Argument Quality Prediction for Ranking Documents

Notebook for the Touché Lab on Argument and Causal Retrieval at CLEF 2023

Moritz Plenz[1,†], Raphael Buchmüller[2,†] and Alexander Bondarenko[3]

[1]Heidelberg University
[2]Universität Konstanz
[3]Friedrich-Schiller-Universität Jena

Abstract

This paper presents our (team Renji Abarai) participation in Touché 2023 Task on Argument Retrieval for Controversial Questions. In our approaches submitted to the task, we investigate the influence of argument quality estimation on the overall ranking effectiveness of retrieval approaches. In particular, we train several feature-based classifiers that predict the rhetorical quality of arguments in web documents. Additionally, we use ChatGPT prompted with a few examples as a quality and stance predictor. Then, we re-rank top-10 results retrieved by ChatNoir, a BM25F-based search engine, using the predicted stance and argument quality. In total, we submit seven runs that exploit different types of quality and stance predictors. Re-ranking based on ChatGPT predictions for both, the stance and the argument quality, yields our best-performing run. Applying this re-ranking method to the official baseline outperforms all submissions to the shared task.

Keywords

Argument Quality, Stance Classification, Argument Retrieval

1. Introduction and Background

The Touché 2023 Task on Argument Retrieval for Controversial Questions was organized with the idea in mind to develop retrieval approaches that help users find relevant opinions and arguments on socially important debated topics like “Should education be free?” [1]. Specifically, the task was to rank web documents from the ClueWeb22-B dataset [2] for 50 search topics by relevance to the topic and argument quality and to detect the documents’ stances.

Previous editions of the Touché shared tasks on argument retrieval have shown that overall retrieval effectiveness can be improved by document re-ranking based on argument quality estimation [3, 4]. In our submissions to the task, we thus focus on exploiting various argument quality classifiers for ranking web documents. We also consider stance prediction to ensure that documents contain arguments.

Over the last decade, the paradigms to tackle argument quality and stance prediction have shifted dramatically. Traditionally, the respective approaches were based on manually selected linguistic features, for example, bag-of-word counts for stance detection [5, 6]. With the rise

CLEF 2023: Conference and Labs of the Evaluation Forum, September 18–21, 2023, Thessaloniki, Greece

[†]These authors contributed equally.

plenz@cl.uni-heidelberg.de (M. Plenz); raphael.buchmueller@uni-konstanz.de (R. Buchmüller); alexander.bondarenko@uni-jena.de (A. Bondarenko)

© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings (CEUR-WS.org)

CEUR-WS.org/Vol-3497/paper-264.pdf

of deep learning and big data, automatic feature extraction became possible, such as semantic embeddings from BERT [7]. Both, the manual and the automatic features, can then be used in classification models to obtain a final prediction. Recently, generative language models such as ChatGPT[1] have been exploited for various NLP tasks, including text classification.

In this paper, we investigate: (i) the effectiveness of different classifiers for argument quality prediction that are based on manually and automatically extracted features, as well as the effectiveness of prompting GPT-3.5-turbo [8], and (ii) the influence of predicting argument quality on the overall document ranking effectiveness. Our code and data are publicly available on GitHub.[2]

2. Approaches and Runs

For our runs submitted to the shared task, we re-rank top-10 results retrieved via the API of the BM25F-based search engine ChatNoir [9], using the ChatNoir-Retriever Python library[3] integrated into the PyTerrier framework [10]. To query ChatNoir, we pre-process the topic titles as follows: First, we remove punctuation and stopwords using a custom list of stopwords, since the standard existing lists (e.g., from NLTK [11]) contain terms that are important for argumentation (e.g., should, because, etc.). Afterwards, the query terms are lowercased and lemmatized with the Stanza NLP package [12].

We submitted seven runs in total via the official task submission platform TIRA [13]. Our baseline approach simply uses the results returned by ChatNoir for the pre-processed queries (topic titles). The other six submissions re-rank the baseline results based on the estimated argument quality and predicted document stance. For argument quality prediction, we train feature-based and neural classifiers using handcrafted and automatically extracted features, or prompt ChatGPT as described in Sections 2.1 and 2.2, respectively. In Section 2.3 we describe our final submissions.

2.1. Feature Extraction

This Section describes our document-level features which we extract manually or automatically as described in Sections 2.1.1 and 2.1.2, respectively. Then, Section 2.1.3 introduces our feature-based classifiers.

2.1.1. Manual Feature Extraction

We manually create a set of 32 linguistic features that can potentially serve as indicators of argument quality, as identified by existing studies. In the subsequent paragraphs, we present the implemented features, categorizing them into four distinct groups: length-based, occurrence-based, list-based, and external.

[1]https://openai.com/blog/chatgpt
[2]https://github.com/Heidelberg-NLP/renji_abarai
[3]https://github.com/chatnoir-eu/chatnoir-pyterrier

Length. The lengths of certain pieces of text can hint at their argument quality. The varying length of paragraphs, words, or the document itself can be linked to the argumentation's completeness, clarity, complexity, evidence, and coherence [14, 15].

As an initial step, we extract the length of the **document** by calculating the number of characters that may indicate the document's completeness. Longer documents allow for comprehensive analysis, addressing multiple perspectives, and providing ample supporting evidence, strengthening the argument [16]. Then, we split the text into **paragraphs** using line breaks and compute the average sentence length (in characters) within each paragraph. Examining the average sentence length within each paragraph can leverage insights into the document's structure and clarity [17]. We also calculate the average length of **sentences** in the whole document. Longer sentences may demonstrate higher complexity and detailed reasoning, while shorter sentences can convey information more succinctly and maintain the reader's attention. Finally, we compute the average **word** length in the text. Longer words may suggest technical or specialized vocabulary, indicating a more advanced level of discourse. On the other hand, shorter words may contribute to simplicity and accessibility.

Occurrences. We further count the number of occurrences of selected characters or words relative to the overall text size to asses rhetorical impact and linguistic style as possible indicators of the argument's effectiveness.

We calculate the number of **stopwords** drawn from the NLTK [11] stopword list, including common articles, conjunctions, and prepositions, as possible hints towards lexical choices, grammatical structure, and overall text complexity. The **punctuation marks**, **numerics**, and **external links** are counted to grasp the texts' linguistic style, use of rhetorical devices, and logical reasoning. For similar reasons, we extract the number of words that start with an **uppercase** letter and the ones that are written **fully uppercase**, as well as **sentences** starting with an uppercase letter to check for a more formal style of writing. Connected to that, we include the average number of sentence types for a **question**, **exclamation**, and **statements**.

Word lists. We furthermore determine features based on selected word lists to assess the argument's sophistication, professionalism, and appropriateness. In particular, we use the **academic word** list [18] and **profanity words** [19] to estimate the levels of formality, precision, and adherence to scholarly discourse. We extract keywords from the **original question** and count their appearances in the respective arguments to examine the accordance between the question and argument. Also, we check for **vocabulary richness** with and **without stopwords** by determining the ratio of different words per text. Finally, we compile a list of **argumentative words** using ChatGPT-3.5 and manual modification based on our own experience.

External. Subsequently, we extend our feature set by incorporating external sources with additional features of higher complexity. We use the neural argument mining framework TARGER [20][4] to extract the **number of arguments** (number of tokens tagged as a claim or a premise) from the given texts and the SentimentIntensityAnalyzer from the rule-based model Vader [21] to assess **subjectivity** and **sentiment** to grasp affective and subjective aspects that

[4]https://demo.webis.de/targer-api/apidocs/ with the "tag-essays-dependency" model.

can influence an argument's effectiveness and credibility. We include **7 readability** measures and the number of **spelling mistakes** for similar reasons. We estimate readability using the state-of-the-art measures like Flesch [22], Flesch-Kincaid [23], Gunning Fog [24], Coleman Liau [25], Dale Chall [26], Automated Readability Index [27], Linsear Write [28], and Spache [29], all implemented in the NLTK library [11]. For checking grammar and spelling mistakes, we use `hunspell` [30]. Incorporating the mentioned features can provide insights into an argument's accessibility, clarity, and overall quality.

One could also integrate features focusing on external knowledge bases [31, 32] such as ConceptNet [33], but for this work we restrict our analysis to more language-oriented features.

2.1.2. Automatic Feature Extraction

To automatically extract features from documents, we use an instruction-based fine-tuned embedding model INSTRUCTOR [34] that allows the creation of task- and domain-specific embeddings. The model was trained on 30 different datasets spanning a wide range of tasks (e.g., retrieval, clustering, classification, etc.) and domains (e.g., science, finance, etc.) and achieved state-of-the-art performance on the 70 diverse benchmarks, improving over the previous best results. To deal with long documents, we split them into sentences using the NLTK's sentence tokenizer, represent each sentence with the INSTRUCTOR-XL model[5] from Hugging Face [35], and calculate the mean of the values across each feature of the sentence embeddings array. Each sentence is passed to the model using the following format: `<Instruction>: <Input>` with the instruction text "Represent a sentence for argumentation quality classification".

2.1.3. Classifiers for Argument Quality

To train classifiers that predict the argument quality of documents, we use manual labels from the Touché 2021 Task 1 ('low / no arguments', 'medium', and 'high' quality) [3] and split the data into the train (3,000 samples) and validation (700 samples) sets. To account for the discrepancy between the task topics in 2021 (used for training classifiers) and 2023 (used for the actual prediction), we ensure that all the manually judged documents retrieved for one topic are in the same split. By doing so, we tackle the so-called cross-topic argument quality prediction. For both feature types, manually and automatically extracted, we use the same 6 classification models: feedforward neural network (FNN) with 3 hidden layers, LightGBM [36], logistic regression, naïve Bayes, SVM, and random forests. The classifiers' hyperparameters[6] are optimized with a grid search in a five-fold cross-validation on the train set.

The classifiers predict both, the class labels of argument quality and the probabilities of each class, on the validation set. We further train a metalearner classifier on the predictions of individual classifiers on our validation set. By testing different models and feature sets, the most effective metalearner is based on random forests with feature vectors combining class predictions and probabilities from all the 6 classification models trained on manual features and embeddings (i.e., 12 classifiers in total). The metalearner's hyperparameters are optimized with a grid search in a five-fold cross-validation on the validation set.

---

[5] https://huggingface.co/hkunlp/instructor-xl

[6] Available on GitHub: https://github.com/Heidelberg-NLP/renji_abarai

In Section 2.3, we provide a detailed description of how we use the argument quality predictions to re-rank the top-10 initially retrieved documents.

2.2. Prompting ChatGPT

Using few-shot prompting, we predict the argument quality of a given document with ChatGPT[7]. Our prompt consists of a brief instruction as well as 3 examples per label (i.e., 9 examples in total). We do *not* include queries in the prompt, as we assume argument quality to be independent of the query. The instruction gives a description of argument quality, which is taken from the shared task website.[8] Partially we slightly modified some examples (taken from our train set; see Section 2.1.3) to make them clearer or shorter. The examples are presented as a message history, i.e., the input to ChatGPT looks as if it provided the previous correct quality labels. Following the annotation guidelines, we ask ChatGPT to predict the quality as 'high', 'medium', or 'low'.[9]

Sampling. Since the ChatGPT API from OpenAI only returns the generated text, but not the corresponding logits, we cannot use the logits to infer the class probabilities. To alleviate this issue, we predict the quality of each document 3 times and aggregate the (potentially different) predictions into one final score. The concrete aggregation depends on each specific run and is described in Section 2.3.[10]

Context length. The prompt and document often exceed the maximum context length of ChatGPT.[11] Therefore, we truncate 218 out of 500 initially retrieved documents. We truncate documents such that they are at most 10,000 characters. Where possible, we truncate documents such that they end after a paragraph. We hypothesize that this makes truncated documents more natural, as they do not end in the middle of a paragraph, sentence, or even word.

Manual curation. In about 16 % of the cases, ChatGPT does not return one of the desired labels, but generates a different text. Hence, we manually curate these text spans according to the following two rules: (i) If it is clear what label ChatGPT assigned, then we choose that label, (ii) otherwise we assign the label 'low'. The assigned label is clear if the generated text is a label as well as additional text (usually an explanation of the label).[12]

Stance prediction. Similar to argument quality prediction, we use ChatGPT to predict the stance of documents given a query (the query is now included in the prompt). Here, the possible

---

[7] https://openai.com/blog/chatgpt, accessed via the API. Version `gpt-3.5-turbo-0301`
[8] https://touche.webis.de/
[9] The complete prompt is at https://github.com/Heidelberg-NLP/renji_abarai/blob/main/prompt/quality.txt.
[10] At the time of our experiments, one could not set a temperature parameter.
[11] At the time of our experiments: 4,096 tokens.
[12] For example, "The article presents both sides of the debate about homework very well, discussing the pros and cons of assigning homework to students. It also provides alternatives to homework, such as the Exercise Your Brain program implemented by P.S. 118, and highlights the positive feedback from parents and students alike. Overall, the article seems well written and informative, so I would say it has a high rhetorical argument quality."

labels for ChatGPT are 'pro', 'con', 'neutral', and 'none' (i.e., a text does not take any stance because it, for instance, contains only facts but not arguments). We provide one instance as an example per label. Due to the lack of examples for 'neutral' and 'none' in our train set, we write a 'neutral' example ourselves and take the first paragraphs from a Wikipedia article (that contain some definitions, but no arguments) as 'none'.[13] We again manually curate the predictions, where we assign the 'none' label if no clear label was predicted.

For stance, we again collect from ChatGPT 3 predictions per document. To aggregate the stance label we (i) take the majority label. If there is no majority we (ii) assign 'none' if 'none' was predicted once, and (iii) 'neutral' otherwise.

2.3. Submission Description

For all our submissions, we first retrieve 10 documents per query with ChatNoir using lemmatization and custom stopwords removal (see Section 2). Then, we re-rank these documents by different criteria, which are described below for each run. If our re-ranking strategy results in a tie, then documents are ranked by the relevance score obtained from the initial retrieval with ChatNoir. All submissions use the stance predicted by ChatGPT (see Section 2.2).

Run ***baseline***: simply uses the ChatNoir results when queried with the topic titles after removing customized stopwords and lemmatization.

Run ***meta_qual_score***: top-10 initially retrieved documents are re-ranked based on the predicted numerical quality labels using a metalearner (sorted by the quality labels in descending order).

Run ***meta_qual_prob***: analogous to *meta_qual_score* but the documents with the same quality label are sorted by the class probability of the metalearner.

Run ***ChatGPT_mmEQhl***: documents are re-ranked based on the quality predictions from ChatGPT. To aggregate the predictions from the 3 samples, we convert the discrete labels to scalars: 0, 1, and 2 for 'low', 'medium', and 'high' quality, respectively. Then, we take the average over all 3 predictions.

Run ***ChatGPT_mmGhl***: analogous to *ChatGPT_mmEQhl*, except that the 'medium' quality prediction has a weight of 1.1. This means that documents where 'medium' is predicted twice are ranked higher than documents where 'low' and 'high' quality is predicted once (assuming that the third prediction is the same in both cases).

Run ***stance_ChatGPT***: analogous to *ChatGPT_mmEQhl* but we additionally consider the stance prediction. Documents with stance 'none' include by definition no arguments and are therefore of low argument quality. Thus, we first (i) place the stance 'none' documents at the bottom of the ranking, and then, (ii) rank the remaining documents based on the predicted argument quality.

Run ***stance-certainNO_ChatGPT***: analogous to *stance_ChatGPT*, except that documents are only considered to have no stance if that is predicted by the majority.

[13]The complete prompt is at https://github.com/Heidelberg-NLP/renji_abarai/blob/main/prompt/stance.txt.

3. Evaluation Results

Following the shared task setting,[14] we evaluate our approaches for the *rhetorical argument quality* and the *general topic relevance*, both measured with nDCG@10 [37]. We also report the macro-averaged F1 scores for our stance prediction (four stance classes). Additionally, we evaluate the stance prediction in a binary fashion, where one class is 'none' (i.e., a document does not take any stance) and the other class is any stance (i.e., 'pro', 'con' or 'neutral'). We report this value since our runs *stance_ChatGPT* and *stance-certainNO_ChatGPT* depend on the quality of binary stance prediction.

None of our initially submitted runs to the task outperforms the official task baseline in terms of relevance and argument quality (see the lower part (*Ours*) and the middle part (*BL*) of Table 1). The task's official retrieval baseline uses the results that the search engine ChatNoir [9] returned for the topics' titles used as queries without any pre-processing, i.e., it is an argumentation-agnostic BM25F-based retrieval system. The worse effectiveness of our runs is likely due to the poor performance of our initial retrieval (run 'baseline' in Table 1; details on the run are in Section 2.3) that was used in the re-ranking step.

First, we analyze our initial submissions to the task in Section 3.1. Then, Section 3.2 describes the results when applying our most promising re-ranking strategies to the official task baseline.

3.1. Evaluation of the Submitted Runs

Our most effective initial submission ('stance_ChatGPT' run) exploits ChatGPT to predict the argument quality and stance. Then, a two-step re-ranking strategy is applied to our 'baseline' run: (i) move the 'no stance' documents to the bottom of the ranked list and then (ii) re-rank the remaining documents based on the predicted argument quality in descending order. Overall, while almost all of our re-ranking strategies improve over our baseline run, statistically significant improvements are achieved only by two ChatGPT-based re-ranking approaches and only quality-wise (see Table 1).

As for the stance prediction, our submitted results achieved a macro-averaged F1 of 0.599, which outperforms the official task baseline stance detector (Flan-T5 model [38] in zero-shot settings). Note that these values should not be directly compared as they are evaluated on different retrieved documents. However, when applying our stance classifier on the documents retrieved by the task baseline, we achieve a macro F1 of 0.556, which is more than twice the official task baseline effectiveness.

For our stance-based re-ranking techniques, we only consider whether (i) the document contains arguments (i.e, classes 'pro', 'con', and 'neutral') or whether (ii) the document does not contain any arguments (i.e., class 'none'). Thus, we also calculate the macro F1 scores in a binary classification fashion. We observe that ChatGPT consistently achieves scores higher than 0.75, which may explain the high effectiveness of re-ranking approaches using the predicted stance.

[14] https://touche.webis.de/clef23/touche23-web/argument-retrieval-for-controversial-questions.html

Table 1
Evaluation results of our submissions to the task (*Ours*), the official task baseline (*BL*), and our ChatGPT-based re-ranking approaches applied to the official task baseline (*Ours w/ BL*; experiments were conducted after the task was officially ended). The dagger † indicates a statistically significant improvement ($p < 0.05$, Bonferroni corrected) over our baseline run. For the stance scores marked by $^{(*)}$ we aggregate the 3 stance predictions by assigning the majority label if it exists and assigning 'neutral' otherwise. Thus, these values are *not* as in our official submission described in Section 2.2. We also evaluated stance prediction in a binary (*bin.*) fashion: either any stance or no stance.

		nDCG@10		macro F1	
	Configuration (run)	Quality	Relevance	Stance	Stance bin.
Ours w/ BL	stance_ChatGPT	0.840	0.841	0.556	0.754
	stance-certainNO_ChatGPT	0.840	0.842	$0.557^{(*)}$	$0.762^{(*)}$
	ChatGPT_mmGhl	0.834	0.833	0.556	0.754
	ChatGPT_mmEQhl	0.834	0.832	0.556	0.754
BL	ChatNoir [9] / Flan-T5 (stance) [38]	0.831	0.834	0.203	0.432
Ours	stance_ChatGPT	$0.815^{\dagger}$	0.744	0.599	0.780
	stance-certainNO_ChatGPT	$0.811^{\dagger}$	0.746	$0.604^{(*)}$	$0.783^{(*)}$
	ChatGPT_mmGhl	0.789	0.718	0.599	0.780
	ChatGPT_mmEQhl	0.789	0.718	0.599	0.780
	meta_qual_prob	0.774	0.697	0.599	0.780
	meta_qual_score	0.771	0.712	0.599	0.780
	baseline	0.766	0.708	0.599	0.780

3.2. Improving the Task Baseline

In the post-task evaluation, we apply our most promising ChatGPT-based re-ranking strategies to the official task baseline run. Again, our re-ranking consistently improves over the baseline, although the improvements are not statistically significant (see the upper part (*Ours w/ BL*) of Table 1). Overall, the most effective systems combine the official task baseline with our stance-based re-ranking approaches *stance_ChatGPT* and *stance-certainNO_ChatGPT*, resulting in an nDCG@10 score of about 0.84 for both quality and relevance dimensions.

In general, while our re-ranking strategies consider only the argument quality estimation and predicted stance (unless there are ties), they also improve the relevance-based ranking effectiveness. This observation aligns with the findings of previous Touché iterations [3, 4]. A possible reason is that documents that contain high-quality arguments tend to be perceived as more relevant by human annotators. A precise investigation of the reasons for this observation might be an interesting avenue for future work.

4. Conclusions and Future Work

In this paper, we described our participation in the Touché shared task on argument retrieval for controversial questions. With our proposed approaches, we aimed to investigate the influence of ranking documents based on the predicted argument quality and stance on the overall document ranking effectiveness. Thus, we proposed to re-rank initially retrieved documents with BM25F by boosting the position in the ranked list of documents that have a higher rhetorical argument quality and by penalizing documents that do not take any stance.

While none of our approaches submitted to the task were able to outperform the official task baseline, our re-ranking strategies improved the retrieval effectiveness of our initial ranked list of documents. These improvements, however, were not sufficient to reach the task baseline effectiveness, due to the much worse performance of our first-stage retrieval. We also found that ChatGPT used for argument quality prediction contributed to better improvements than our trained feature-based classifiers. Additionally, we applied our most promising ChatGPT-based re-ranking strategies to the official task baseline result. Our re-ranking consistently improved over the baseline, although the improvements were not statistically significant.

Our experiments showed the potential of including argument quality and stance predictions into retrieval pipelines for queries that address controversial topics and thus request high-quality argumentation in retrieved documents. Interesting directions for future work might be improving the effectiveness of argument quality and stance classifiers as well as developing more sophisticated re-ranking approaches.

Acknowledgments

This work has been partially supported by the Deutsche Forschungsgemeinschaft (DFG) in the projects "ACQuA 2.0: Answering Comparative Questions with Arguments" (project 376430233) and "ACCEPT: Prespectivized Argument Knowledge Graphs for Deliberation" as part of the priority program "RATIO: Robust Argumentation Machines" (SPP 1999).

References

[1] A. Bondarenko, M. Fröbe, J. Kiesel, F. Schlatt, V. Barriere, B. Ravenet, L. Hemamou, S. Luck, J. Reimer, B. Stein, M. Potthast, M. Hagen, Overview of Touché 2023: Argument and causal retrieval, in: Working Notes of CLEF 2023, CEUR Workshop Proceedings, CEUR-WS.org, 2023.

[2] A. Overwijk, C. Xiong, J. Callan, ClueWeb22: 10 billion web documents with rich information, in: Proceedsings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2022), Association for Computing Machinery, 2022, pp. 3360–3362. URL: https://doi.org/10.1145/3477495.3536321.

[3] A. Bondarenko, L. Gienapp, M. Fröbe, M. Beloucif, Y. Ajjour, A. Panchenko, C. Biemann, B. Stein, H. Wachsmuth, M. Potthast, M. Hagen, Overview of Touché 2021: Argument Retrieval, in: Proceedings of the 12th International Conference of the CLEF Association, CLEF 2021, volume 12880 of *Lecture Notes in Computer Science*, Springer, 2021, pp. 450–467. URL: https://doi.org/10.1007/978-3-030-85251-1_28.

[4] A. Bondarenko, M. Fröbe, J. Kiesel, S. Syed, T. Gurcke, M. Beloucif, A. Panchenko, C. Biemann, B. Stein, H. Wachsmuth, M. Potthast, M. Hagen, Overview of Touché 2022: Argument Retrieval, in: Proceedings of the 13th International Conference of the CLEF

Association, CLEF 2022, volume 13390 of *Lecture Notes in Computer Science*, Springer, 2022, pp. 311–336. URL: https://doi.org/10.1007/978-3-031-13643-6_21.
[5] D. Küçük, F. Can, Stance detection: A survey, ACM Comput. Surv. 53 (2020). URL: https://doi.org/10.1145/3369026. doi:10.1145/3369026.
[6] M. Hardalov, A. Arora, P. Nakov, I. Augenstein, A survey on stance detection for mis- and disinformation identification, in: Findings of the Association for Computational Linguistics: NAACL 2022, Association for Computational Linguistics, Seattle, United States, 2022, pp. 1259–1277. URL: https://aclanthology.org/2022.findings-naacl.94. doi:10.18653/v1/2022.findings-naacl.94.
[7] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. URL: https://aclanthology.org/N19-1423. doi:10.18653/v1/N19-1423.
[8] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, Advances in neural information processing systems 33 (2020) 1877–1901.
[9] J. Bevendorff, B. Stein, M. Hagen, M. Potthast, Elastic ChatNoir: Search engine for the ClueWeb and the Common Crawl, in: Proceedings of the 40th European Conference on IR Research, (ECIR 2018), volume 10772 of *Lecture Notes in Computer Science*, Springer, 2018, pp. 820–824. URL: https://doi.org/10.1007/978-3-319-76941-7_83.
[10] C. Macdonald, N. Tonellotto, S. MacAvaney, I. Ounis, PyTerrier: Declarative experimentation in Python from BM25 to dense retrieval, in: Proceedings of the 30th ACM International Conference on Information and Knowledge Management, CIKM 2021, ACM, 2021, pp. 4526–4533. URL: https://doi.org/10.1145/3459637.3482013.
[11] S. Bird, E. Klein, E. Loper, Natural Language Processing with Python, O'Reilly, 2009. URL: http://www.oreilly.de/catalog/9780596516499/index.html.
[12] P. Qi, Y. Zhang, Y. Zhang, J. Bolton, C. D. Manning, Stanza: A python natural language processing toolkit for many human languages, in: A. Celikyilmaz, T. Wen (Eds.), Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Association for Computational Linguistics, 2020, pp. 101–108. URL: https://doi.org/10.18653/v1/2020.acl-demos.14.
[13] M. Fröbe, M. Wiegmann, N. Kolyada, B. Grahm, T. Elstner, F. Loebe, M. Hagen, B. Stein, M. Potthast, Continuous integration for reproducible shared tasks with TIRA.io, in: Proceedings of the 45th European Conference on IR Research, ECIR 2023, Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2023, pp. 236–241.
[14] A. Wyner, R. Mochales-Palau, M.-F. Moens, D. Milward, Approaches to text mining arguments from legal cases, Springer, 2010.
[15] R. Swanson, B. Ecker, M. Walker, Argument mining: Extracting arguments from online dialogue, in: Proceedings of the 16th annual meeting of the special interest group on discourse and dialogue, 2015, pp. 217–226.
[16] H. Wachsmuth, T. Werner, Intrinsic quality assessment of arguments, in: Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, International

Committee on Computational Linguistics, 2020, pp. 6739–6745. URL: https://aclanthology.org/2020.coling-main.592.
[17] G. H. Poll, C. A. Miller, J. G. Van Hell, Sentence repetition accuracy in adults with developmental language impairment: Interactions of participant capacities and sentence structures, Journal of Speech, Language, and Hearing Research 59 (2016) 302–316.
[18] A. Coxhead, A new academic word list, TESOL quarterly 34 (2000) 213–238.
[19] L. von Ahn, et al., Offensive/profane word list, Retrieved June 24 (2009) 2018.
[20] A. N. Chernodub, O. Oliynyk, P. Heidenreich, A. Bondarenko, M. Hagen, C. Biemann, A. Panchenko, TARGER: Neural argument mining at your fingertips, in: Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Association for Computational Linguistics, 2019, pp. 195–200. URL: https://doi.org/10.18653/v1/p19-3031.
[21] C. J. Hutto, E. Gilbert, VADER: A parsimonious rule-based model for sentiment analysis of social media text, in: Proceedings of the Eighth International Conference on Weblogs and Social Media, ICWSM 2014, The AAAI Press, 2014, pp. 216–225. URL: http://www.aaai.org/ocs/index.php/ICWSM/ICWSM14/paper/view/8109.
[22] R. Flesch, A new readability yardstick., Journal of applied psychology 32 (1948) 221.
[23] R. Flesch, Flesch-kincaid readability test, Retrieved October 26 (2007) 2007.
[24] J. Bogert, In defense of the fog index, The Bulletin of the Association for Business Communication 48 (1985) 9–12.
[25] M. Coleman, T. L. Liau, A computer readability formula designed for machine scoring., Journal of Applied Psychology 60 (1975) 283.
[26] E. Dale, J. S. Chall, A formula for predicting readability: Instructions, Educational research bulletin (1948) 37–54.
[27] R. Senter, E. A. Smith, Automated readability index, Technical Report, Cincinnati Univ OH, 1967.
[28] L. Si, J. Callan, A statistical model for scientific readability, in: Proceedings of the tenth international conference on Information and knowledge management, 2001, pp. 574–576.
[29] G. Spache, A new readability formula for primary-grade reading materials, The Elementary School Journal 53 (1953) 410–413.
[30] J. Ooms, hunspell: High-Performance Stemmer, Tokenizer, and Spell Checker, 2023.
[31] M. Plenz, J. Opitz, P. Heinisch, P. Cimiano, A. Frank, Similarity-weighted construction of contextualized commonsense knowledge graphs for knowledge-intense argumentation tasks, in: Proceedings of the 61th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Toronto, Canada, 2023.
[32] P. Heinisch, M. Plenz, A. Frank, P. Cimiano, Accept at semeval-2023 task 3: An ensemble-based approach to multilingual framing detection, in: Proceedings of the 17th International Workshop on Semantic Evaluation (SemEval-2023), Association for Computational Linguistics, Toronto, Canada, 2023.
[33] R. Speer, J. Chin, C. Havasi, Conceptnet 5.5: An open multilingual graph of general knowledge, in: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI'17, AAAI Press, 2017, p. 4444–4451.
[34] H. Su, W. Shi, J. Kasai, Y. Wang, Y. Hu, M. Ostendorf, W. Yih, N. A. Smith, L. Zettlemoyer, T. Yu, One embedder, any task: Instruction-finetuned text embeddings, CoRR

abs/2212.09741 (2022). URL: https://doi.org/10.48550/arXiv.2212.09741.

[35] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, A. M. Rush, Transformers: State-of-the-art natural language processing, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020, Association for Computational Linguistics, 2020, pp. 38–45. URL: https://doi.org/10.18653/v1/2020.emnlp-demos.6.

[36] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, T. Liu, LightGBM: A highly efficient gradient boosting decision tree, in: Proceedings of the Annual Conference on Neural Information Processing Systems, NeurIPS 2017, 2017, pp. 3146–3154. URL: https://proceedings.neurips.cc/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html.

[37] K. Järvelin, J. Kekäläinen, Cumulated gain-based evaluation of IR techniques, ACM Trans. Inf. Syst. 20 (2002) 422–446. URL: http://doi.acm.org/10.1145/582415.582418.

[38] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, E. Li, X. Wang, M. Dehghani, S. Brahma, A. Webson, S. S. Gu, Z. Dai, M. Suzgun, X. Chen, A. Chowdhery, S. Narang, G. Mishra, A. Yu, V. Y. Zhao, Y. Huang, A. M. Dai, H. Yu, S. Petrov, E. H. Chi, J. Dean, J. Devlin, A. Roberts, D. Zhou, Q. V. Le, J. Wei, Scaling instruction-finetuned language models, arXiv (2022). doi:10.48550/arXiv.2210.11416.