Answering Comparative Questions: Better than Ten-Blue-Links?

Matthias Schildwächter
University of Hamburg, Germany

Alexander Bondarenko
Martin Luther University of Halle-Wittenberg, Germany

Julian Zenker
University of Hamburg, Germany

Matthias Hagen
Martin Luther University of Halle-Wittenberg, Germany

Chris Biemann
University of Hamburg, Germany

Alexander Panchenko
University of Hamburg, Germany

ABSTRACT

We present CAM (comparative argumentative machine), a novel open-domain IR system to argumentatively compare objects with respect to information extracted from the Common Crawl. In a user study, the participants obtained 15% more accurate answers using CAM compared to a "traditional" keyword-based search and were 20% faster in finding the answer to comparative questions.

KEYWORDS

HCI, Comparative Question Answering, Keyword Search, Natural Language Processing

ACM Reference Format:
Matthias Schildwächter, Alexander Bondarenko, Julian Zenker, Matthias Hagen, Chris Biemann, and Alexander Panchenko. 2019. Answering Comparative Questions: Better than Ten-Blue-Links?. In *2019 Conference on Human Information Interaction and Retrieval (CHIIR '19), March 10–14, 2019, Glasgow, United Kingdom.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3295750.3298916

1 INTRODUCTION

Everyone faces choice problems on a daily basis. Besides choosing what to wear or what to have for lunch, people compare all kinds of options: cameras to buy, universities to study at, or even programming languages to use. Question answering platforms like Quora, Reddit, or StackExchange are packed with comparative questions like "How does X compare to Y with respect to Z?". An informed choice then is often based on an objective argumentation why to favor one of the candidates (e.g., comparing important aspects).

Specific product comparison systems, such as Compare.com or Check24, allow to compare any subset of objects in narrow domains such as cameras. Other systems like WolframAlpha aim at providing comparative functionality across domains, but also often only use some (limited) structured database while ignoring the rich textual content available on the web. Somewhat surprising, no system is currently able to satisfy comparative information needs for the general domain with sufficient coverage and explanations. No available system is able to support comparisons on a broad range of object types with arguments about relative qualities or even supporting objective arguments about the best choice. Indeed, web search engines are able to directly answer many factoid questions but do not treat comparative questions any special beyond returning default search results. Advanced question answering systems, such as IBM's Watson [6], answer factoid questions very well, but do not really handle comparative questions of everyday users.

We present CAM (comparative argumentative machine), a system that aims at solving the shortcomings mentioned above. CAM is a tool for answering comparative general-domain questions based on information extracted from the web-scale Common Crawl.[1]

2 RELATED WORK

Commercial systems like GoCompare, Compare.com, Diffen.com, and Versus.com offer high-precision comparison capabilities based on well-curated structured data sources focusing on single domains. But their low coverage in other than their focus domains rules out answering most of the comparative questions found on portals like Quora or Yahoo! Answers—that themselves form a good source of (argumentative) comparisons and results.

Previous text-based comparison approaches have mostly focused on the biomedical domain. Fiszman et al. [7] collected sentences comparing drug therapies using manually crafted patterns to recognize the subjects of comparison and the comparison direction. They reached a very high precision at moderate recall. On a set of full-text articles on toxicology, Park and Blake [13] succeeded in training a highly precise Bayesian Network for identifying comparative sentences relying on lexical clues and dependency parsers. More recently, Gupta et al. [9] described a system based on manually collected patterns on the basis of lexical matches and dependency parses in order to identify comparison targets and to classify the type of comparison into the classes given by Jindal and Liu [11]: gradable vs. non-gradable and superlative comparisons.

Building a general-domain argumentative comparison facility comes with the additional challenge of argument mining from user-generated content [15]. Text is typically noisy, misses argument structures and contains poorly formulated claims. On the other hand, specialized jargon and idiosyncrasies of a platform can be utilized [4] (e.g., hashtags for mining argumentative tweets).

Aker et al. [1] confirmed the findings of Stab and Gurevych [17] that information from dependency parsers does not help to find the (general) argument structure in persuasive essays and Wikipedia articles while simpler structural features such as punctuation are more effective. Daxenberger et al. [3] noted that claims across different domains share lexical clues and further stated that current

[1]**Demo, API & code: http://ltdemos.informatik.uni-hamburg.de/cam/**

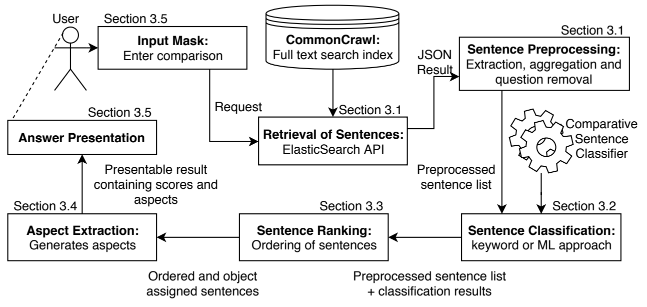


Figure 1: Design of the CAM system.

datasets are too small for recent DNN-based classifiers resorting to traditional feature engineering for argument mining.

Some argument mining systems work on larger corpora of user-generated content to find the most relevant argument for a given claim [10] or to oppose different argumentative viewpoints [19]. Web-scale systems for comparing query results [18] or for retrieving single arguments matching a user query [16] form the inspiration for our new CAM system (comparative argumentative machine).

3 THE CAM SYSTEM DESIGN

To ensure a wide coverage, a comparative answer of our CAM system for two objects is based on argumentative structures extracted from web-scale text resources. The system looks for textual structures asserting that one of the compared objects is superior to the other, that they are equal, or that they are not comparable. A comparison of two objects o and o' in the CAM sense is defined as "o **?** o' w.r.t. $a_i, \ldots, a_j \in \mathbf{a}$", where **?** is in $\{>, <, =, \neq\}$ and $\mathbf{a} = \{a_1, \ldots, a_k\}$ is the set of comparison aspects of o and o'. We thus focus on mining sentences like "*Python=o* is better than *Matlab=o′* for *web development=a_i*."

The design of our CAM system is shown in Figure 1. It consists of the following generic stages, which are further described in details: (1) retrieval of relevant sentences, (2) classification of comparative sentences, (3) ranking of the comparative sentences, (4) extraction of object aspects, and (5) presentation of the answer.

3.1 Sentence Retrieval

Our CAM system uses an Elasticsearch full text index of a linguistically pre-processed corpus [12] containing 14.3 billion English sentences from the Common Crawl. To retrieve textual argumentative structures relevant to a comparative user input, the index is queried for sentences matching the input objects and containing comparison aspects; sentences without aspects are used as a fallback. Questions are removed from the initial retrieval results since they usually do not help in returning an argumentative answer.

3.2 Sentence Classification

We use a classifier to distinguish between four classes: the first object from the user input is better / equal / worse than the second one ($>$, $=$, $<$) w.r.t. a comparison aspect, or no comparison is found ($\neq$). The classifier uses the text between both objects to identify the "polarity" and is inspired by the best model reported by Franzek et al. [8]: XGBoost [2] using word frequencies as representations, which achieves a high F1 score of 0.92 for $\neq$, a good F1 of 0.74 for $>$ but a rather low F1 of 0.46 for $<$. We identified the main issue in missing negation handling, for which we added a simple keyword-based heuristic to our CAM system inverting common negations.

3.3 Sentence Ranking and Object Comparison

To rank comparative sentences (category $>$ or $<$), we score them by combining the classifier confidence and the Elasticsearch score[2] according to the following heuristic s:

$$s = \begin{cases} \alpha + e + e_{max}, & \text{if confidence} \geq \gamma, \\ (\alpha + e) \cdot \delta, & \text{otherwise,} \end{cases}$$

where e is the Elasticsearch score of the sentence, e_{max} is the maximum Elasticsearch score of any comparative sentence retrieved for the user input, and $\alpha = w_{a_i} e_{max}$ if the user-specified aspect a_i is present in the sentence and $\alpha = 0$ otherwise. For the α aspect boost, the weights w_{a_i} are specifiable in the user interface. Confidently classified sentences obtain a boost of e_{max} while scores of low confidence sentences are decreased by a factor of δ; we set $\gamma = 0.8$ and $\delta = 0.1$ in our experiments.

For scoring a CAM output "$o > o'$ w.r.t. $\mathbf{a}$", we sum up the s-scores of all sentences *supporting* the statement. To this end, we have developed a heuristic to include two directions of comparison and thus taking into account that a statement like "Python is better than Matlab" (class $>$) is also supported by "Matlab is worse than Python" (class $<$); important factors being the object ordering and the polarity.

3.4 Aspect Extraction

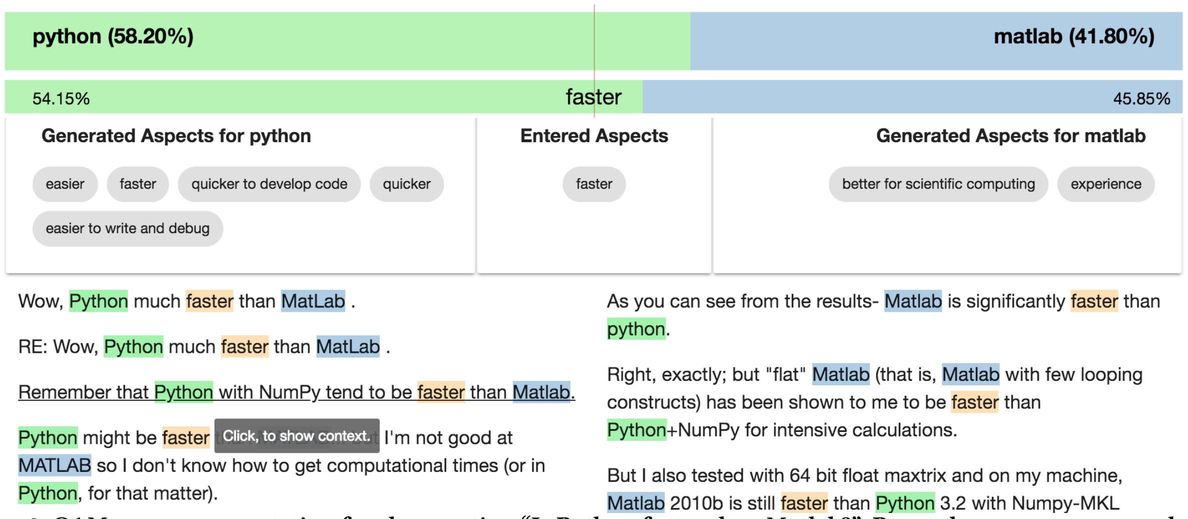
In addition to user-specified comparison aspects, CAM generates up to ten supplementary aspects (even when no comparison aspect at all was provided by the user). We use three different methods for aspect mining: (1) searching for comparative adjectives and adverbs; (2) searching for phrases with comparative adjectives / adverbs and a preposition like *to*, *for*, etc. (e.g., "*quicker to* develop code" or "*better for* scientific computing"); (3) searching for specific hand-crafted patterns like "because of higher *speed*", "since it has more *options*", "as we have proven its *resilience*" or "reason for this is the *price*". An extracted aspect is assigned to the object with the higher co-occurrence frequency (cf. Figure 2 for examples).

3.5 CAM User Interface

The user interface consists of a question input form (Figure 3) and an answer presentation component (Figure 2). The input interface allows to submit a comparative question in the form of two compared objects and their aspects. The answer presentation summarizes the sentences retrieved from the Common Crawl providing decision-making support for the informed choice.

Input Form. The input form is divided into three parts (cf. Figure 3). On the top, the user enters two comparison target objects. In the middle, the interface allows to add an arbitrary number of aspects and assign them a weight indicating their importance (1 to 5; used to boost the scores of the sentences containing the aspect). On the bottom, one of the three different search models can be selected: *Default* is based on keyphrases like "better than" or "faster

[2]https://www.elastic.co/guide/en/elasticsearch/guide/current/scoring-theory.html


python (58.20%)

matlab (41.80%)

54.15% faster 45.85%

Generated Aspects for python

easier | faster | quicker to develop code | quicker | easier to write and debug

Entered Aspects

faster

Generated Aspects for matlab

better for scientific computing | experience

Wow, Python much faster than MatLab .

RE: Wow, Python much faster than MatLab .

Remember that Python with NumPy tend to be faster than Matlab.

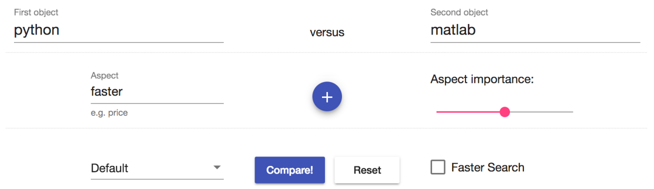
Python might be faster Click, to show context. I'm not good at MATLAB so I don't know how to get computational times (or in Python, for that matter).

As you can see from the results- Matlab is significantly faster than python.

Right, exactly; but "flat" Matlab (that is, Matlab with few looping constructs) has been shown to me to be faster than Python+NumPy for intensive calculations.

But I also tested with 64 bit float maxtrix and on my machine, Matlab 2010b is still faster than Python 3.2 with Numpy-MKL

Figure 2: CAM answer presentation for the question "Is Python faster than Matlab?". Pro and con sentences are shown.


First object: python versus Second object: matlab

Aspect: faster (e.g. price) + Aspect importance:

Default | Compare! | Reset | Faster Search

Figure 3: CAM input form.

than" to find comparative sentences; *BoW* is built upon the word frequency-based XGBoost classifier described above, and *Infersent* uses sentence embeddings. The *Faster Search* option limits the number of queried fall-back sentences to 500 in order to speed up the answer construction.

Answer Presentation. On the top of the comparative answer presentation (cf. Figure 2), different score bars are given. The overall score distribution allows the user to grasp a general impression for the entered comparative question while the aspect-specific score bars show the distribution for the individual user-specified aspects.

Additionally, up to ten automatically generated aspects are presented in a clickable manner to allow the user to only display result sentences for such aspects (disjunctive filter interpretation). The user-specified aspects are used on both result sentence sides while the generated aspects only filter the corresponding column.

The objects in the displayed sentences are highlighted with their respective colors from the score bars, while the aspect highlighting uses different colors. Clicking on a result sentence reveals its Common Crawl context—by default the ±3 sentences around it, with the possibility of expanding to the whole original document.

4 EVALUATION

We compare our new CAM system to a keyword-based search in two user studies with 14 and 9 participants on 34 comparison topics.

4.1 Experimental Setup

The 34 topics (two compared objects + one aspect) for our studies were created from comparative Quora questions containing the phrase "better than" and also being present on comparison pages like Diffen.com and DifferenceBetween.net. For each topic, we manually double-checked that the underlying corpus of our CAM system and the keyword-based search (i.e., the 14.3 billion Common Crawl sentences) allows to answer the comparison; we only included topics with at least 20 support sentences. One of the topics for instance has *mp3* and *wma* as objects and *compression ratio* as the aspect. Given the ground truth answer *worse*, a study participant should answer that mp3 is worse than wma with respect to compression ratio. To clarify potential ambiguities or subjectivities, descriptions for the participants were added to the topics (e.g., to inform a potential music afficionado who might claim a worse compression rate being better since it might come with an improved sound quality).

In a **Group A** setup, we focus on the question whether users are faster answering correctly when using the CAM system. A G*Power analysis [5] did output a required sample size of 272 comparisons to be a able to measure a statistically significant difference in answer times. We thus decided to engage 14 participants on all 34 topics (477 comparisons). Each participant uses both experimental systems alternatingly (CAM / keyword-based search). Since every participant should work on each topic just once with one of the systems but not the other, we randomly split the 34 topics into two groups (one for each system). To avoid any order bias, the topics of each group were presented in random order.

The participants were informed that they should give an answer as quickly as possible; the whole study took about one hour per participant and ended with a questionnaire. We measured the time for different phases (e.g., the time needed to enter a query and the time needed to determine an answer) and the correctness of a participant's answer with respect to manually derived gold labels.

In a **Group B** setup, we focussed on collecting some more "natural" feedback using a less forced study environment. We had 9 different participants (not from Group A) who were allowed to just "play" with the systems for any and as many of our 34 comparison topics as they liked. In total, 85 comparisons were performed.

4.2 Study Participants

Among the 14 Group A participants, 9 were male (5 female), 13 indicated 18–24 as their age (1 was in the 25–34 range), 8 participants had an Engineering & Computer Science background (3 from Arts, Culture & Entertainment, 1 from Law & Public Policy, 2 selected "other") with 9 having a Bachelor's degree. The participants characterized themselves as having a proficient (nine) or intermediate (five) English level. Seven participants stated to use comparison websites rarely or never (once a year or less), whereas five used them once a month and two even once a week.

Group B consisted of five female and four male participants, 1 participant was 13–17 years old, 2 participants fell in the 18–24 age range, 5 in the 25–34 range, and one in 35–44. This group was dominated by an Engineering & Computer Science background (five out of nine); one from Education, one from Business, one from Arts, Culture & Entertainment, and one "other" background. Four participants already had a Master's degree, two were students, one had a Bachelor's degree, one a doctorate degree, and one selection of "other". Six participants rated their English level as proficient and three as intermediate. Five participants stated they used comparison websites rarely or never, whereas two used them once a month and two even once a week or more.

4.3 Results and Discussion

A Shapiro-Wilk test [14] verified the visual assumption of a log-normal distribution ($\alpha = 0.05$) of the different measured times for CAM usage and keyword-based search. Therefore, t-tests were used to check whether the null hypothesis of same answer determination times or same total times can be rejected.

Figure 4 shows the time distributions of Group A. **Until typing** indicates the participants being about 19% faster starting to enter a query with the CAM system (in Group B, the CAM users were even about 25% faster). **Typing** is the time from the first key stroke until the query is submitted. The Group A participants again were faster with the CAM interface (about 24% on average); the Group B participants needed about twice as long, being slightly faster with the CAM system. The **loading** phase measures the time the system needs to show the answer (from sending the query until the result is presented). On average, keyword-based search loads slightly faster than CAM since CAM uses a keyword-based search subroutine with some further post-processing.

Most importantly, the time the users need to give their answer (**determination** in Figure 4) shows that the Group A participants were significantly faster when using the CAM system (about 39% difference). In Group B, the participants were slower in general, but interestingly they were also slightly slower using CAM than keyword-based search. One potential reason is that the participants explored the new CAM interface more even providing verbal feedback during their work (remember that Group B was allowed to "play" with the systems).

For the overall task (**total** in Figure 4), Group A was significantly faster when using the CAM interface while the more exploratory Group B was overall slower but with no substantial advantage for either system. Our main focus in a Group B was on observing participants behavior which is why they were allowed to test, play with and comment on the systems while using them.

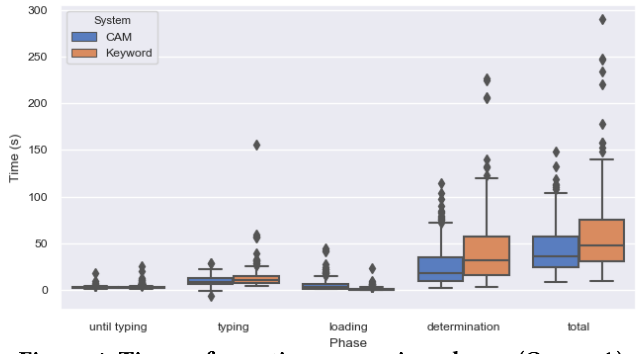


Figure 4: Times of question answering phases (Group A).

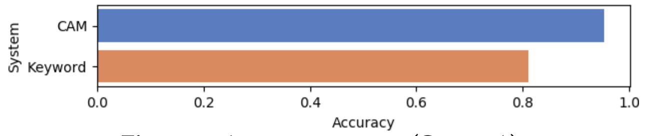


Figure 5: Answer accuracy (Group A).

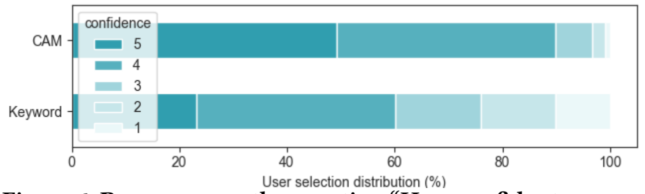


Figure 6: Responses on the question "How confident are you that the determined answer is correct?" (Group A).

Besides statistically significant quicker answers, the Group A participants also made fewer errors using the CAM system (cf. Figure 5). The average CAM accuracy in Group A is 95% (9 of 14 participants reached 100%), whereas for the keyword-search it is 81% (with a best result of 94%). The Group B participants also were more accurate using CAM (84%) than using keyword-based search (75%).

In the evaluation questionnaire, we asked the participants to rate the system features on the scale from 1 (very negative) to 5 (very positive). The question "How convenient was it to use the CAM system?" and the statement "Learning the usage of CAM is..." achieved values between 4 and 5 for both groups, which is very positive. In addition, the participants of both groups on average were almost one point more confident that an answer determined by CAM was correct than for keyword-based search (cf. Figure 6 for Group A; 5 being the highest confidence).

5 CONCLUSION

Our new CAM system helps users to faster and more confidently find answers on comparative questions compared to a keyword-based search. Moreover, a summary provided in the answer serves to support a decision-making process. While the objects of comparison and the important aspects have still to be stated explicitly, this gives rise to comparative question handling in search engines once respective questions can be identified automatically. A demo of our CAM system is online[3] and available as open source.[4]

[3] http://ltdemos.informatik.uni-hamburg.de/cam/
[4] https://github.com/uhh-lt/cam

ACKNOWLEDGMENTS

This work has been supported by the Deutsche Forschungsgemeinschaft (DFG) within the project "Argumentation in Comparative Question Answering (ACQuA)" (grant BI 1544/7-1 and HA 5851/2-1) that is part of the Priority Program "Robust Argumentation Machines (RATIO)" (SPP-1999).

REFERENCES

[1] Ahmet Aker, Alfred Sliwa, Yuan Ma, Ruishen Lui, Niravkumar Borad, Seyedeh Ziyaei, and Mina Ghobadi. 2017. What works and what does not: Classifier and feature analysis for argument mining. In *Proceedings of ArgMining@EMNLP 2017*. 91–96.

[2] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A scalable tree boosting system. In *Proceedings of KDD 2016*. 785–794.

[3] Johannes Daxenberger, Steffen Eger, Ivan Habernal, Christian Stab, and Iryna Gurevych. 2017. What is the essence of a claim? Cross-domain claim identification. In *Proceedings of EMNLP 2017*. 2055–2066.

[4] Mihai Dusmanu, Elena Cabrio, and Serena Villata. 2017. Argument mining on Twitter: Arguments, facts and sources. In *Proceedings of EMNLP 2017*. 2317–2322.

[5] Franz Faul, Edgar Erdfelder, Albert-Georg Lang, and Axel Buchner. 2007. G* Power 3: A flexible statistical power analysis program for the social, behavioral, and biomedical sciences. *Behavior Research Methods* 39, 2 (2007), 175–191.

[6] David A. Ferrucci, Eric W. Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John M. Prager, Nico Schlaefer, and Christopher A. Welty. 2010. Building Watson: An overview of the DeepQA project. *AI Magazine* 31, 3 (2010), 59–79.

[7] Marcelo Fiszman, Dina Demner-Fushman, François-Michel Lang, Philip Goetz, and Thomas C. Rindflesch. 2007. Interpreting comparative constructions in biomedical text. In *Proceedings of BioNLP@ACL 2007*. 137–144.

[8] Mirco Franzek, Alexander Panchenko, and Chris Biemann. 2018. Categorization of comparative sentences for argument mining. *CoRR* abs/1809.06152 (2018). http://arxiv.org/abs/1809.06152

[9] Samir Gupta, A. S. M. Ashique Mahmood, Karen Ross, Cathy H. Wu, and K. Vijay-Shanker. 2017. Identifying comparative structures in biomedical text. In *Proceedings of BioNLP@ACL 2017*. 206–215.

[10] Xinyu Hua and Lu Wang. 2017. Understanding and detecting supporting arguments of diverse types. In *Proceedings of ACL 2017 (Volume 2: Short Papers)*. 203–208.

[11] Nitin Jindal and Bing Liu. 2006. Mining comparative sentences and relations. In *Proceedings of AAAI 2006*. 1331–1336.

[12] Alexander Panchenko, Eugen Ruppert, Stefano Faralli, Simone Paolo Ponzetto, and Chris Biemann. 2018. Building a web-scale dependency-parsed corpus from CommonCrawl. In *Proceedings of LREC 2018*.

[13] Dae Hoon Park and Catherine Blake. 2012. Identifying comparative claim sentences in full-text scientific articles. In *Proceedings of DSSD@ACL 2012*. 1–9.

[14] Samuel Sanford Shapiro and Martin B. Wilk. 1965. An analysis of variance test for normality (complete samples). *Biometrika* 52, 3/4 (1965), 591–611.

[15] Jan Snajder. 2017. Social media argumentation mining: The quest for deliberateness in raucousness. *CoRR* abs/1701.00168 (2017). http://arxiv.org/abs/1701.00168

[16] Christian Stab, Johannes Daxenberger, Chris Stahlhut, Tristan Miller, Benjamin Schiller, Christopher Tauchmann, Steffen Eger, and Iryna Gurevych. 2018. ArgumenText: Searching for arguments in heterogeneous sources. In *Proceedings of NAACL 2018 (Demonstrations)*. 21–25.

[17] Christian Stab and Iryna Gurevych. 2014. Identifying argumentative discourse structures in persuasive essays. In *Proceedings of EMNLP 2014*. 46–56.

[18] Jian-Tao Sun, Xuanhui Wang, Dou Shen, Hua-Jun Zeng, and Zheng Chen. 2006. CWS: A comparative web search system. In *Proceedings of WWW 2006*. 467–476.

[19] Henning Wachsmuth, Martin Potthast, Khalid Al Khatib, Yamen Ajjour, Jana Puschmann, Jiani Qu, Jonas Dorsch, Viorel Morari, Janek Bevendorff, and Benno Stein. 2017. Building an argument search engine for the web. In *Proceedings of ArgMining@EMNLP 2017*. 49–59.