Which is Better for Deep Learning: Python or MATLAB? Answering Comparative Questions in Natural Language

Viktoriia Chekalina[1,2], **Alexander Bondarenko**[3], **Chris Biemann**[4], **Meriem Beloucif**[4], **Varvara Logacheva**[2], **and Alexander Panchenko**[2]

[1]Philips Research Russia
[2]Skolkovo Institute of Science and Technology, Russia
[3]Martin-Luther-Universität Halle-Wittenberg, Germany
[4]Universität Hamburg, Germany
{viktoriia.chekalina, v.logacheva, a.panchenko}@skoltech.ru
alexander.bondarenko@informatik.uni-halle.de
{biemann, beloucif}@informatik.uni-hamburg.de

Abstract

We present a system for answering comparative questions (*Is X better than Y with respect to Z?*) in natural language. Answering such questions is important for assisting humans in making informed decisions. The key component of our system is a natural language interface for comparative QA that can be used in personal assistants, chatbots, and similar NLP devices. Comparative QA is a challenging NLP task, since it requires collecting support evidence from many different sources, and direct comparisons of rare objects may be not available even on the entire Web. We take the first step towards a solution for such a task offering a testbed for comparative QA in natural language by probing several methods, making the three best ones available as an online demo.

1 Introduction

Comparison of objects of a particular class (e.g., holiday destinations, mobile phones, programming languages) is an essential daily task that many individuals require every day. According to Bondarenko et al. (2020a), comparative questions constitute around 3% of queries submitted to major search engines—a non-negligible amount. Answering a comparative question (*What is better, X or Y?*) requires collecting and combining facts and opinions about compared objects from various sources. This challenges general-purpose question answering (QA) systems that rely on finding a direct answer in some existing datasets or extracting from web documents.

Nowadays, many websites (e.g. Diffen, WolphramAlpha, or Versus) provide users with a comparison functionality. Furthermore, the task of answering comparative questions has recently attracted the attention of the research community (Kessler and Kuhn, 2014; Arora et al., 2017; Yang et al., 2018). Most of the current research suggests that an answer to a comparative question not only should indicate the "winner" of comparison but also provide arguments in favor of this decision and arguments that support the alternative choice.

Therefore, we argue that a comparative QA system should be a combination of an argument mining engine and a dialogue system that mimics a human expert in the field. In this work, we make the first step towards the development of such technology. Namely, we develop a Comparative Question Answering System (CoQAS), an application that consists of a Natural Language Understanding (NLU) module that identifies comparative structures (objects, aspects, predicates) in free input questions and a Natural Language Generation (NLG) module that constructs an answer. We tested various options for both NLU and NLG parts ranging from a simple template-based generation to Transformers-based language models.

The main contributions of our work are threefold: (i) we design an evaluation framework for comparative QA, featuring a dataset based on Yahoo! Answers; (ii) we test several strategies for identification of comparative structures and for answer generation; (iii) we develop an online demo using three answer generation approaches. A demo of the system is available online.[1] Besides, we release our code and data.

2 Related Work

Text Generation Most of the current text natural language generation tasks (Dušek and Jurčíček, 2016; Freitag and Roy, 2018) are based on se-

[1]https://skoltech-nlp.github.io/coqas

quence to sequence model architecture (Sutskever et al., 2014). The existing generation methods are developed by employing attention mechanism (Bahdanau et al., 2015) and pointer-generator network (See et al., 2017). More recent work on text generation focus on generating natural language using multitask learning from multi-document or multi-passage sources (Hsu et al., 2018; Nishida et al., 2019). However, in our generation task, we have a list of arguments used to build the final answer. This makes our task similar to unsupervised summarization. There exist several approaches for tackling the latter task, e.g. graph-based (Litvak and Last, 2008) and neural models (Isonuma et al., 2019; Coavoux et al., 2019). A common approach to summarization is based on the TextRank graph algorithm (Mihalcea, 2004; Fan and Fang, 2017).

Comparative QA According to Li and Roth (2006), questions can be divided into 6 coarse and 50 fine-grained categories, such as factoid questions, list questions, or definition questions: we focus on comparative questions. Sun et al. (2006) proposed one of the first works on automatic comparative web search, where each object was submitted as a separate query, then obtained results were compared. Opinion mining of comparative sentences is discussed by Ganapathibhotla and Liu (2008) and Jindal and Liu (2006), yet with no connection to argumentation mining. Instead, comparative information needs are partially satisfied by several kinds of industrial systems mentioned above. Schildwächter et al. (2019) proposed Comparative Argumentative Machine (CAM)[2], which a comparison system based on extracting and ranking arguments from the web. The authors have conducted a user study on 34 comparison topics, showing that the system is faster and more confident at finding arguments when answering comparative questions in contrast to a keyword-based search. Wachsmuth et al. (2017) presented args.me, a search engine for retrieving pro and con arguments given for a given controversial topic. The input to this system is not structured but rather a query in a free textual form. The Touché shared task on argument retrieval at CLEF (Bondarenko et al., 2020b, 2021) featured a related track. The task was to retrieve from a large web corpus documents answering comparative question queries like “What IDE is better for Java: NetBeans or Eclipse?”.

[2]https://ltdemos.informatik.uni-hamburg.de/cam

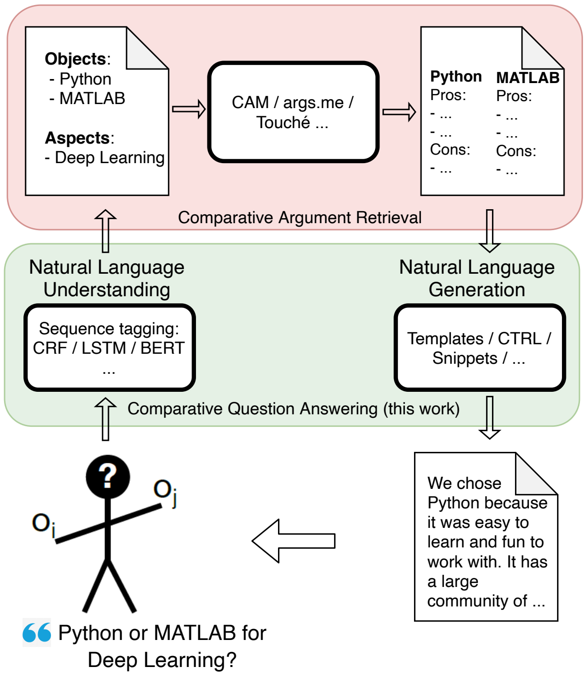

Figure 1: The comparative QA workflow. A user submits a comparative question, the NLU module identifies compared objects and aspects and transfers them to CAM to retrieves comparative arguments. Then, the NLG module represents the arguments in textual form.

3 System Design

Our system is designed to help the user make a proper choice by fully and reasonably describing the possible advantages and disadvantages of each of the matching options. For this purpose, we have defined structures that contain significant information about the desired comparison: compared *objects*, comparison *aspects*, and *predicates*.

In the example “Which is better for Deep Learning: Python or MATLAB?”, the objects are entities that the user wants to compare (*Python, MATLAB*). The predicate is the entity that frames the comparison (*better*); it introduces a comparison relation between the objects and is often represented by a comparative adjective or adverb. Finally, the comparison aspects are shared properties along which the two objects are compared, e.g., *Deep Learning*.

Our comparative question answering system is based on CAM (Schildwächter et al., 2019), which retrieves pro/con arguments for a pair of compared objects. We extend CAM by enabling it to process natural language questions and generate coherent human-like answers as depicted in Figure 1.

Comparative Argument Mining CAM mines sentences in favor or against two compared objects

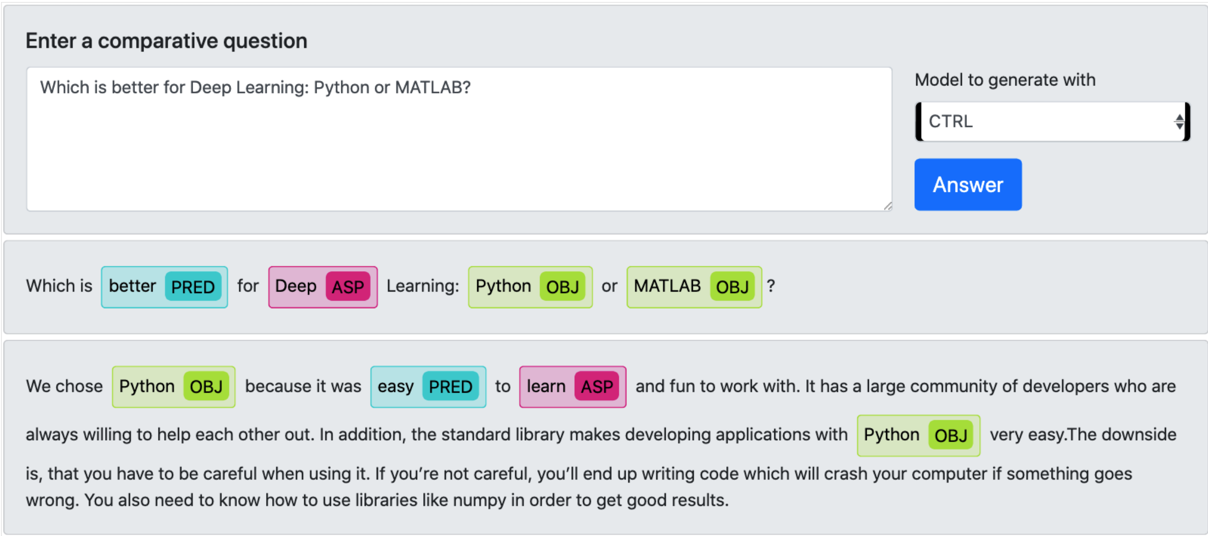

Figure 2: The interface of the Comparative Question Answering System (CoQAS).

with respect to an aspect specified by the user. First, using the Elasticsearch BM25, CAM retrieves sentences containing the two compared objects and the comparison aspect from the Common Crawl-based corpus featuring 14.3 billion sentences (Panchenko et al., 2018). Then, CAM classifies the sentences as comparative or not and identifies the "winner" of the two compared objects in the sentence context. Besides, it extracts aspects and predicates from the retrieved comparative sentences (Panchenko et al., 2019). Finally, CAM outputs a list of argumentative pro/con sentences and shows the "winner" of the comparison along with the comparison aspects.

Comparative Question Answering We extend CAM with natural language question understanding (described in Section 4) and natural language answer generation (described in Section 5) modules. The first module is developed to automatically identify the compared objects and the comparison aspect in a user-provided natural-language comparative question. This information is passed to CAM, which queries DepCC for comparative sentences. The NLG module receives the output of CAM and transforms the retrieved argumentative sentences into a short text, the generated answer. The structure of our modular system is presented in Figure 1.

The user interface (Figure 2) contains an input form for submitting a comparative question and an output box for a generated answer. To improve the readability of the answer and help find the arguments in it, NLU module also labels the output with identified objects, aspects, and predicates. In Figure 2, we present an example of the system's web interface in action.

In the NLG module, we use several approaches to response generation: an information retrieval-based approach and an approach built upon pre-trained language models. These techniques provide different answers: the first is more structured, and the second one is based on experience and opinions. Therefore, we allow a user to choose a generation model from different types: CAM, CTRL, and Snippets (cf. Figure 2).

Finally, for integration into NLP applications, e.g., personal assistants and chatbots, we also provide a RESTful API for our comparative QA.

4 Natural Language Understanding

The goal of the NLU module is to identify the objects to compare and comparison structure aspects and predicates if they were specified. We cast this as a sequence labeling task.

Training Dataset To train the NLU, we created *Comparely*, a dataset with comparative sentences manually labeled with objects, aspects, and predicates. First, we extracted comparative sentences for 270 object pairs from the dataset of (not) comparative sentences by Panchenko et al. (2019). We extracted them from DepCC corpus (Panchenko et al., 2018) using CAM. We then performed manual labeling (two annotators) using WebAnno (Yimam et al., 2013). Some of the extracted sentences were not comparative, so the annotators were instructed to discard them. The majority of sentences were labeled once, but we also labeled 200 of them multiple times to compute the inter-annotator agree-

Table 1: Statistics of the NLU dataset.

	Object	Aspect	Predicate
# occurrences	7,555	2,593	3,990
# per sentence	2.51	1.35	1.34
Avg. # words	1.04	1.37	1.16

ment. The Cohen's κ for the aspect labeling is 0.71 (substantial agreement). For predicates and objects, the values are 0.90 and 0.93, respectively—perfect agreement. The dataset consists of 3,004 sentences, each of them has a comparison of two or more distinct objects and at least one aspect or predicate. The average length of sentence is 26.7 words (Table 1). The majority of sentences compare more than one pair of objects across multiple parameters (i.e., sentences often contain more than one aspect or predicate). As the NLU processed not statements but questions, for the further improvement of the dataset, we could use comparative questions from (Bondarenko et al., 2020a).

This dataset is essentially similar to the ones by (Arora et al., 2017; Kessler and Kuhn, 2014). They also contain comparative statements labeled with objects, aspects, and predicates. The primary difference of our dataset is domain diversity. The mentioned datasets are drawn from a single domain, namely, camera reviews. The information contained in such sentences is difficult to generalize. Thus, they demonstrate a proof of concept rather than a resource that can be used for real-world tasks. On the other hand, *Comparely* features objects of different domains. It was created based on real-world objects that are frequently compared. It contains data from three domains: brands, generic objects, and computer science. The two former domains are more numerous: 41% and 46% sentences deal with objects of brands and generic domains, respectively. The remaining 13% are devoted to objects of the computer science domain.

Method Identification of comparative question components (objects, aspects, predicates, or none) is a sequence-labeling task, where the classifier should tag respective tokens in an input question. We test several common baselines starting with simple one-layer bidirectional LSTM described by (Arora et al., 2017) where the input is encoded with GloVe (Pennington et al., 2014) embeddings. For some further improvements, we add Conditional Random Field (Sutton and McCallum, 2012)

Table 2: Evaluation in terms of F1 of the NLU tagger.

Model	Objects	Aspects	Predicates
RoBERTa	**0.925**	**0.685**	**0.894**
BERT	0.829	0.563	0.869
ELMO	0.654	0.487	0.825
BiLSTM-CRF	0.631	0.475	0.766
BiLSTM	0.582	0.328	0.730

to LSTM and use context-based ELMO (Peters et al., 2018) embeddings for token representations. We also experiment with Transformers (Vaswani et al., 2017) using a pre-trained BERT model (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), which is its modification yielding better performance. For every classifier, during training, we tune hyperparameters by varying a batch size (from 16 to 100) and a learning rate (from 10^{-6} to 10^{-2}). To find a proper converge of the training process, we apply two types of learning rate schedulers: Linear With Warmup and Slanted Triangular.

For the model with the highest achieved F1 (RoBERTa), we employ stochastic weight ensembling (Goodfellow et al., 2015; Garipov et al., 2018), i.e., we interpolate between the weights obtained by training a certain model with different random seeds. All models were trained on the *Comparely* dataset and tested on its manually relabeled subset of 400 sentences. The overview of the classifiers' effectiveness is shown in Table 2.

Results and Discussion The evaluation shows that comparison aspect classification is the hardest task: the baseline one-layer BiLSTM achieves an F1 of 0.33, and the most effective RoBERTa-based classifier achieves an F1 of 0.69. The most reliable classification was achieved for predicting the compared objects with an F1 of 0.58 for the baseline and an F1 of 0.93 for RoBERTa. An addition of a CRF layer and the use of pre-trained ELMo embeddings to the BiLSTM classifier slightly improved the results. Transformers demonstrated significant improvement in classification effectiveness over the baseline. Finally, we choose to deploy a RoBERTa-based classifier in the NLU module of our system.

5 Comparative Answer Generation

Based on comparative sentences retrieved by CAM, we develop several generation approaches to construct a human-like concise answer: (1) genera-

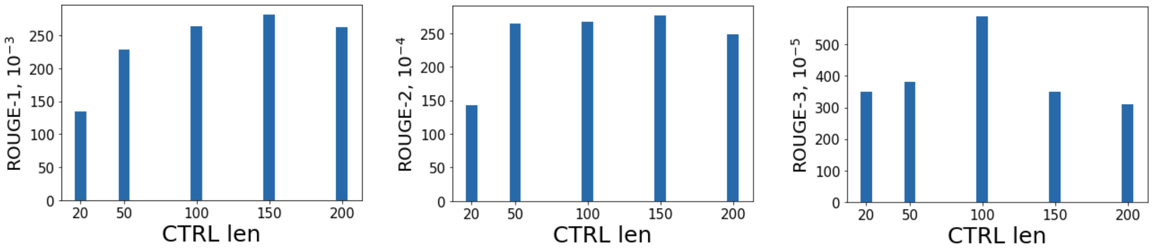

Figure 3: Dependence of ROUGE metrics on the maximum length of the generated sequence (CTRL model).

tion with pre-trained Transformers-based language models, (2) retrieval of argumentative sentences ranked by CAM or TextRank, (3) extracting context of sentence retrieved by CAM as support for the "winning" object, and (4) entering extracted comparative structures in templates.

5.1 Generation Methods

Pre-trained Language Models Pre-trained language models have been shown to contain commonsense knowledge, so they can be successfully used for question answering (Andrews and Witteveen, 2019) and for generating sensible and coherent continuation of text. Therefore, we use Transformers-based CTRL (Keskar et al., 2019) models for answering comparative questions.

CTRL allows explicit control codes to vary the domain and the content of the text. We use the Links control code, which forces the model to produce text similar to online news and reports. We feed into CTRL phrase "Links Which is better in respect to the aspect: $object_1$ or $object_2$?" and a row question from the input.

We also vary the maximum number of tokens generated by CTRL. We experiment with different length set, including: 20, 50, 100, 150, and 200 and generate answers to questions from the Yahoo! Answers dataset (cf. Section 5.2). For the evaluation part, we calculate ROUGE-1, ROUGE-2, ROUGE-3 scores between generated texts and corresponding Yahoo!'s "best answers". According to the results (cf. Figure 3), a model with a maximum length of 100 tokens gives the highest ROUGE-3 score (we select this length parameter for our further experiments).

Sentence-Retrieval-Based Methods The CAM output contains a list of the argumentative sentences ranked by the BM25 inverted index-based score. Every sentence is a supportive argument for the superiority of the respective compared object. Sentence-retrieval-based methods try to extract the most representative sentences and display it in the proper form. To create an answer, CAM: Bullet points mentions a "winner" defined by CAM with respect to aspects if they exist. It also takes the top-3 sentences supporting each of the objects and produces a list for highlighting the advantages and disadvantages of each object in comparison.

An alternative way of retrieving the most relevant sentences is clustering. This approach is used in TextRank: Bullet points. TextRank is a graph-based summarization algorithm. We use the version proposed by Mallick et al. (2019). We represent sentences with hidden states of a LSTM network pre-trained on Wikipedia. TextRank iteratively updates the weights of edges and sets the node weights to be proportional to the importance of adjacent edges. To make the graph sparse, we remove the edges with a score below average.

We create separate graphs for sentences supporting each of the objects. We apply TextRank to each of them and then cluster them. Clustering divides the nodes in graphs by semantic similarity and thus allows identifying groups of sentences supporting a particular idea. Then, we apply TextRank again to each of the clusters separately and select the three most characteristic sentences from each cluster as produced by Chinese Whispers (Biemann, 2006), an iterative clustering algorithm, which assigns vertices to the most common class among their neighbors. Argumentative sentences selected in this way are displayed as a bullet-list after declaring the "winner" object of comparison.

Document-Retrieval-Based Method To compose an answer, CAM: First snippets takes the first sentence related to the "winner" object in CAM output. Then it finds a document corresponding to this sentence and extracts the surrounding context. The obtained context consists of 3 sentences and is considered to be a system answer.

Table 3: Evaluation of generation methods on the Yahoo! Answers. The best models of each type are highlighted.

Method	Type	ROUGE-1	ROUGE-2	ROUGE-3
CTRL: Question, len$\leq$100	Language Model	0.2423	**0.0226**	**0.0023**
CTRL: Which-better-x-y-for-z, len$\leq$100	Language Model	**0.2454**	0.0200	0.0021
CAM: First snippets	Doc.Retrieval	**0.2162**	**0.0167**	**0.0017**
CAM: Bullet points	Sent.Retrieval + Slots	**0.2298**	**0.0328**	**0.0040**
TextRank: Bullet points	Sent.Retrieval + Slots	0.2203	0.0238	0.0036
Templates	Object/Aspect Slots	0.1969	0.0195	0.0016

Template-Based Answer Besides the argumentative sentences, CAM extracts aspects and predicates from them. The predicates are adjectives or adverbs, which allows using templates of the following form: "I would prefer to use $Object_1$ because it is $Predicate_1$ and $Predicate_2$. In addition, it is $Predicate_3$, ..., and $Predicate_i$. However, you should also take into account that $Object_2$ is $Predicate_{i+1}$, ..., and $Predicate_k$". Here $Object_1$ is the winner of comparison.

5.2 Experiments

Evaluation Dataset To evaluate the answer generation module of our system, we use information extracted from Yahoo! Answers. Namely, we get a subset of L6–Yahoo! Answers Comprehensive Questions and Answers version 1.0 (multi-part) retrieved from Yahoo! Webscope. We take pairs of objects that we used for generating *Comparely* and extract a subset of questions from the Yahoo! Answers dataset which contains these objects, yielding 1,200 questions.

Additionally, we extract the answers to these questions, which are labeled by users as "best answer", and use them to evaluate our NLG methods.

Evaluation Metric Generated and reference texts are usually compared by a number of matched N-grams: BLUE (precision), ROUGE (recall), METEOR (F-score). For the all-round representation of the similarity of the text, we select F1 score from ROUGE-N outputs as an evaluation metric. We evaluate our generation models on the Yahoo! Answers dataset using the "best answer" (defined by users) as the reference.

Discussion of Results Evaluation results are provided in Table 3. CTRL models receive the highest ROUGE-1 scores that describe overlapping of single words, and CTRL's high performance relative to it can be explained by the fact that the pre-trained language model stores information about a vast dictionary and, with some probability, yields the words that are placed in the standard answer. While the language-model-based system may yield grammatically correct answers, they may not necessarily satisfy the information need of the user. For example, the CTRL answers the question "What should I eat an orange or an apple?" with "It is simple: eat what you like and don't worry about it."

Despite having low ROUGE-1, sentence retrieval-based approaches (Text Rank: Bullet points, CAM: Bullet points) have consistently higher ROUGE-2 and ROUGE-3. The generated answers are more structured and built on sentences marked by the system as comparative. They often contain typical 2-gram and 3-gram sequences as found in explanations. Answers from CAM: First snippets, consisting of a single comparative sentences only, perform worse on all metrics. Interestingly, CAM: Bullet points has better performance than TextRank: Bullet points. It could indicate that modeling relevance by a standard index provides more accurate results than clustering. Meanwhile, template-based generation performs poorly. This indicates that the grammatical structure is essential for the answer generation task.

We choose 50 random sentences from the Yahoo! Answers dataset as described in Section 6 and calculate ROUGE-N scores for every generation method and Yahoo!'s "best answers". For each group of methods, we select one providing the best result—CTRL: Question 100, CAM: First snippets, and CAM: Bullet points—and add them to the system demonstration engine.

6 User Study

To additionally evaluate the proposed answer generation methods, we also collect human assessments in a small user study for the three models with the highest ROUGE scores (CTRL: Question 100,

Table 4: User study results for answer completeness and fluency (30 questions, 3-point Likert scales).

	Answers a question (%)			Answer fluency (%)		
Method	Complete	Partial	Does not	Complete	Partial	Not fluent
Yahoo! Best Answer	62	28	10	86	6	8
CTRL: Question 100	30	37	33	80	12	8
CAM: Bullet points	28	58	14	22	48	30
CAM: First snippets	23	49	28	27	38	35

CAM: Bullet points, and CAM: First snippets).

Experimental Setup For our study, we randomly sampled 30 comparative questions from the Yahoo! Answers dataset and generated answers using three methods: CTRL: Question 100, CAM: Bullet points, and CAM: First snippets. Additionally, since we used Yahoo!'s "best answers" as ground truth for automatic evaluation, we asked our participants to also assess the quality of the human "best answers". For the user study, we internally recruited five (under-)graduate students. We focused on the two answer evaluation criteria: (1) Whether an answer is complete ("Does it answer the question?") and (2) how fluently it is written. The 120 question–answer pairs (3 generated answers and Yahoo!'s "best answer" for 30 questions) were randomly ordered, and the participants had to rate the answer completeness and fluency on a three-point Likert scale (3: fully answers/fluent, 2: partially answers/fluent, and 1: does not answer/not fluent at all).

Results and Discussion The inter-annotator agreement shows a slight overall agreement between the five annotators (Fleiss' κ = 0.20 for answer completeness and κ = 0.13 for fluency) such that we decided to increase the reliability by calculating the κ-scores for all combinations of three or four annotators. We then decided to include only the three participants with the highest agreement (κ = 0.32 for answer completeness and 0.30 for fluency; both fair agreement) and to remove the two "outlier" participants from the study.

Table 4 summarizes the study results as the ratio of votes collected from the three annotators (we cannot use majority voting since about 60% of the question-answer pairs do not have a majority vote). Not surprisingly, the human-written answers are perceived as the most complete and fluent. The participants were almost equally satisfied with the answers generated by CTRL: Question 100 and CAM: Bullet points. However, they assessed the CTRL answers as much more fluent. Interestingly, the relatively low inter-annotator agreement might indicate that humans have different perceptions of answer completeness and fluency (even some "best answers" were rated as incomplete and not fluent). For completeness, we calculated the statistical significance of the user study results using Bonferroni corrected p-values. For the pair CTRL: Question 100 (our best NLG model) and the Yahoo! Best Answer: $p \ll 0.05$ for the answer completeness and $p \gg 0.05$ for the answer fluency. For the CTRL model, Pearson's r = 0.121 between the answer completeness and fluency (small correlation), and for the "best answers", r = 0.407 (medium correlation). The results show that our proposed system is almost as fluent as the human-written answers but still needs some improvement in terms of adequacy.

7 Conclusion

We present a comparative question answering system targeted at answering comparative questions, such as "Is X better than Y with respect to Z?". Our system is based on the Comparative Argument Mining (CAM) system—a tool that retrieves from a large corpus textual comparative arguments for two to-be-compared objects. We extend CAM with an NLU module that identifies objects and aspects in a user textual query and highlights them in the answer, and a generation module that gives a concise and coherent answer based on the retrieved information. Evaluation of generation methods showed that a CTRL-based answer generation has a better performance with respect to ROUGE-1, and Sentence Retrieval Methods provide superior ROUGE-2 and ROUGE-3 scores. We hope that the presented testbed for comparative QA and the set of baseline approaches will pave the way for further research.

Acknowledgments

This work was partially supported by the DFG through the project "ACQuA: Answering Comparative Questions with Arguments" (grants BI 1544/7-1 and HA 5851/2-1) as part of the priority program "RATIO: Robust Argumentation Machines" (SPP 1999). The demo is hosted using the Zhores supercomputer infrastructure (Zacharov et al., 2019). We thank the following Skoltech students who conducted several preliminary experiments related to this work as a part of their research assignments: Anna Shalova, Nikita Borovkov, Filipp Furaev, and Anton Razzhigaev. Finally, we thank Artem Shelmanov for providing a training script for the RoBERTa-based sequence tagger.

References

Martin Andrews and Sam Witteveen. 2019. Unsupervised natural question answering with a small model. In *Proceedings of the Second Workshop on Fact Extraction and VERification (FEVER)*, pages 34–38, Hong Kong, China.

Jatin Arora, Sumit Agrawal, Pawan Goyal, and Sayan Pathak. 2017. Extracting entities of interest from comparative product reviews. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, pages 1975–1978, New York, NY, USA. Association for Computing Machinery.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Chris Biemann. 2006. Chinese whispers - an efficient graph clustering algorithm and its application to natural language processing problems. In *Proceedings of TextGraphs: the First Workshop on Graph Based Methods for Natural Language Processing*, pages 73–80, New York City. Association for Computational Linguistics.

Alexander Bondarenko, Pavel Braslavski, Michael Völske, Rami Aly, Maik Fröbe, Alexander Panchenko, Chris Biemann, Benno Stein, and Matthias Hagen. 2020a. Comparative web search questions. In *Proceedings of the 13th ACM International Conference on Web Search and Data Mining (WSDM 2020)*, pages 52–60, Houston, USA. Association for Computing Machinery.

Alexander Bondarenko, Maik Fröbe, Meriem Beloucif, Lukas Gienapp, Yamen Ajjour, Alexander Panchenko, Chris Biemann, Benno Stein, Henning Wachsmuth, Martin Potthast, and Matthias Hagen. 2020b. Overview of Touché 2020: Argument retrieval. In *Working Notes Papers of the CLEF 2020 Evaluation Labs*, volume 2696 of *CEUR Workshop Proceedings*.

Alexander Bondarenko, Lukas Gienapp, Maik Fröbe, Meriem Beloucif, Yamen Ajjour, Alexander Panchenko, Chris Biemann, Benno Stein, Henning Wachsmuth, Martin Potthast, and Matthias Hagen. 2021. Overview of Touché 2021: Argument retrieval. In *Proceedings of the 43rd European Conference on IR Research (ECIR 2021)*, volume 12036 of *Lecture Notes in Computer Science*, Berlin Heidelberg New York. Springer.

Maximin Coavoux, Hady Elsahar, and Matthias Gallé. 2019. Unsupervised aspect-based multi-document abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 42–47, Hong Kong, China. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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, NAACL-HLT 2019*, pages 4171–4186. Association for Computational Linguistics.

Ondřej Dušek and Filip Jurčíček. 2016. A context-aware natural language generator for dialogue systems. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 185–190, Los Angeles, CA, USA. Association for Computational Linguistics.

Qiaoqing Fan and Yu Fang. 2017. An answer summarization method based on keyword extraction. In *BIO Web of Conferences*, volume 8, pages 30–37. EDP Sciences.

Markus Freitag and Scott Roy. 2018. Unsupervised natural language generation with denoising autoencoders. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3922–3929, Brussels, Belgium. Association for Computational Linguistics.

Murthy Ganapathibhotla and Bing Liu. 2008. Mining opinions in comparative sentences. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 241–248, Manchester, UK. Coling 2008 Organizing Committee.

Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P Vetrov, and Andrew G Wilson. 2018. Loss surfaces, mode connectivity, and fast ensembling of dnns. In *Advances in Neural Information Processing Systems*, volume 31, Montreal, Quebec, Canada. Curran Associates, Inc.

Ian J. Goodfellow, Oriol Vinyals, and Andrew M. Saxe. 2015. Qualitatively characterizing neural network optimization problems.

Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. A unified model for extractive and abstractive summarization using inconsistency loss. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 132–141, Melbourne, Australia. Association for Computational Linguistics.

Masaru Isonuma, Junichiro Mori, and Ichiro Sakata. 2019. Unsupervised Neural Single-Document Summarization of Reviews via Learning Latent Discourse Structure and its Ranking. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019*, pages 2142–2152, Florence, Italy. Association for Computational Linguistics.

Nitin Jindal and Bing Liu. 2006. Mining comparative sentences and relations. In *Proceedings, The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference*, pages 1331–1336, Boston, Massachusetts. Innovative Applications of Artificial Intelligence (AAAI) Press.

Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL: A conditional transformer language model for controllable generation. *CoRR*, abs/1909.05858.

Wiltrud Kessler and Jonas Kuhn. 2014. A corpus of comparisons in product reviews. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 2242–2248, Reykjavik, Iceland. European Language Resources Association (ELRA).

Xin Li and Dan Roth. 2006. Learning question classifiers: the role of semantic information. *Natural Language Engineering*, 12(3):229–249.

Marina Litvak and Mark Last. 2008. Graph-based keyword extraction for single-document summarization. In *Coling 2008: Proceedings of the workshop Multi-source Multilingual Information Extraction and Summarization*, pages 17–24, Manchester, UK. Coling 2008 Organizing Committee.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Chirantana Mallick, Ajit Kumar Das, Madhurima Dutta, Asit Kumar Das, and Apurba Sarkar. 2019. Graph-based text summarization using modified textrank. In *Soft Computing in Data Analytics*, pages 137–146. Springer.

Rada Mihalcea. 2004. Graph-based ranking algorithms for sentence extraction, applied to text summarization. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 170–173, Barcelona, Spain. Association for Computational Linguistics.

Kyosuke Nishida, Itsumi Saito, Kosuke Nishida, Kazutoshi Shinoda, Atsushi Otsuka, Hisako Asano, and Junji Tomita. 2019. Multi-style generative reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2273–2284, Florence, Italy. Association for Computational Linguistics.

Alexander Panchenko, Alexander Bondarenko, Mirco Franzek, Matthias Hagen, and Chris Biemann. 2019. Categorizing comparative sentences. In *Proceedings of the 6th Workshop on Argument Mining*, pages 136–145, Florence, Italy. Association for Computational Linguistics.

Alexander Panchenko, Eugen Ruppert, Stefano Faralli, Simone P. Ponzetto, and Chris Biemann. 2018. Building a web-scale dependency-parsed corpus from CommonCrawl. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, pages 1816–1823, Miyazaki, Japan. European Language Resources Association (ELRA).

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018*, pages 2227–2237. Association for Computational Linguistics.

Matthias Schildwächter, Alexander Bondarenko, Julian Zenker, Matthias Hagen, Chris Biemann, and Alexander Panchenko. 2019. Answering comparative questions: Better than ten-blue-links? In *Proceedings of the 2019 Conference on Human Information Interaction and Retrieval (CHIIR 2019)*, pages 361–365. Association for Computing Machinery.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Jian-Tao Sun, Xuanhui Wang, Dou Shen, Hua-Jun Zeng, and Zheng Chen. 2006. CWS: A comparative web search system. In *Proceedings of the 15th* *international conference on World Wide Web, WWW 2006, Edinburgh, Scotland, UK, May 23-26, 2006*, pages 467–476. Association for Computing Machinery.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems*, pages 3104–3112, Montreal, Quebec, Canada.

Charles Sutton and Andrew McCallum. 2012. An introduction to conditional random fields. *Found. Trends Mach. Learn.*, 4(4):267–373.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008, Long Beach, CA, USA. Curran Associates, Inc.

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 *4th Workshop on Argument Mining (ArgMining 2017) at EMNLP*, pages 49–59, Copenhagen, Denmark. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Seid Muhie Yimam, Iryna Gurevych, Richard Eckart de Castilho, and Chris Biemann. 2013. WebAnno: A flexible, web-based and visually supported system for distributed annotations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 1–6, Sofia, Bulgaria. Association for Computational Linguistics.

Igor Zacharov, Rinat Arslanov, Maksim Gunin, Daniil Stefonishin, Andrey Bykov, Sergey Pavlov, Oleg Panarin, Anton Maliutin, Sergey Rykovanov, and Maxim Fedorov. 2019. "Zhores"—Petaflops supercomputer for data-driven modeling, machine learning and artificial intelligence installed in Skolkovo Institute of Science and Technology. *Open Engineering*, 9(1):512–520.