Exercise 5 - Sequence and Anger Regression using Transformers

Contextualized and Transformed

Deadlines

The deadline for Exercise 5 is **04.12.2022, 23:59 CET** (Zurich Time)
The deadline for the peer review is **12.12.2022, 23:59 CET (Zurich Time)**. You will find instructions for the peer review process at the end of this document.
The deadline for feedback to your peer reviewers is **16.12.2022, 23:59 CET (Zurich Time)**

Learning goals

This exercise introduces you to named entity recognition and transformer-based neural architectures. By completing this exercise, you should …

- … understand NER as a task and the IOB-format.
- …be able to use pretrained models like BERT and fine-tune them on a specific task (using HuggingFace, Pytorch, TensorFlow)

Please keep in mind that you can always consult and use the exercise forum if you get stuck (note that we have a separate forum for the exercises).

Deliverables

We encourage you to use Colab to develop your notebooks, since there you have access to GPU time. **After you have finished the assignment, download your notebook as a .ipynb file.** That way your reviewers can view and execute your code. Or can view your already executed code.

Please hand in your code and your lab report. Hand in the following files and name them exactly in the following fashion:

- ex05_ner.ipynb
- ex05_sent.ipynb
- ex05_labreport.pdf

zip it and name the zip-folder *ex05_ml4nlp1.zip*.

In case your sessions time out on Colab due to long training times: Although we expect this to not be the case, get in contact with us early and we will figure out a solution

Please submit the lab report in PDF format. The lab report should contain a detailed description of the approaches you have used to solve this exercise. Please also include the results. In this exercise description, we highlight places in green where we expect a statement about an issue in your lab report.

Please note:

- Your peers need to be able to run your code. If it does not work, you will not be able to obtain the maximum number of points.
- DO NOT submit any data files!

Data

For this exercise, you will work with a small part of the polyglot-ner dataset, described in this paper; and with a documentation of the corresponding python-package. However, these resources are just for reference, you can import the dataset into Python directly using the 'datasets'-library.

The polyglot-ner dataset contains 40 languages. Choose one language to work with from of that dataset. The following conditions need to hold for the language:

- It is not English.
- There must exist a pretrained Hugging Face BERT-base model for the language (an alternative solution is a multilingual model). (Check out this link for a list of all pretrained HuggingFace models.)
- The polyglot-ner dataset needs to contain at least 7000 sentences in this language.

Using the datasets-library to extract two training sets, one containing 1,000 sentences and one containing 3,000 sentences.
Also, extract an evaluation set (in this exercise, we don't differentiate between development and test set) of 2,000 sentences.

NOTE: Check out this part of the documentation to only download the parts of the dataset you need.

Named Entity Recognition using BERT

Implement a named entity recognition system for your chosen language. Use HuggingFace's BertForTokenClassification-class and initialize it with a pretrained Hugging Face BERT-base model of your chosen language. This HuggingFace guide for fine-tuning serves as a good starting point. Before passing the data to the model, you need to encode it using a HuggingFace tokenizer. Use the tokenizer corresponding to your BERT model. When provided with the right arguments, the tokenizer can also pad and truncate the input.

You can reduce the amount of code for this exercise by using the Trainer class explained at the bottom of the HuggingFace guide.

You will create 3 fine-tuned versions of the system:

1. Fine-tuned with 1,000 sentences
2. Fine-tuned with 3,000 sentences
3. Fine-tuned with 3,000 sentences and frozen embeddings

Let each fine-tuned model predict on the evaluation set to compute f1-micro and f1-macro scores.

Then, answer:

1. When initializing the BertForTokenClassification-class with BERT-base you should get a warning message. Explain why you get this message.
2. Which model performed best on the evaluation set?
3. Are there differences between f1-micro and f1-macro score? If so, why?
4. How large is the performance gap between 1'000 and 3'000 sentences for finetuning?
5. Is it better to freeze or not to freeze the embeddings?

Please ensure to run your code on Google Colab with GPU as a hardware accelerator selected. Select the GPU at "Edit" → "Notebook Settings".

NOTE: Loading several BERT-models at once into memory will lead to an out-of-memory error (at least as long as you don't have more than ~40G memory available). To avoid this, load and fine-tune only one model at a time, and then delete the model from memory (or overwrite it). If you want to be extra cautious, you can save the model state before deletion. Specifically on Colab, the BERT model already uses up most of the GPU-memory, and with a large batch size, you may get an out-of-memory error. Reducing the batch size should solve the problem.

Part 2: Regression Competition: How angry are you?

BERT is the eldest Transformer and it is still very popular and very well-performing. But, of course, there are many more transformers out there and some seem to be performing better on different tasks.

In the last exercise, you used CNN's to explore the emotion detection task. Now that you have the data loading ready, we ask you to adapt it to read the Regression set of the same dataset in **English**. The idea of this part is straightforward. We expect you to submit a fine-tuned Transformer model for the Affect of **Anger** in the English Regression task. We have uploaded the txt files of the dataset in the exercise's material folder. We hope you will explore a few models (at least briefly read through their papers and choose one to train). Of course, you can also do this experimentally. We leave this decision to you. You are allowed to complete Part 2 by using simpletransformers (easier to implement) by following their Regression Guide. Of course, you can always use HuggingFace just like you did for Part 1.

Tip: Preprocessing also matters when adapting to the Twitter domain ;)

Please have a statement in your report as to why and what hypothesis led you to choose this architecture. Defend why you chose/chose not to do any preprocessing? Did your results support the hypothesis? Why/Why not?

To make this more fun, let's pronounce this a competition. The students who submit the highest Pearson-r score anonymously, get to win a mystery present. If someone gets a better performance than the instructors, they also get a flat bonus on their final grade. However, let's set some fair rules:

- Please do not use any additional data (but you can use some clever preprocessing that uses external libraries)
- Make sure your results are reproducible (therefore a model can be retrained and get similar results)

Take the best performing model and evaluate it on the test set. Report your test set results (Pearson-r score) in the forum "Exercise 5 Test Set Performance"

Important:

- In order to protect your anonymity from the Exercise 5 Test Set Score Peer review: When posting in the forum you now have the option to choose a pseudonym. Post under a pseudonym that has no connection to you and does not allow for any inferences about who you are.
- If you work in a group, it is enough if only one group member posts the test set result.
- Only evaluate on the test set selected, please, use only the official training set for training. Do not optimize hyperparameters based on test set results.

Peer Review Instructions

If you are not already registered on Eduflow follow this link https://app.eduflow.com/join/GXHN93 and register with the E-mail address you use for OLAT. Then you should be added to the course page automatically.
As soon as the deadline for handing in the exercise expires you will have time to review the submissions of your peers. You need to do **2 reviews** to get the maximum number of points for this exercise.
Here are some additional rules:

- If you do not submit 2 reviews, the maximum number of points you can achieve is 0.75 (from a total of 1).
- Please use full sentences when giving feedback.
- Be critical, helpful, and fair!
- **All reviews are anonymous: Do not put your name into the python scripts, the lab report or the file names.**
- You must also give your reviewers some feedback. The same criteria as above apply.
- If you consistently provide very helpful feedback, you can be awarded a bonus of 0.5 in total in case you didn't achieve the full 6 points from all exercises. A maximum of 6 points from the exercises can go into the final grade.

Groups:

- You can create groups of two to solve the exercise together.
- Both students should submit the solutions separately.
- If you did not already work together for the previous exercise, write a small post in the "Groups"-thread in the exercise forum on OLAT to notify the instructors about the group.
- As a group member, you still have to review two submissions with your own eduflow account. However, you may work together in the group to write all 4 reviews.

Good luck with working with transformers