Computational Argumentation 2022

Assignment 3 – Argument Quality Assessment

Maximilian Spliethöver

mspl@mail.uni-paderborn.de

June 09, 2022

Paderborn University

Assignment 3 – Overview

Learning goals

- assessing the quality of argumentative documents automatically using text features and ML techniques

Tasks

Develop a feature-based supervised approach to automatically assess the quality of arguments in student essays on the *confirmation bias* dimension.

- *confirmation bias*: the absence of opposing arguments

Python version: Please use Python 3.8.10.

Deadline: June 20 2022, 23:59 CEST

General steps

1. Identify *relevant* textual/argumentative features
   - n-grams, POS, token stats, etc.
   - contextual features to better model the discourse
2. Choose a supervised ML model
   - SVM, Graph models, Neural networks, etc.
3. Train and evaluate the model
   - n-fold cross-validation, grid search for hyperparameter tuning, etc.
   - should finish training in a **reasonable** time!

Example: Stab and Gurevych (2016)

- annotated 402 essays with *confirmation bias/myside bias* labels
- binary classification task (feature-based, supervised)
- try different feature combinations, including:
  - word dependencies
  - opposing arguments markers (*adversative transitions*)
  - word sentiment
  - ...
- SVM architecture as ML model
- performed 10-fold cross validation on the training set

In the context of this assignment, confirmation bias means not acknowledging the other side's arguments. In other words, the absence of opposing arguments Stab and Gurevych (2016).

Other related papers: Wachsmuth et al. (2017)

Task details – Document level annotations

- address the task on the document level, with binary labels
  - **true**: the essay lacks opposing arguments
  - **false**: the essay includes opposing arguments

these 2 we need to extract from json file. n match
IDs with CSVs.- TRAIN & TEST

Now we have features n confirmation bias. what to put on X & Y.

Word2Vec or Tf-Idf as features

```
[
  {
    "id": "1337",
    "confirmation_bias": true
  },
  {
    "id": "42",
    "confirmation_bias": false
  },
  {
    "id": "23",
    "confirmation_bias": true
  }
]
```

Task details – Feature selection & model training

Feature selection

- find relevant features that represent the quality dimension
- examples include:
    - Embeddings, TF-IDF, Bag-of-Words
    - Part-of-Speech tags, Named Entity tags Word2Vec
    - Contextual features

Machine learning model/algorithm

- ML model that learns from your features to classify tokens
- binary classification task confirmation bias
- examples: SVM, Graph models, Neural Networks
- split the data into training and test sets (we provide IDs)
    - train the model on the training set
    - evaluate it on the test set

Whatever your choices, make sure to justify them properly in the documentation.
Also, don't just use word embeddings exclusively; this will not suffice!

Task details – Hyperparameter optimization

- find the best parameters for your model & features on this task
- basically to improve predictions on this task
- only include the best parameters in your submission
  - still include the code in your submission
  - it shouldn't be executed though

Whatever your results, make sure to explain them properly in the documentation.
Otherwise we cannot consider your work on this!

Assignment protocol

What you get from us

- material: example approach that addressed argument quality assessment
- annotated data: JSON file that contains annotations
- evaluation code: to evaluate your approach

What we expect from you

- the code to: extract features, train a ML model and export the results
  - please export predictions to `predictions.json`
- documentation of your chosen features, ML models and details of how you trained and evaluated (in PDF or TXT format)
  - also: any special instructions to reproduce the results

DOCUMENTATION

- ZIP file in the required format containing the two things above
  - don't re-upload the corpus files; expect them in the `./data` directory
  - filename: `ca22-assignment3_<group-name>.zip`

Assignment protocol – Submission files

Please adhere to the following directory structure and naming conventions:

```
ca22-assignment3_<group-name>.zip
├── data
│   └── <DATA-FILES>     << Expect the data files here (don't submit them).
├── main.py              << Your main code file.
├── documentation.pdf    << Your code/approach documentation.
└── predictions.json     << Predictions generated by your approach on
                            the test set (don't submit them).
```

Assignment grading

(F)
- the submitted code does not run **or** does not finish in a reasonable time
- the chosen features/model are not documented/justified at all

(B)
- documentation of the chosen features exist
- the effectiveness of your approach gets close to our baseline
- you do a hyperparameter optimization on your training data

(A)
- everything from (B)
- the selected features incorporate argumentative knowledge of the text and are well justified **and** explained in the documentation
- the chosen features and model are able to match or even beat our baseline (simply using the latest word embeddings or language models won't be enough!)

NOTICE: Code and document are checked for plagiarism. Any assignments with identified plagiarism will receive an **(F)** with further actions being taken.

Baseline scores

`F1-Score: 0.74`

accuracy(not a good measure always for unbalanced or biased data), precision, recall, f1 score (combination of precision & recall)

References

C. Stab and I. Gurevych. Recognizing the absence of opposing arguments in persuasive essays. In *Proceedings of the Third Workshop on Argument Mining (ArgMining2016)*, pages 113–118, Berlin, Germany, Aug. 2016. Association for Computational Linguistics. doi: 10.18653/v1/W16-2813. URL `https://aclanthology.org/W16-2813`.

H. Wachsmuth, G. Da San Martino, D. Kiesel, and B. Stein. The impact of modeling overall argumentation with tree kernels. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2379–2389, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1253. URL `https://aclanthology.org/D17-1253`.