Homework: Germeval Subtask 1

Andreas Stephan, Anastasiia Sedova, Benjamin Roth

Practical Machine Learning for Natural Language Processing

Due: Thursday, 02.07.2024, 11:30

Overview

In this homework, you will work on Subtask 1 of the GermEval 2024 Shared task (`https://ofai.github.io/GermEval2024-GerMS/`)on Sexism Detection. This task focuses on the detection of subtle forms of sexism and misogyny in German comments from an Austrian newspaper's online forum.

Project goals:

1. Develop models that can accurately identify sexism and misogyny in online comments.
2. Explore a strategy (or, if you prefer, multiple strategies) for making different predictions on handling differing opinions among annotators when labeling data.

Data Description

The training (Train.jsonl), development (Dev.jsonl), and test (Test.jsonl) datasets are available on Moodle. The is an example train file:

```
{
   "id": "99eb7b71acfeeb0ae584489218b94eac",
   "text": "Die Eignung ist und bleibt eine Bringschuld des Bewerbers.
            Man kann nicht nach dem Geschlecht auswählen, wenn man
            das Wohl der Firma vor Augen hat. Denn Nivellieren kann man
            nur nach unten.",
   "annotations": [
      {"user": "A002", "label": "2-Vorhanden"},
      {"user": "A009", "label": "0-Kein"},
      {"user": "A010", "label": "0-Kein"},
      {"user": "A012", "label": "0-Kein"}
   ]
}
```

The labeler_id's are strings. The labels have the following meanings:

- **0-Kein:** no sexism/misogyny present
- **1-Gering:** mild sexism/misogyny
- **2-Vorhanden:** sexism/misogyny present
- **3-Stark:** strong sexism/misogyny
- **4-Extrem:** extreme sexism/misogyny

Note that the test set does not contain ground truth labels. We will compute the test set performance on our side.

Task Description

The goal is to predict sexism and misogyny. Therefore you will be required to make multiple types of predictions. While you're free to solve the task however you want, we'll provide you with a "standard" solution. In the following, we describe the expected submission and then the proposed way of doing it.

Expected Submission

First, you are required to send ALL code in a .zip folder on Moodle You can submit Google Colab notebooks in the form of Jupyter Notebooks (.ipynb files) and additional python code as .py files. Your task is to provide predictions in the following format:

- `id`: the id of the example in the dataset for which the predictions are submitted
- `bin_maj`: Predict 1 if a majority of annotators assigned a label other than 0-Kein, predict 0 if a majority assigned 0-Kein. If there was no majority, either label is considered correct for evaluation.
- `bin_one`: Predict 1 if at least one annotator assigned a label other than 0-Kein, 0 otherwise.
- `bin_all`: Predict 1 if all annotators assigned labels other than 0-Kein, 0 otherwise.
- `multi_maj`: Predict the majority label if there is one, if there is no majority label, any of the labels assigned is counted as a correct prediction for evaluation.
- `disagree_bin`: Predict 1 if there is a disagreement between annotators on 0-Kein versus all other labels and 0 otherwise.

You are expected to submit those predictions in two files in a comma-separated format (.csv), one for the development set and one for the test set. Additionally, we expect a performance evaluation on the development set (as we will not hand out ground truth data for the set test) where you calculate the accuracy for each prediction type, i.e., how often your model's prediction agrees with the gold labels in the development set.

No further explanation is required if you choose to implement the standard solution. If you opt for an alternative approach to the standard solution, please provide a clear description of your solution's methodology and instructions on how to execute the code.

Standard solution

The idea is to train an individual model for all of the 5 submission prediction types. For each of the expected predictions, go with the following steps:

1. **Load data:** Begin by loading the training (Train.jsonl), development (Dev.jsonl), and test (Test.jsonl) datasets provided on Moodle. **[0.5 points]**

2. **Create per prediction Type dataset:** For each prediction type, e.g. `bin_maj`, transform the training, dev, and test split into the specific prediction type. **[1.5 points]**

3. **Train model:** Leverage the code from the HuggingFace notebook to train a model for each annotation type. If time permits, experiment with different hyperparameters or different notebooks. Hint: Google for a BERT model optimized for German. **[1.5 points]**

4. **Generate predictions:** Utilize the trained models to predict labels for the dev and test set and save the predictions. **[0.5 point]**

5. **Compute performance:** Compute the performance on the development set and save it. **[0.5 point]** If performance is better than random (60% in binary case, 25% in multi-label case) you achieve additional **[0.5 points]**.

You find inspiration for all steps in the Google Colab notebook discussed on 6.6.24 or in the Google Colab notebook discussed on 13.6.24.
After these steps are performed, aggregate the predictions into final `dev.csv` and `test.csv` files **[2.5 points]**.
Note: There are multiple possibilities to split the workload among the team. E.g. by step, by prediction type, etc.

This means in total, you can achieve $4.5 \times 5 + 2.5 = 25$ **points.** Thus the final project equals around 1-2 exercises.

Bonus Challenge

We want to encourage you to participate in the real GermEval2024 shared task. This is why the best-performing participating team in the GermEval2024 competition will be rewarded with a 50€ voucher from us! If you're interested, please talk to us directly or send us an e-mail so we can further discuss organizational questions.

Additional Resources

- https://ofai.github.io/GermEval2024-GerMS/
- https://huggingface.co/docs/transformers/en/main_classes/trainer

HAVE FUN!