SAPIENZA
UNIVERSITÀ DI ROMA

Data Mining Technology for Business and Society

Homework 3

June 2020

Task 0: Pre-processing

The first part of the assignment consists on cleaning the "TRAIN SET" and "DEV SET" sets. As first point, all datapoints that are labeled with "NEI"(not enough information) have been discarded because only datapoints with label SUPPORTED or REFUTED can be used from now on. After that, using Flair, NER (Named Entity Recognition) is performed in order to get the entities for every claim. Once all entities have been found, another discrimination has been performed. All entities that are multiple token and that are not in the BERT vocabulary have been discarded. Moreover, as last step, every datapoint is enriched with additional information, i.e. a dictionary containing the entity name, the start and end character have been add to each datapoint.
The code used to accomplish this task is: "task0.py"

Task 1: Use LAMA to get predictions for masked entities

For this second part of the assignment, LAMA repository has been used to predict, for every claim, the entity. In particular, every entity in every claim has been masked using the [MASK] string character (This part have been included in the pre-processing step. I.e the code used to mask the entities of DEV SET and the OFFICIAL TEST SET has been inserted into the "task0.py").
This part is composed by 3 .py files:

- task1.1.py
- task1.2.py
- task1.3.py
- evaluation_metric3.py

In particular, the evaluation_metric3.py is a light modification of the evaluation_metric.py provided by LAMA. This file has been used to speed up the computation and returns all the necessary information used to accomplish this task. In particular, it returns the log_prob (that is the probability that BERT associates with the correct token) and the word_form (which is the predicted word)

Task 1.1

In this first part of Task 1, the goal is to compute the accuracy, on the DEV SET, by using a single string matching heuristic. I.e. it has been looked to the first prediction returned by the LAMA. If the first predicted entity is equal to the real entity, then the prediction succeed otherwise no. In order to optimize the heuristic, a sort of normalization have been performed, in particular lowercase and stemming.
The Accuracy returned is = 10%
The code used to accomplish this task is: "task1.1.py"

Task 1.2

In this second part of Task 1, the goal is to compute the accuracy, on the DEV SET, by increasing the basket of possible predicted entities, in fact now it has been looked at the top10 predictions. I.e. if the first predicted entity is in the top10 retrieved entities the prediction succeed otherwise no. In order to optimize the heuristic also here string normalization has been performed as before (lowercase and stemming).
The Accuracy returned is = 55%
The code used to accomplish this task is: "task1.2.py"

Task 1.3

For this third part of Task 1, it has been shown how the accuracy changes while changing a probability threshold. In particular, now is not enough that the real entity belongs to the top10 predicted, but also the probability associated to the corrected predicted label has to be ¿= to given threshold. For example, if in the set of the top10 predicted entities there is the real entity but its probability is below a threshold, then this prediction is not counted as a succeed. The log_prob is the probability that BERT associates with the correct token, and this one has been used as metric.
As the two plot below show, 24 different thresholds, spanning from -0.5 to -12 have been set. As it is possible seeing from the two plots, as soon as the log_probability decreases then the accuracy starts increasing.

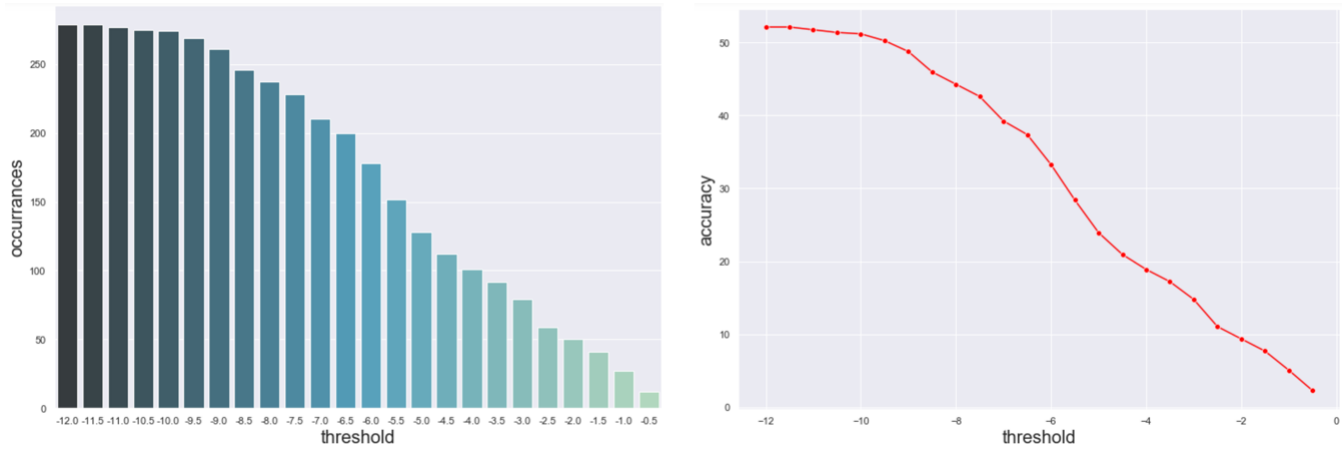

Figure 1: On the left the bar plot and on the rightLine Plot

Task 2: Train a classifier on MASK and gold representations

For this last part of the assignment, a binary fact-checking classifier have been trained to predicted the label SUPPORTS or REFUTES for the OFFICIAL TEST SET.

In particular, to get the vectorial representation, it has been picked the vector of the last layer of the contextual embedding associated with the [MASK] token and the last layer of contextual embedding with the entity token. After that, these two vectors have been concatenated together and have been used as input to train classifiers. More precisely, the DEV SET vectorials representations have been used to train the classifiers. The modality used to obtain the vectorial representation has been used also for obtaining the OFFICIAL TEST SET vectorials representaions.

I.e, every input is represented by a vector of dimension 1536.

Because the vector dimension is very large, PCA has been performed before training the classifiers. Since PCA is interested on components that maximize the variance, data should be on the same scale. For that reason, input data are first standardized and then PCA (using sklearn library) has been performed. After have normalized the data, PCA analysis returned that a dimensionality reduction of 250 leave an explaned variance of 93%, for that reason, after PCA, the vectors have a dimension of 250.

After that, 5 different classifiers have been trained and, by using GridSearch to tune the parameters, the best classifier has been SupportVectorMachine with an accuracy of 63%. Below is reported the table with all the classifiers that have been tried with their different paramiters combinations:

Table 1: Tested models and parameters

model	**parameter**	**tested values**
kNN	number of neighbors	1, 50, 100, 150, 200
kNN	weights	'uniform', 'distance'
DTC	max depth	5, 10
DTC	max features	1, 7, 13, 19, 23
SVM	kernel	'rbf', 'linear'
SVM	gamma	0.1, 0.01, 0.001, 0.0001
SVM	C	1, 10, 100, 1000
RF	number of estimators	10, 255, 500
RF	max depth	10, 55, 100
RF	bootstrap	True, False
ada	number of estimators	500, 1000, 2000
ada	learning rate	0.01, 0.1, 1

After that, by using the SVM classifier, the predictions are performed by passing the vectors representing the OFFICIAL TEST SET. The predicted labels have been stored into the predicted.jsonl file.