HOMEWORK 3 – DATA MINING FOR BUSINESS AND SOCIETY

Group members:

- ***Alessandro Taglieri, 1890945***
- ***Guglielmo Lato, 1257406***

Deep Learning and Natural Language Processing

- **Homework Description:**

In this paragraph we report what we did in this homework through a several points:

1. At first we clone LAMA repository and we have prepared the environment where we've worked on. We followed instructions in the README on LAMA repository : https://github.com/facebookresearch/LAMA .
2. We downloaded all file (train, dev and set) and Bert vocab from the links written in the homework and we installed Flair – Named Entity Recognition. This tool is used to get entity in the claims.
3. We start implementing **Task 0**: preparing and pre-processing input data. We discard all claims with NEI info on 'labels'. After that we used Flair to get entities for every claim. We only keep entities that are single token (one word, without space and "-"). If there are more entities in a single claim we discard the claim because, we want accept only claims with exactly one entity.
4. After the pre-processing task we created all data-point for every claim with single entity. Every entity is transformed in lowercase and we delete all puntuaction in this word. For example we noticed that flair recognized last word as entity including also "." as end character of this entity. So, we delete this '.' from the entity and split 'end character' minus one. Now we have all data_point to be processed with LAMA tool.
5. We built an automatic script that simulates evaluate_generation.py. We insert automatically the bert option and we run this script for every data_point in input. We have changed also evaluation_metrics.py to obtain from this function (it is called in evaluate_generation.py) also a dataframe with all info about top 10 predicitions. We need this dataframe because we have only a string message with predictions. We preferred to use a dataframe structure for a better manipulation in the next step to get info on all lama predictions.

Task 1.1:

For every data point we verify if LAMA recognizes the masked entity in the right way as first prediction. If this is happens we return a 'SUPPORT' label, otherwise 'REJECT'. We append these labels in a list and in the end, we calculate accuracy.

ACCURACY: 21 %

Task 1.2:

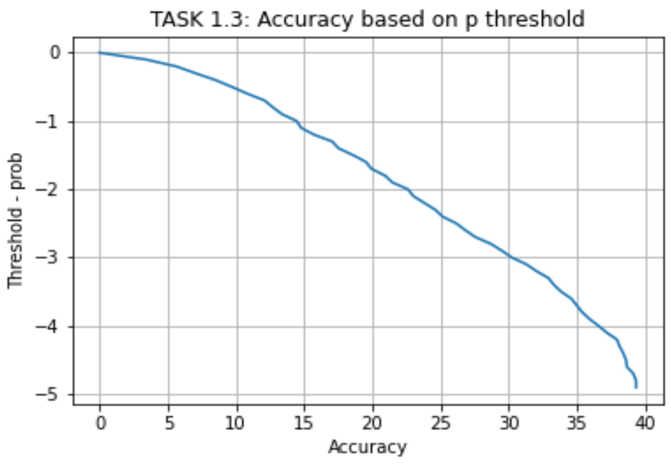
For every data point we verify if LAMA recognizes the masked entity in the right way as one among the top 10 predictions. If this happened we return a 'SUPPORT' label, otherwise 'REJECT'. We append these labels in a list and in the end, we calculate accuracy from this list.

ACCURACY: 41 %

HOMEWORK 3 – DATA MINING FOR BUSINESS AND SOCIETY

Task 1.3:

For every data point we verify if LAMA recognizes the masked entity in the right way as one among the top 10 predictions. If this happened we return a 'SUPPORT' label, otherwise 'REJECT'. In the case of SUPPORT label, we get also get the prob of prediction. We append this info (label and prob) in a list for all data points. After that, we build a list of threshold (from -5 to 0). We filter every prob based on this range.
We calculate accuracy based on this threshold.
This following figure is the plot that we've created. In the y-axis we have all threshold (from -5 to 0) and in the x-axis we have all accuracy values.



Task 2:

For this task we built a new automatic script to simulate get_contextual_embeddings.py . We inserted automatically the bert option and we run this script for every data_point that we have in input. For every data_point we get a single vector that is a concatenation of the two representations (one for masked sentence and one for the no-masked sentence). We calculate these vectors for each claim (only for those filtered as described in task0) in train_set. These are used to the train a binary classification model (in this case we use SVM) using the results from LAMA tool (for each claim).
After the training, we apply the model to get the predictions for our test set, using the Accuracy metrics, we implements the GridSearch technique to finetune the hyperparameters of the SVM model. The best parameter obtained are the following:

```
{'C': 15, 'coef0': 0.001, 'degree': 3, 'gamma': 'scale', 'kernel': rbf}
```

After that we get all vectors for all the claims in test set and we get the predictions for this official test set.
The result of predictions are reported in the json file that we attached in .zip file. This json contain all prediction for every claim (identified by its ID)

We obtained a score accuracy equal to 0.80 .

INFO TO EXECUTE TASKS: In the .zip file you should see task1_1.py, task1_2.py, task1_3.py, task2.py. To execute all scripts, you should put them in LAMA/lama repository, in this way they should work well in according to LAMA repository. As we written before, we changed content of evaluation_metrics.py (in LAMA/lama). You can find this new file in the .zip .