Creating an Argument Search Engine for Online Debates

Notebook for the Touché Lab on Argument Retrieval at CLEF 2020

Maximilian Bundesmann[1], Lukas Christ[2], and Matthias Richter[3]

[1] University of Leipzig, Germany
mb74fawu@studserv.uni-leipzig.de
[2] University of Leipzig, Germany
lc85futa@studserv.uni-leipzig.de
[3] University of Leipzig, Germany
mr75syri@studserv.uni-leipzig.de

Abstract. Consulting web search engines has become an everyday procedure for many internet users. One specific task that gained attention in recent work is the retrieval of arguments for controversial topics. Most of the preexisting difficulties that search engines have to face also apply for this task. However, certain challenges become even more important, such as providing an appropriate heterogeneity in the result set. We present an argument search engine for the argsme corpus. Our focus is on preprocessing the corpus while also addressing the heterogeneity problem and implementing a query expansion feature. Furthermore, we provide a brief evaluation of our retrieval results.

Keywords: Information Retrieval · Argumentative Conversations · Online Debates · Argument Search

1 Introduction

Nowadays, search engines are used by everybody. Consulting a search engine is the easiest way to find desired information like today's weather, news articles or just any arbitrary image. However, there are still some problems for which modern search engines fail to deliver satisfying answers yet. One of these challenges is the search for arguments in large document collections, e.g. for debates such as *"Are plastic bottles good?"* or *"Are speed limits wrong?"*. Search engines that find arguments for these kinds of queries could be classified as *argument search engines*. There are already some solutions available, e.g. Args.me [22] or ArgumenText [20].

This report is created in the context of the Touché shared task on argument retrieval [5]. Its goal is to develop an argument search engine that retrieves arguments from the argsme corpus [2], which provides almost 390,000 arguments from over 55,000 online debates. We develop a search engine to find good arguments in the corpus. This report describes our approach and evaluates its performance.

First of all in Section 2, we summarize the state of the art of information retrieval in the context of argument search engines. Section 3 introduces our search engine's architecture. Following, we give a short overview of the corpus, describe some necessary preprocessing steps and show the ideas behind all separate components. In Section 4 we present the results of the final evaluation.

Argument search refers to collecting relevant premises and conclusions to a given topic that is usually of controversial nature. The goal of such a search engine is to provide the user with supported statements that help him to gather knowledge about his topic of interest and potentially assist his decision making.

2 Related Work

Previous work has been carried out that tackled various tasks of argument search, including automatically detecting evidence that support a given claim [19], determining argument relevance [17] or acquiring a corpus of arguments [1]. The latter one along with the work presented by Wachsmuth et al. [22] constitutes the basis for this work.

2.1 Retrieval Models

The heart of an argument search engine is a proper retrieval model. The challenge here is to find the best arguments w.r.t. a corpus and a free-text query.

Several argument search engines are e.g. args.me [22] or ArgumenText [20]. These engines are based on the retrieval model Okapi BM25. Potthast et al. [17] performed a user study to evaluate different well-known retrieval models. In this examination Lucene's BM25, Terrier's implementations of DPH [3], DirichletLM [25], and TFIDF were considered. The retrieved arguments by the different retrieval models were rated by their relevance, rhetoric, logic, and dialectic quality. DPH proved to yield the best results overall.

2.2 Preprocessing

Predicting argument quality is a challenging task. Wachsmuth et al. [21] discuss the concept of argument quality. Wei et al. [24] rank argumentative reddit posts in order to find the most persuasive ones. Their approach is machine learning-based. The same applies for the approach proposed by Persing and Ng [16]. Potthast et al. [17] provide a subset of the argsme corpus in which arguments were manually annotated with a rating for the quality aspects defined by Wachsmuth et al. [21]. Furthermore, an overall quality rating was assigned to each argument.

2.3 Query Expansion

Several different approaches have been explored that aim to improve the recall for user queries [8]. Some examples are pseudo-relevance feedback (using terms from the top ranked documents) or interactive query refinement that requires the user to readjust his query. Another option is to use search query logs to obtain rewritings that users perform to improve their query terms with respect to their information need. However, some of these methods are out of the scope for this work, or require additional data. For instance, search query logs are unavailable for this task. Therefore, we focus on a few automatic query expansion (AQE) methods based on word embeddings. Diaz et al. [12] and Zuccon et al. [26], for instance, utilized such "model-based" approaches.

2.4 Clustering

The aim of clustering here is to guarantee a diverse result set in order to present the user a variety of different arguments. Carbonell and Goldstein [7] introduce Maximal Marginal Relevance (MMR), a measure that allows building a ranking incrementally. MMR balances the quality and heterogeneity of the result set.

Another approach to build a diverse ranking incrementally is proposed by Kaptein et al. [15]. Deselaers et al. [11] describe a method to diversify image search results using a "novelty" measure. A problem in diversifying search results is that classic evaluation measures like (n)DCG do not take diversity of results into account. Clarke et al. [9] thus propose an alternative evaluation framework.

3 Methods

Our argument search engine's architecture is depicted in Figure 1. The central module is the Apache Lucene Core[4] search library that realizes indexing and retrieval. Before indexing, the corpus is preprocessed. During preprocessing, quality ratings for all documents are computed.

At run time, the user's queries are enriched by a query expansion module. Then, after retrieving a set of relevant documents via the Lucene Core, results are ranked considering both the scores obtained by the retrieval system and the quality ratings. Eventually, the last component can perform clustering on the top ranked results.

3.1 Preprocessing

The quality of the arguments contained in the corpus is heterogeneous. Some documents do not contain arguments at all. Therefore we aim to assign ratings to the documents indicating their argumentative quality. More formally, we create a mapping

$$q : D \to [0, 1] \tag{1}$$

[4] https://lucene.apache.org/

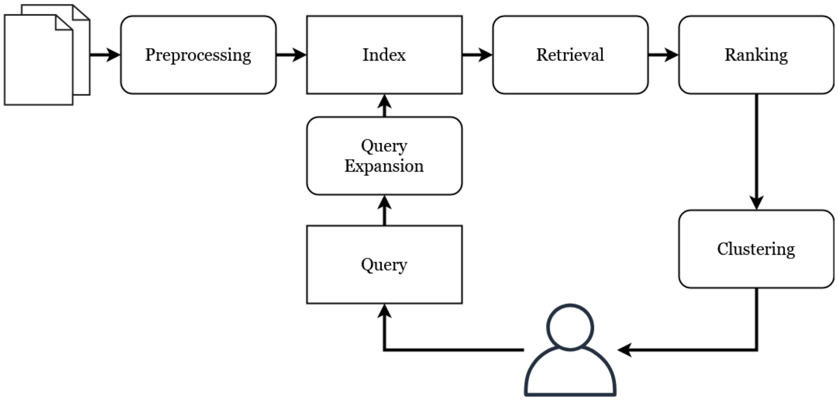

Fig. 1. All components of the introduced pipeline can be changed independently.

where D is the corpus as a set of documents. If $q(d_1) > q(d_2)$, the argumentative quality of d_1 is considered higher than that of d_2.
These ratings are then used in our retrieval model, as the user should only receive arguments with high quality. To compute them, we employ a machine learning approach. Argument quality is a rather elusive concept that can not be quantified directly. Wachsmuth et al. [21] break it down into three aspects:

- *Logical quality*: are the premises acceptable and do they really imply the conclusion?
- *Rhetorical quality*: is the argument formulated in a persuasive manner?
- *Dialectical quality*: does the argument contribute to resolving the issue?

Capturing the logical dimension of argument quality with computational features is a hard task. Solving it is beyond the scope of this project.The dialectical quality dimension is not available in our corpus either. Other than the corpus of reddit posts used by Wei et al. [24], the argsme corpus does not contain information about replies to a post or citations of a post. Thus, the only quality dimension we aim to quantify is rhetorical quality.
To achieve this, we compute 22 features for each argument. Most of them can already be found in Wei et al. [24] and Persing and Ng [16]. In the following, the features are briefly described.
Linguistic competence features aim at quantifying the argument's author's linguistic skills: *average sentence length, word length*, *type/token ratio*, *number of punctuation marks per sentence*, *different POS-Tags (whole text)*, *conjunctives per sentence*, *modal verbs per sentence*, *emojis* (Use of emojis might coincide with rather colloquial language and lack of seriousness), *non-stopwords ratio.*

Sources and Examples: claims are more persuasive when they are supported by examples and sources. The following rule-based features are intended to capture them: *number of references per sentence*, *examples per sentence*, *URLs per sentence*, *percentages per sentence*, *year specifications per sentence*

Subjectivity, ad hominem and emotionality: Arguments are more persuasive when they are presented in an objective manner, without anecdotal evidence or attacking the opponent personally. We aim to quantify subjectivity and emotionality with the following features:

- *number of first person pl. pronouns per sentence* indicate subjectivity. We do not count first person singular words. Persing and Ng [16] argue that objective arguments frequently start with phrases like "I think..." or "I believe...", too.
- *number of second person pronouns per sentence* may indicate personal attacks
- *Sentiment Analysis* is able to indicate high emotionality. We use VADER (Hutto et al. [14]).
- *hedge words/phrases per sentence*: these phrases may indicate a more polite, indirect and differentiated formulation. We use a list[5] to identify such phrases.
- *number of definite articles / number of articles*: Persing and Ng [16] argue that a lack of definite articles often means a lack of specifity and objectivity
- *average concreteness*: Brysbaert et al. [6] provide ratings for word concreteness obtained by crowd sourcing. This feature describes the average degree of abstractness/concreteness in the argument.
- *components of emotions*: Words affect our emotions. According to Warriner et al. [23], there are three components of each emotion:
  - valence, i.e. "pleasantness"([23]), ranges from "happy" to "unhappy"
  - arousal is "the intensity of emotion provoked by a stimulus" ([6])
  - dominance denotes "the degree of control exerted by a stimulus" ([23])

  For each of these emotion components, Warriner et al. provide word ratings. We build three features: average valence of words in the argument, average arousal and average dominance.

Before normalizing all features we filter out odd documents based on rules. To give an example, the average word length in an argument is expected to be between 2 and 16. Odd documents are assigned the rating 0.0. Such documents are typically spam or short meta-posts like e.g. "I accept", "Vote Pro" etc.

As training data we use the Webis-ArgQuality-20 Corpus [13]. It contains about 1600 arguments from the argsme corpus. Furthermore, it provides ratings for all three argument quality dimensions as well as for combined/overall argument quality. These continuous ratings range from -4.0 (not an argument) to 4.0.

We train several several machine learning models: Linear Regression, Decision Tree Regression and Support Vector Regression (SVR) with different kernels. For each type of model we train one instance on rhetorical quality and another instance on combined quality. Both instances' parameters are optimized via grid search.

[5] https://github.com/words/hedges

All models perform rather poorly, confirming that argument quality prediction is a difficult problem. SVR with a quadratic kernel achieves the best results (MSE of 1.641 for rhetorical and 1.475 for combined quality). Moreover, we train an ensemble model (Linear Regression) using the predictions of all models as features. As expected, it outperforms all single models (MSE of 1.468 for rhetorical and 1.322 for combined quality).

An interesting detail is that all models, even those trained on rhetorical quality, perform better in predicting combined argument quality than in predicting rhetorical quality. In other words, overall quality seems to be easier to grasp than rhetorical quality, at least with our approach. This hypothesis is statistically significant for $p < 0.01$. One explanation may be that some of our features also capture aspects of dialectical and logical quality: For example, providing sources to support a claim could indicate logical correctness. Features related to subjectivity and emotionality might at least be able to suggest low dialectical quality, as a very emotional and/or subjective post is often unlikely to contribute to resolving an issue.

Finally, to obtain the desired quality function $q : D \rightarrow [0, 1]$, we let the trained models predict the combined quality of every argument in the argsme corpus. We compute the predictions of the best single model (quadratic SVR trained on combined quality) and the ensemble model, leading to two candidates q_{svr}, q_{ens} for q. Figure 2 shows the distributions of the ratings generated by both models.

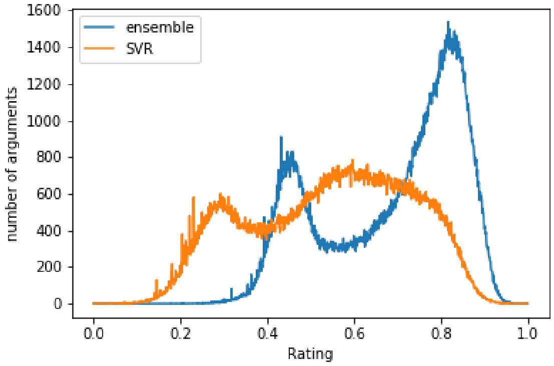

Fig. 2. Distributions of ratings produced by ensemble classifier (q_{ens}) and SVR classifier (q_{svr})

We choose q_{svr} for q, even though the SVR model's MSE is higher than that of the ensemble method. The main reason for this decision is that there are almost no "bad" arguments according to q_{ens}, which is certainly inaccurate.

3.2 Retrieval Model

Like the implementation of args.me we decide to use Apache Lucene for the indexing and retrieval tasks. We index the extended corpus which is generated during preprocessing. In the first step of query processing, stopwords are removed from the query. Before results are retrieved, the query is extended using additional query expansion methods. For the ranking, we implement different methods. As a baseline, we use Lucene's BM25 implementation. Furthermore, we extend the Lucene search core with an implementation of the DPH concept [4]. These retrieval methods do not consider the quality ratings q of our extended corpus. To gain a profit from q we perform a reranking. The scoring function for a document d is given by:

$$score(d) = \alpha \cdot s'(d) + (1 - \alpha) \cdot q(d) \tag{2}$$

where s' is the normalized score of the retrieval model. A reasonable value of α ($\alpha \in [0, 1]$) can be determined empirically. Initially, we set $\alpha = 0.5$.

3.3 Query Expansion

A query is a short representation of the user's information need. However, these few words may not be sufficient to encompass the entire concept that the user wants to express. This can lead to highly relevant documents not being found by the retrieval system due to vocabulary missmatch. That is, the user may choose terms for his query that do not appear in a relevant document. To mitigate this gap, automatic query expansion methods can be used. In this section we briefly describe the components of AQE and our implementation.

For our query expansion component we decided to use one simple baseline approach and two more sophisticated concepts. As baseline, we employ WordNet to fetch semantically similar words for each individual query term. This method can not grasp the concept of the entire query as one unit. However, as many of the queries provided for the shared task only consist of few terms, such as "speed limit" or "nuclear weapons", this simple AQE method can potentially provide a useful enhancement.

The other two expansion procedures both rely on word embeddings. We use fastText[6] to obtain vector representations from the argsme corpus. We combine these locally trained representations with pre-trained embeddings offered by fastText. Then, we adapt a query expansion method as proposed by Diaz et al. [12]. This model-based expansion procedure searches the word embeddings for semantically similar terms in order to estimate an alternative to the original query by interpolating the query language model p_q with that of the expansion language p_{q^+} as follows:

$$p_q^1(w) = \lambda p_q(w) + (1 - \lambda) p_{q^+}(w) \tag{3}$$

[6] https://fasttext.cc/

All newly found terms are then weighted and the best ones (matching the modeled language) are selected to augment the query.

Even though the work presented by Zuccon et al. [26] does not directly focus on AQE, we also use their insights to realize another expansion method. They investigate different ways to estimate translation probabilities for terms that belong to the same language model. Similarly, our goal for AQE is to find words w that are likely to be "translations" of the initial query terms:

$$p_t(w|q) = \Sigma_{u \in q} p_t(w|u) p(u|q) \tag{4}$$

where $p_t(w|u)$ describes the probability of translating term u into w which can be approximated by a normalized cosine similarity.

Naturally, both expansion techniques operate on each query as a whole to incorporate their relatedness. Eventually, the expansion terms and their respective weights are returned to our search core. Note that for the current implementation only one expansion method is used at a time.

3.4 Clustering and Reranking

A more experimental component of our search engine is the clustering/reranking module.

In retrieving arguments, not only the argumentative quality of the returned results is important. Another aspect of an argument search engine's utility is the heterogeneity of the returned arguments. In every use case, the user benefits from receiving a wide variety of semantically different arguments.

A problem of the argsme corpus is that an argumentative document usually contains more than one argument. Nevertheless, documents may often be semantically similar. Moreover, optimizing heterogeneity can conflict with optimizing quality. Both goals need to be balanced.

As the Touchè task is evaluated using nDCG, we first make sure that our results are of high quality (w.r.t to the query and the argumentation quality). Then, the top 8 results are clustered and reranked in order to diversify the top results.

Semantic clustering is implemented using Latent Semantic Analysis (Deerwester et al. [10]) for 3 topics. This provides a vector of size 3 for each of the top 8 documents. Now, the distance $dist(d_1, d_2) = 1 - SIM(d_1, d_2)$, i.e. dissimilarity between two documents d_1 and d_2 can be described in terms of the 3-dimensional vectors generated by LSA.

In the following, let R be the ranking and $R[i]$ the document with rank i in R. Similar to Deselaers et al. [11], we employ a notion of a document's novelty. Novelty of a document $R[i]$ is related to $R[i]'s$ predecessors in the ranking $R[1]...R[i-1]$:

$$Nov(R[i]) := \Sigma_{k=1}^{i-1} \frac{1}{k} dist(R[i], R[i-k]) \tag{5}$$

We weight the dissimilarity depending on the number of ranks between $R[j]$ and $R[j-k]$: documents should not be similar to their immediate predecessor.

Based on novelties, we define a measure for R's diversity/heterogeneity:

$$heterogeneity(R) := \Sigma_{j=2}^{|R|} \frac{1}{j} Nov(R[j]) \tag{6}$$

The likelihood that a user actually looks at a document d decreases with d's rank. Because of that, our heterogeneity measure weights each document's novelty depending on its rank.

Next, heterogenity of a ranking R needs to be balanced with R's quality. To achieve this, we compute a reranking R' of R that maximizes

$$\gamma * quality(R') + (1 - \gamma) * heterogeneity(R') \tag{7}$$

We use nDCG with our retrieval model's ratings to compute $quality(R')$. Note that the problem of finding an optimal R' can be framed as Mixed Integer Program. However, since we restrict ourselves to reranking only the top 8 documents, we find an optimal solution using brute force. Table 1 shows the effect of our reranking on a dummy corpus.

Table 1. Examples for reranking on a dummy subset of the argsme corpus, for different values of γ in Equation 7. In Brackets the dummy quality value for each "argument".

Initial ranking ($\gamma = 1.$)	$\gamma = 0.5$	$\gamma = 0$
Vote Con! (1.0)	Vote Con! (1.0)	Vote PRO. (0.625)
Vote Con (0.875)	Vote for Con. (0.75)	Please extend ... (0.25)
Vote for Con. (0.75)	Extend my arg... (0.375)	vote for pro (0.5)
Vote PRO. (0.625)	Vote Con (0.875)	extend all arg... (0.125)
vote for pro (0.5)	Please extend ... (0.25)	Vote Con (0.875)
Extend my arguments. (0.375)	vote for pro (0.5)	Vote for Con. (0.75)
Please extend all arguments (0.25)	Vote PRO. (0.625)	Extend my arg... (0.375)
extend all arguments (0.125)	extend all arg... (0.125)	Vote Con! (1.0)

The hyperparameter $\gamma \in [0, 1]$ in Equation 7 could be set by the user. Alternatively, γ could be further investigated in order find a reasonable value. This is beyond the scope of our project. For the evaluation, we turn off the clustering component (i.e. set γ to 1), because nDCG does not consider heterogeneity.

4 Evaluation and Results

For the final evaluation of our system, we decide to use the combination of DPH and baseline query expansion. Additionally, we augment the scoring function with our quality ratings as described in Equation 2 using $\alpha = 0.5$. The clustering component is not used, since it can not be expected to have a positive impact on nDCG scores, as pointed out in Section 3.4. Among the various retrieval models, DPH should show the best performance according to the findings of Potthast et al. [17]. Moreover, some quick experiments with manually labelled test data have shown that among our query expansion methods, the baseline expansion achieves the most satisfying results.

The result of the final run which was evaluated via tira.io [18] reaches a sound nDCG@5 of 0.804. For the older version of the corpus, this run was the best performing among all participants[7], indicating that the employed combination could be suitable to perform argument search tasks. We did not submit a run for the more recent corpus version.

5 Conclusion and Outlook

We implemented an argument search engine for the argsme corpus. The results, however, are not very convincing yet. We suppose they could be improved in future work, considering the following aspects. As the search engine proved to benefit from our argument quality ratings, these ratings could be further investigated. More sophisticated features and models could be tested. What is more, the weighting of the ratings in our retrieval model, i.e. the hyperparameter α in Equation 2, could be optimized. One of the major downsides of our approach is that it does not analyse the semantics of potentially relevant documents. Thus, the precision is often rather low. Future work could tackle this issue. A closer investigation of the query expansion component (e.g. investigating more queries) would probably improve our search engine's results, too. We implement a reranking component to diversify the top-ranked results. However, we were not able to evaluate its quality within the scope of this work. What is more, the reranking component is only implemented in a proof-of-concept style, limited to the top 8 documents.

To conclude, we aimed to address the complex problem of argument retrieval using several different methods. There is much space for extending and enhancing our approach in order to improve its performance.

References

1. Ajjour, Y., Wachsmuth, H., Kiesel, J., Potthast, M., Hagen, M., Stein, B.: Data acquisition for argument search: The args. me corpus. In: Joint German/Austrian Conference on Artificial Intelligence (Künstliche Intelligenz). pp. 48–59. Springer (2019)
2. Ajjour, Y., Wachsmuth, H., Kiesel, J., Potthast, M., Hagen, M., Stein, B.: Data Acquisition for Argument Search: The args.me corpus. In: Benzmüller, C., Stuckenschmidt, H. (eds.) 42nd German Conference on Artificial Intelligence (KI 2019). pp. 48–59. Springer (Sep 2019). https://doi.org/10.1007/978-3-030-30179-8_4
3. Amati, G.: Frequentist and bayesian approach to information retrieval. In: European Conference on Information Retrieval. pp. 13–24. Springer (2006)
4. Amati, G.: Frequentist and bayesian approach to information retrieval. In: European Conference on Information Retrieval. pp. 13–24. Springer (2006)

[7] We were automatically assigned the team name *Weiss Schnee*

5. Bondarenko, A., Fröbe, M., Beloucif, M., Gienapp, L., Ajjour, Y., Panchenko, A., Biemann, C., Stein, B., Wachsmuth, H., Potthast, M., Hagen, M.: Overview of Touché 2020: Argument Retrieval. In: Working Notes Papers of the CLEF 2020 Evaluation Labs (Sep 2020)
6. Brysbaert, M., Warriner, A.B., Kuperman, V.: Concreteness ratings for 40 thousand generally known english word lemmas. Behavior research methods **46**(3), 904–911 (2014)
7. Carbonell, J., Goldstein, J.: The use of mmr, diversity-based reranking for reordering documents and producing summaries. In: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval. pp. 335–336 (1998)
8. Carpineto, C., Romano, G.: A survey of automatic query expansion in information retrieval. Acm Computing Surveys (CSUR) **44**(1), 1–50 (2012)
9. Clarke, C.L., Kolla, M., Cormack, G.V., Vechtomova, O., Ashkan, A., Büttcher, S., MacKinnon, I.: Novelty and diversity in information retrieval evaluation. In: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval. pp. 659–666 (2008)
10. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. Journal of the American society for information science **41**(6), 391–407 (1990)
11. Deselaers, T., Gass, T., Dreuw, P., Ney, H.: Jointly optimising relevance and diversity in image retrieval. In: Proceedings of the ACM international conference on image and video retrieval. pp. 1–8 (2009)
12. Diaz, F., Mitra, B., Craswell, N.: Query expansion with locally-trained word embeddings. arXiv preprint arXiv:1605.07891 (2016)
13. Gienapp, L., Stein, B., Hagen, M., Potthast, M.: Efficient Pairwise Annotation of Argument Quality. In: 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020). pp. 5772–5781. Association for Computational Linguistics, Online (Jul 2020), https://www.aclweb.org/anthology/2020.acl-main.511
14. Hutto, C.J., Gilbert, E.: Vader: A parsimonious rule-based model for sentiment analysis of social media text. In: Eighth international AAAI conference on weblogs and social media (2014)
15. Kaptein, R., Koolen, M., Kamps, J.: Result diversity and entity ranking experiments: Anchors, links, text and wikipedia. Tech. rep., AMSTERDAM UNIV (NETHERLANDS) INTELLIGENT SYSTEMS LAB AMSTERDAM (2009)
16. Persing, I., Ng, V.: Why can't you convince me? modeling weaknesses in unpersuasive arguments. In: IJCAI. pp. 4082–4088 (2017)
17. Potthast, M., Gienapp, L., Euchner, F., Heilenkötter, N., Weidmann, N., Wachsmuth, H., Stein, B., Hagen, M.: Argument Search: Assessing Argument Relevance. In: 42nd International ACM Conference on Research and Development in Information Retrieval (SIGIR 2019). ACM (Jul 2019). https://doi.org/10.1145/3331184.3331327, http://doi.acm.org/10.1145/3331184.3331327
18. Potthast, M., Gollub, T., Wiegmann, M., Stein, B.: TIRA Integrated Research Architecture. In: Ferro, N., Peters, C. (eds.) Information Retrieval Evaluation in a Changing World. The Information Retrieval Series, Springer (Sep 2019). https://doi.org/10.1007/978-3-030-22948-1_5
19. Rinott, R., Dankin, L., Alzate Perez, C., Khapra, M.M., Aharoni, E., Slonim, N.: Show me your evidence - an automatic method for context dependent evidence detection. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 440–450. Association for Computational

Linguistics, Lisbon, Portugal (Sep 2015). https://doi.org/10.18653/v1/D15-1050, https://www.aclweb.org/anthology/D15-1050

20. Stab, C., Daxenberger, J., Stahlhut, C., Miller, T., Schiller, B., Tauchmann, C., Eger, S., Gurevych, I.: Argumentext: Searching for arguments in heterogeneous sources. In: Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: demonstrations. pp. 21–25 (2018)
21. Wachsmuth, H., Naderi, N., Hou, Y., Bilu, Y., Prabhakaran, V., Thijm, T.A., Hirst, G., Stein, B.: Computational argumentation quality assessment in natural language. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers. pp. 176–187 (2017)
22. Wachsmuth, H., Potthast, M., Al-Khatib, K., Ajjour, Y., Puschmann, J., Qu, J., Dorsch, J., Morari, V., Bevendorff, J., Stein, B.: Building an argument search engine for the web pp. 49–59 (Sep 2017). https://doi.org/10.18653/v1/W17-5106, https://www.aclweb.org/anthology/W17-5106
23. Warriner, A.B., Kuperman, V., Brysbaert, M.: Norms of valence, arousal, and dominance for 13,915 english lemmas. Behavior research methods **45**(4), 1191–1207 (2013)
24. Wei, Z., Liu, Y., Li, Y.: Is this post persuasive? ranking argumentative comments in online forum. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). pp. 195–200 (2016)
25. Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to ad hoc information retrieval. In: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Association for Computing Machinery, New York, NY, USA (2001). https://doi.org/10.1145/383952.384019, https://doi.org/10.1145/383952.384019
26. Zuccon, G., Koopman, B., Bruza, P., Azzopardi, L.: Integrating and evaluating neural word embeddings in information retrieval. In: Proceedings of the 20th Australasian document computing symposium. pp. 1–8 (2015)