Assignment 5
CSE 447/547M: Natural Language Processing

University of Washington

Due: March 15, 2019 at 11:59pm

In this assignment, you will get acquainted with basic concepts from machine translation. Instead of actually building a system that does machine translation (as we discussed in lecture, this can be an intense engineering project), you will build a classifier that can tell whether a translation was created by a human or by a machine.

Partners

For this assignment, you have the option of working with *one* other person and submitting one set of deliverables between the two of you. Feel free to utilize Piazza to find a partner.

Dataset

The data you will be using for this assignment is derived from the Quality Estimation task dataset[1] from WMT18[2], which was the 2018 iteration of the annual Workshop on Statistical Machine Translation [Specia et al., 2018]. The data consists of the following files:

- `de_en.train`: labeled sentence pairs for training.
- `de_en.dev`: labeled sentence pairs to tune your models.
- `de_en.test.jsonl`: unlabeled data in the JSON Lines format. As always, do not use the test data in any way to train your models.

The training and development instances each consist of a *source* sentence in German, a *candidate* translation of the sentence into English, and a label indicating whether the candidate comes from a machine (`M`) or a human (`H`). Each example occupies one line, with the three fields separated by tabs. As this is a nonstandard format, we provide an AllenNLP `DatasetReader` for your use, though you are free to modify it as needed. We also provide a `Predictor`, for use with the second part of this assignment.

1 Building a Classifier

Your task is to build a classifier that guesses whether a candidate sentence is a human or machine translation of a source sentence. You are free to design your model architecture however you would like. In addition to the source and candidate sentence, you are welcome to use any existing resources or tools to expand on your input. You may even use additional data, with the exception of the test data. Download the data and starter code[3] to get started.

[1] http://www.statmt.org/wmt18/quality-estimation-task.html

[2] http://www.statmt.org/wmt18

[3] https://courses.cs.washington.edu/courses/cse447/19wi/assignments/A5.tgz

We are aware that this dataset may be difficult! After all, the machine output is coming from a high-performing system. We are also aware that you don't have a lot of time to complete this assignment. As such, we will be evaluating primarily based on how well you justify your choice of model architecture. We suggest you start simple and explore a few possibilities. Your grade will not depend on your classifier's accuracy, provided that we see you put in an effort to beat the baseline.

Deliverables

1. Design a neural classifier for this task. Explain how your classifier works, including the resources and algorithms you used, and any procedures you used to estimate its performance while making empirical design decisions. Be sure to properly cite any existing tools or libraries.
2. Implement your neural classifier as an AllenNLP `Model`.
3. Create an AllenNLP configuration file that uses the provided `DatasetReader` and trains using your model.
4. Report the accuracy of your model on the training and development sets across the hyperparameter values you tested (at most 5).

For ease of grading, register your `Model` as **mt_classifier** and name the file `mt_classifier.py`. Place it in a folder called `mt_classifier`. Provide your configuration in `mt_classifier_config.jsonnet`.

2 Kaggle

As in assignment 4, we will be using Kaggle. We have created a separate competition[4] for this assignment. Your grade won't depend directly on your performance in the competition; we're doing this (1) to motivate you through friendly low-stakes competition and (2) to incentivize you to help us find out how hard this discrimination task really is.

To generate a Kaggle-ready submission, use the `kaggle.py` script included in the starter code. To create a Kaggle submission, run `python kaggle.py <test-preds.tsv> <out.csv>`. The first argument is the path to the output generated by the provided AllenNLP `Predictor`, and the second is the output filepath where the Kaggle submission file will be created. This will create a csv file with two columns, `Id` and `Category`. Each sentence pair in the test data will be assigned a unique ID and will be paired with its predicted label. This file can be submitted to Kaggle for evaluation.

Each day (in UTC[5]), you will be allowed three submssions. As discussed in assignment 4, public score and private score are calculated on different portion of test set. Before the end of the competitions, please mark on Kaggle which submission you would like to be used for evaluation (on the private test data).

Deliverables

1. Report your Kaggle account details.[6]
2. Generate a Kaggle-ready submission with your model and submit it to the competition.

Try to beat both the random and "basic" baselines with your model. If you do, describe in your writeup any changes you needed to make and your tuning procedure.

[4] https://www.kaggle.com/t/b02d16c0cd2b456aae519acdefd2879e

[5] https://www.timeanddate.com/time/aboututc.html

[6] https://goo.gl/forms/bBlRDH6WWPy5HeVq2

Submission Instructions

Run `./turnin.sh` to generate a single gzipped tarfile (`A5.tgz`) and submit it to Canvas. The script checks file naming and will fail when there is a missing file or unexpected naming. If you are submitting as a pair, only one person should submit the tarfile. The other person should submit a single text file listing the full name of both partners.

- **Code**: You will submit the config file for your best model, along with a folder containing your `Model` and the provided `DatasetReader` and `Predictor`. We assume that you always follow good practice of coding (commenting, structuring), and these factors are not central to your grade.
- **Report** (use the filename `A5.pdf` and include in the tarfile): Your writeup should be no more than two pages long, in pdf (one-inch margins, reasonable font sizes, LaTeX-typeset).
- **Kaggle**: Make sure to make submissions online. To keep everything fair, the competition will close on March 18, 2019 at 11:59pm (Pacific Time) for everyone. If you have late days, you can use them to make submissions on Kaggle. Any submissions made after you have run out of late days will not be counted. Be sure to select a submission that was made on time according to how many late days you have remaining.

Late Policy

You are free to use any and all remaining late days that you have on this assignment. If you choose to work with a partner, you will only be able to use the minimum number of remaining late days between the two of you.

References

Lucia Specia, Varvara Logacheva, Frederic Blain, Ramon Fernandez, and André Martins. WMT18 quality estimation shared task training and development data, 2018. URL http://hdl.handle.net/11372/LRT-2619. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.