Project

Opinion Mining Evaluation Forum

The task

According to Oxford Languages, **Opinion Mining** consists of identifying and categorizing opinions expressed in a piece of text, especially in order to determine whether the writer's attitude towards a particular topic, product, etc. is positive, negative, or neutral.

This general definition encompasses more specific tasks, namely **sentiment analysis** and **emotion detection**. These tasks may aim at identifying the text's **polarity** (positive, negative, or neutral) or **discrete emotions** (e.g., anger, happiness, etc.). The same text can express a mix of emotions, so these tasks might sometimes work in a **multilabel** setting (i.e., each text is associated with more than one emotion label).

Project description

In this project, we will take on the task of **emotion detection**, where your input is a sentence, and the goal is to predict one label with the emotion expressed in that sentence.

We will use a subset of the XED dataset for multiclass emotion detection [1], which consists of English and Finnish movie subtitles from OPUS corpus[1] [2], annotated with emotional polarity and discrete emotions.

The XED dataset originally contains multilabel annotations for discrete emotions, but to keep things simpler, we will provide a single label dataset, extracted from the English dataset, where each example looks like this:

```
You will help our medical research and thereby saving thousands of lives .	8
```

The sentence and the label are separated by a **tab** character (`\t`), and the labels are mapped as follows:

[1] Note that the subset for the project is randomly sampled from the full dataset, so there is no control over what kinds of things might show up in the sets provided. Naturally, we do not endorse any content represented in the dataset.

- 1: Anger
- 2: Anticipation
- 3: Disgust
- 4: Fear
- 5: Joy
- 6: Sadness
- 7: Surprise
- 8: Trust

The project will simulate an **evaluation forum**. Evaluation forums are competitions in which participants test their systems in specific tasks, in the same conditions. Training and/or development sets are provided in advance. On a certain predefined date, a test set is released. Then, participants have a short period of time to return the output of their systems, which is evaluated and straightforwardly compared with one another, resulting in a final ranking where the state-of-the-art system is acknowledged.

Thus, we will provide the training and development sets right away, and your assignment is to create (at least) one approach that, given each sentence, will predict an emotion label from 1 to 8. You can use the training set to train your model and to inspect as you wish, and you should use the development set to have a sense of how well your model is doing, as well as to report the performance of your model in the report.

Closer to the deadline, we will release a test set **without** the emotion labels. Upon running your model on the test set, you should output a file with the original (i.e., not preprocessed) sentences followed by the label your model predicted (in the same format of the training and development sets, i.e.: `sentence\tlabel`). We will then compare that output file with the original test set and obtain the **accuracy** (using scikit-learn [3]) as part of the evaluation process.

Approaches

Just like pretty much every other NLP task, this is an active area of research, so there is no “right solution” for this problem. You can create your model however you wish, as long as you **do not look at the test set.** Possible approaches might include:

- **Rule-based** approaches (although these are likely to be weak by themselves)
- Approaches based on **additional resources** (e.g., emotion knowledge bases)
- Models based on **distances/similarity** metrics
- **Language models** (i.e., based on n-grams)
- **Machine learning** models (e.g., KNN, Naïve Bayes, SVC, etc.)

You can also extract whatever **features** you wish from the sentences, and of course, apply whatever **preprocessing** you find relevant. Note that whatever preprocessing you apply to the training set must be applied to the development set (and test set, upon release) as well!

You can try as many approaches as you wish and even mix multiple approaches into one. You only need to submit the code for the best approach (i.e., the one that obtained the best **accuracy** in the development set). You only need to report that approach against a baseline (a weaker approach you tried).

The project should be done in Python 3. Naturally, you can (and should) use available libraries, such as pandas, scikit-learn, etc. Just make sure to properly credit everything you use in the report. Also, while the approaches you try are likely to have already been tried by someone else, **you should not plagiarize anyone's code.**

Report

In addition to your code and dev/test set predictions, you should deliver a short scientific report (2 to 3 content pages + bibliography, if applicable), with the following information:

- **Group identification** (don't waste a whole page on this, please) with:
  - A short team name (to be used on the leaderboard);
  - Each member's student ID and name.
- A clearly motivated **description of your best approach**, including (when applicable):
  - The preprocessing applied;
  - The features extracted from the sentences;
  - The model/approach implemented;
- Your **experimental setup**:
  - The list of Python libraries needed to run your code, and any other implementation details that can be relevant to reproduce your experiment;
  - How you evaluated your approach:
    - Metrics that will be reported;
    - A very brief description of your baseline (if you report other intermediate approaches in the results, briefly describe them here, so that the reader knows to what your best approach is being compared).
- An **evaluation** of your best approach, which should include:
  - **Precision, Recall** and **F1** (per label and macro avg) and **Accuracy** obtained on the development set by your best approach and by your baseline (i.e., a weaker approach you tried). Feel free to include results for other intermediate approaches you tried;
  - A summary of the results reported;
  - A brief **error analysis**: what are the most common errors made by your approach? Feel free to illustrate it with a confusion matrix and/or specific examples.
- **Future work**
  - If you had more time, what would you have tried?
- **Bibliography:**
  - Citations (articles, links, etc.) of what you used (models, Python libraries, metrics, etc.)

Evaluation criteria

- **Report** (10pt):
  - Overall quality of the report (clarity, structure): **1.5pt**
  - Description of your best approach: **3pt**
  - Experimental setup: **1.5pt**
  - Evaluation - Results: **1.5pt**
  - Evaluation - Error analysis: **2pt**
  - Future work: **0.5pt**
- **Approach** (10pt):
  - Up to **5pt** for its success in the development set and 5pt for its success in the test set, as follows:
    - 2pt for code completeness / reproducibility and predictions delivery;
    - 3pt for outperforming a weaker baseline (preprocessing[2] + `CountVectorizer` + `KNN`) with a dev **accuracy of 27%;**
    - 5pt for outperforming a stronger baseline (preprocessing + `CountVectorizer` + `SVC` with a linear kernel) with a dev **accuracy** of **34%.**
  - Intermediate points might be considered depending on the overall and per label performance of your approach.

Submission

The project should be developed in groups of up to 4 members (ideally 2 to 3). Individual projects are allowed, but not recommended (unless you need that flexibility due to a worker-student status, etc.). If you need to find a group, insert your contact here.

Each group should submit a **zip** file on **Moodle.** The zip file should be named with the team's name and list of student IDs (e.g., `TheBest-id1-id2-id3-id4.zip`), and should contain the following files:

- The **code** needed to run your best approach (**Jupyter Notebook** and/or **.py** files);
- Your **report**, in **PDF** format;
- A text file, named `dev_results.txt`, with the sentences from the dev set, each sentence followed by a tab character and the label predicted by your best approach;
- A text file, named `test_results.txt`, with the sentences from the test set, each sentence followed by a tab character and the label predicted by your best approach.

Important: Make sure to keep the sentences in the same order when outputting the files mentioned above with your labels!

The project should be submitted until **June 3rd, 2022, 11:59 PM** (aka 23:59).

[2] Baselines pre-processing: Removing everything but letters, '?' and '!'; stemming; stop word removal.

Questions

Any questions about the project should be asked during **office hours**, on practical classes' **project support** periods, or on the **Questions** forum on **Moodle**. Make sure to check your colleagues' posts before posting your question – who knows if someone already asked the same question?

Students are also welcome to ask any questions during classes (if there is time).

Academic Honesty / Honor Code

Any instance of cheating diminishes the NOVA brand and is a violation of other students right to fairness and justice. Instances of cheating include:

- **Plagiarism:** NOVA has a strict policy against the deliberate reproducing of work of another person or institution without acknowledgement. All sources used for any piece of work should be fully referenced and acknowledged.
- **Unlawful Collusion:** Collusion between students in the production of materials included in the grading process, whether taking place in the classroom or at home, other than teamwork explicitly assigned by the teacher in unlawful is not allowed. Students are expected to complete their own work.
- **Unlawful Copying**: Copying from another student's examination, with or without their consent, using course material during a closed book exam, submitting the same work for more than one course without prior permission of both instructors, violating any examination recommendations or any rules relating to academic conduct of a course, unauthorized use of cell phones, calculators, dictionary, books, computer during an examination, are instances of unlawful copying.

References

1. Öhman, E., Pàmies, M., Kajava, K. and Tiedemann, J., 2020. **XED: A Multilingual Dataset for Sentiment Analysis and Emotion Detection**. In Proceedings of the 28th International Conference on Computational Linguistics (COLING 2020).
2. Lison, P., and Tiedemann, J., 2016. **OpenSubtitles2016: Extracting Large Parallel Corpora from Movie and TV Subtitles**. In Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)
3. Pedregosa et al., 2011. **Scikit-learn: Machine Learning in Python**, in Journal of Machine Learning Research 12, pp. 2825-2830.