset on the training time any prompt seems to influence to much the generated results deviating from the expected outputs (for example it produce too long description or too shorts like single words). So it is omitted still obtaining good results.

6 Possible improvements

The possible improvements of this pipeline concerns the first classification step. There can be infinite possible techniques to classify the sentence and similarity measures to improve the performances.

In this work we used just one dataset of email messages but clearly use different source of data could lead to a more general learning and results.

References

[1] Agrawal, et al.: Scalable adhoc entity extraction from text collections (2008)

[2] Cer, D., et al.: Universal sentence encoder. CoRR **abs/1803.11175** (2018) 1803.11175