CA – Assignment 4: Argument Generation

- Group: `FakeNews`
- Group members:
    - Adnan Manzoor
    - Sajjad Pervaiz
    - Kevin Taylor
    - Christoph Schäfer

Structure

```
.
└── argument-generation-assignment
    ├── Documentation.pdf
    ├── README
    ├── requirements.txt
    ├── data
    │    ├── essay_prompt_corpus.json
    │    ├── sample_predictions.json
    │    ├── predictions.json
    │    └── train-test-split.csv
    └── code
        ├── evaluation.py
        ├── page_rank.py
        └── t5.py
```

Scripts

`page_rank.py`: PageRanking sentences and selecting these for generating predictions.

`t5.py`: Uses Google's T5 to create predictions.

`evaluation.py`: Script to evaluate the `Rouge F` score.

How to run the scripts

- Make sure you have the same directory structure as above otherwise adjust the paths in the scripts accordingly.
- Run `page_rank.py` to generate the predictions in `data/` directory with name `predictions.json`. (It is important to note that running this script for the first time will lead to a download of around 800Mbs NLTK data [3])
- Run `evaluation` script with `predictions.json` as predictions.

Optional

- Run `t5.py` to generate the predictions in `data/` directory with name `predictions.json`. (Similarly to the textrank approach data of around 250mbs will be downloaded when run for the first time.)

Explanation

We started to use random sentence selection, to figure out what the lowest baseline is, with results around `0.12 rouge-1 f score`. Based on that we realized, that to reach the given baseline marginally, a not too complex method of sentence selection should suffice.

We followed up by picking sentences based on basic features we read about, that indicate a good summarising sentence. These included the longest sentence and selecting sentences based on preselected keywords (e.g. In conclusion..., The best...). This however was not successful enough, as it was in the range of the random selection results.

Finally, we tried extractive summarisation using PageRank evaluation on text, inspired by the given paper from Alshomary et.al.[2]. Therefore we use pre-trained word-embeddings to create similarity matrix and selected the best sentences based on that. This lead to a `rouge-1 f score of 0.138`, with which we reached our goal.

Additionally, we tested Google's T5 (Text-To-Text Transfer Transformer)[1], which easily produced results higher than the baseline around `0.18 rouge-1 f score`, but we just plugged in the library which didn't seem adequate for the course.

[1] https://arxiv.org/pdf/1910.10683.pdf

[2] https://webis.de/downloads/publications/papers/alshomary_2020b.pdf

[3] https://www.nltk.org/data.html