CAM 2.0: End-to-End Open Domain Comparative Question Answering System

Ahmad Shallouf[1*], Hanna Herasimchyk[1*], Mikhail Salnikov[2,3*], Rudy Garrido Veliz[1*], Natia Mestvirishvili[1*], Alexander Panchenko[2,3], Chris Biemann[1], Irina Nikishina[1]

[1]Universität Hamburg, [2]Skoltech, [3]AIRI

{name.surname}@studium.uni-hamburg.de (author2, author4, author5)

{name.surname}@uni-hamburg.de (author1, author7, author8)

{m.salnikov, a.panchenko}@skol.tech

Abstract

Comparative Question Answering (CompQA) is a Natural Language Processing task that combines Question Answering and Argument Mining approaches to answer subjective comparative questions in an efficient argumentative manner. In this paper, we present an end-to-end (full pipeline) system for answering comparative questions called **CAM 2.0** as well as a public leaderboard called **CompUGE** that unifies the existing datasets under a single easy-to-use evaluation suite. As compared to previous web-form-based CompQA systems, it features question identification, object and aspect labeling, stance classification, and summarization using up-to-date models. We also select the most time- and memory-effective pipeline by comparing separately fine-tuned Transformer Encoder models which show state-of-the-art performance on the subtasks with Generative LLMs in few-shot and LoRA setups. We also conduct a user study for a whole-system evaluation.

Keywords: comparative question answering, system demonstration, question answering, question classification, sequence tagging, stance classification, multi-sentence summarization

1. Introduction

The problem of choice has always been considered pressing for modern society: what to wear, which phone to buy, where to go on vacation, and which Large Language Model (LLM) to fine-tune. Multiple services[1,2] are quite successful in providing detailed comparisons of food, gadgets, insurance companies. Moreover, ChatGPT[3] and similar systems based on LLMs might also be queried for a comparison. However, such systems allowing comparison might suffer from one or more shortcomings: (i) limited domain (database / corpus) which ignores the rich textual content available on the web; (ii) unclear origin of the statements and possible hallucination; (iii) no open access to the system; (iv) not applicable to / cannot be transferred to other domains.

In terms of Natural Language Processing, Comparative Question Answering lies concurrently in the field of Question Answering and Argument Mining. The reason for that lies in the subjective nature of such questions: we do not only aim to choose between the objects presented in the question but also to support the answer with relevant arguments to explain our choice. For instance, for the question *"Which phone should I buy: iPhone or Android?"* the answer *"iPhone"* would not be satisfying and self-explanatory at all. Moreover, we might also want to take all viewpoints into account and consider arguments in favor of the object that was not considered as a "winner".

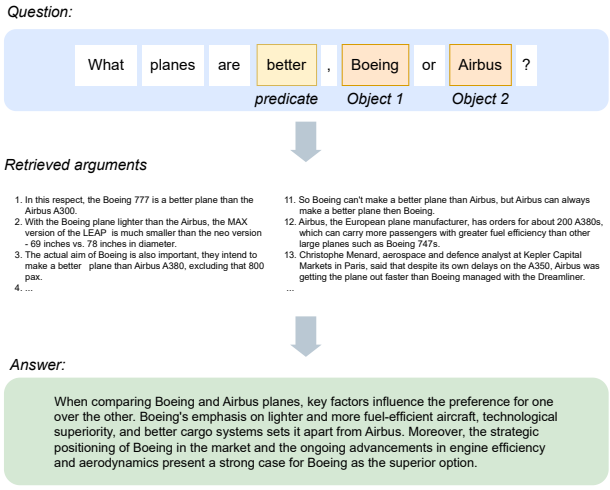

Figure 1: Example of the Comparative Question Answering task. The answer is based on the retrieved arguments from the indexed data.

One of the existing solutions to the shortcomings mentioned above is Comparative Argumentative Machine[4] (CAM) (Schildwächter et al., 2019), a tool

*These authors contributed equally to this work

[1]https://versus.com/en

[2]https://foodstruct.com/compare

[3]http://chat.openai.com

[4]http://ltdemos.informatik.uni-hamburg.de/cam

LREC-COLING 2024, pages 2657–2672

for answering comparative general-domain questions based on information extracted from the web-scale Common Crawl[5]. In this system, the user enters two comparison target objects and optionally aspects and gets a score bar with the overall distribution for two objects as well as sentences retrieved from the Common Crawl providing decision-making support. However, CAM cannot work with natural language questions and does not provide a coherent and detailed answer in the natural language summarising all the arguments presented. Moreover, as CAM was developed in 2019, some technologies of its pipeline need to be updated.

Therefore, we present **CAM 2.0**, an end-to-end (full pipeline) method for answering comparative questions in an argumentative manner. CAM 2.0 accepts natural language questions as input, extracts objects and aspects of comparison, and outputs the answer as a coherent text. It also benefits from the Information Retrieval system as the generated answer accumulates the arguments retrieved from Common Crawl. Moreover, the current research on comparative question answering yielded several relatively independent datasets not connected possibly hampering further progress in this field. We unify the existing datasets under a single easy-to-use evaluation suite and provide a public leaderboard called **CompUGE** for further evaluation. Therefore, the contribution of the following paper is three-fold:

1. We present **CAM 2.0**, an end-to-end (full pipeline) method for Comparative Question Answering which comprises Natural Language Understanding, Information Retrieval, and Natural Language Generation modules with up-to-date language models. We also provide a demonstration system that employs the pipeline and a user study of the developed system.

2. We compare several approaches for each subtask to detect the most time- and memory-efficient pipeline, including Large Language models in few-shot and fine-tuning setups, discussing give-and-takes in effectiveness and memory consumption.

3. We present **CompUGE**[6]: a new benchmark with a set of all existing datasets for Comparative Question Answering and a new public leaderboard for evaluation.

We make all resources (demo, models, datasets, leaderboard, and code) available online[7].

[5] https://commoncrawl.org

[6] https://huggingface.co/spaces/uhhlt/CompUGE

[7] https://github.com/uhh-lt/cam-2.0

2. Related Work

In this section, we introduce the existing datasets and approaches related to Comparative Question Answering that were not covered above.

Comparative Question Identification is the binary classification task to identify whether the question is *Comparative* or *Not*. There exist several papers introducing such datasets for English (Li et al., 2010; Bondarenko et al., 2020a, 2022a; Beloucif et al., 2022) and Russian (Bondarenko et al., 2020a). Moreover, the Mintaka dataset for Knowledge Base Question Answering (Sen et al., 2022), also comprises questions labeled as comparative. Along with the datasets, the authors published baseline approaches to compare with machine learning classifiers, convolutional neural networks, and Transformer Encoders.

Object and Aspect Labeling is the sequence tagging task that aims at identifying certain entities (objects, aspects, predicates, etc.) in the comparative questions. We have encountered at least three different datasets on the task. The dataset Li et al. (2010) is not publicly available, where they aim at extracting objects of comparison. The other two datasets (Beloucif et al., 2022; Bondarenko et al., 2022a) have different annotation schema and tags, which means that they cannot be straightforwardly combined. Chekalina et al. (2021) also presents a dataset with objects and an aspect of comparison. However, it comprises labeled affirmative sentences instead of questions, which makes it inapplicable for our purposes according to (Bondarenko et al., 2022a).

Stance classification is one of the crucial subtasks of the whole pipeline, as we select relevant arguments and detect, in favor of which object the choice is made thanks to the class assigned at this step. There is a dataset published by Panchenko et al. (2019) along with the baseline classifier, which is outperformed by (Ma et al., 2020). Bondarenko et al. (2022a) also introduces datasets and tests them on several models. However, their text excerpts are quite large and are not consistent with the argumentative sentences for our task.

Comparative Summary Generation is quite recent and not a very widespread task. To the best of our knowledge, only two papers exist introducing the datasets and baseline approaches. Chekalina et al. (2021) presents comparative questions with their best answers from Yahoo!Answers and tests several unsupervised approaches like CTRL (Keskar et al., 2019) and template-based answers. Yu et al. (2023) pre-trains LLMs for comparative reasoning using prompts. Moreover, they introduce a dataset called “Diffen” that works exactly with the summarization of arguments for an object pair, but they do not make it public.

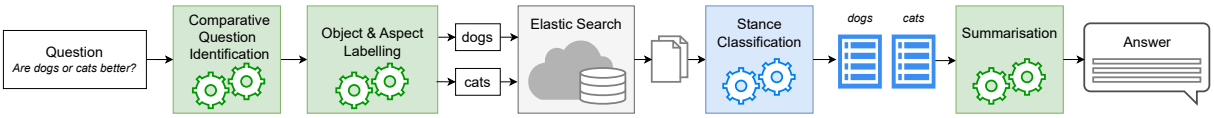

Figure 2: Pipeline of CAM 2.0: Comparative Question identification, Object and Aspect Labeling, Information Retrieval, Stance Classification, Summarization. Input is provided in the natural language and the output is the summary of the retrieved arguments.

Corpus	Task	Train	Dev	Test	Metrics
Bondarenko et al. (2020a)	Comparative Question Identification	10500	1350	3150	F1
Bondarenko et al. (2022a)		21869	2812	6561	
Sen et al. (2022)		14000	2000	4000	
Beloucif et al. (2022)		-	-	795	
Bondarenko et al. (2022a)	Object and Aspect labelling	2471	318	741	F1
Beloucif et al. (2022)		2141	275	642	
Chekalina et al. (2021)		2334	283	360	
Bondarenko et al. (2022a)	Stance classification	669	87	200	F1
Panchenko et al. (2019)		5183	576	1440	
Chekalina et al. (2021)	Summary generation	-	-	51	ROUGE, BERT-score

Table 1: The tasks included in CompUGE.

Touché at CLEF 2020-2022 competitions on Comparative Arguments (Bondarenko et al., 2020b, 2021, 2022b) could also be considered relevant for CompQA as they also aim at retrieving relevant arguments. For instance, Abye et al. (2020) used mainly a fine-tuned BERT model to detect the type of arguments and later implemented several measures per document for re-ranking the results. Huck (2020) applied a combination of *ChatNoir*[8] and *Okapi BM25*, while Sievers (2020) applied GPT-2 (Radford et al., 2019) to create query mocks aiding the information deficit from the users. Main solutions in 2021 used DistilBERT-based models for argument retrieval (Alhamzeh et al., 2021) and the methodology-varied approach with feature selection (Akiki et al., 2021). Arnhold et al. (2022) used a pre-trained DistilBERT model combined with the TARGER-API (Chernodub et al., 2019) to judge the argument quality. Chimetto et al. (2022) decided to use adjectives and comparative adjectives frequency to rank the quality of the arguments. Another solution by Chekalina and Panchenko (2022) employed the deep language model ColBERT [9].

Overall, this paper aims to evaluate the best practices for CompQA subtasks and assess their performance on relevant datasets mentioned above.

3. CompUGE: Benchmark for Comparative Question Answering

As it has been seen in Section 2, there exist multiple datasets for each task of the pipeline, however, we do not cover all of them in the paper. To overcome this limitation, we present **Comp**arative **U**nderstanding and **G**eneration **E**valuation (**CompUGE**): Benchmark for Comparative Question Answering for English. It consists of a public leaderboard built around four CompQA tasks, drawing on existing data, accompanied by performance metrics, and an analysis toolkit. Table 1 represents the tasks and dataset included in the benchmark. We plan to keep it up-to-date and extend it with new arising datasets.

4. CAM 2.0 Pipeline

Figure 2 presents the CAM 2.0 pipeline. It consists of the comparative question identification module, object and aspect labeling, information retrieval, stance classification, and summarization modules.

4.1. Comparative Question Identification

The first step of our pipeline is to classify whether the question is comparative. Otherwise, we do not further process non-comparative questions. To select the best classifying model for comparative question identification, we use the dataset from Webis-2022 (Bondarenko et al., 2022a) as the latest and the largest open dataset available among those listed in Section 2. We do not perform 10-fold cross-validation like the authors of the paper due to the high resource consumption of the models we used. Therefore, we have created and published a test split for evaluation consisting of 70% for training, 9% for validation, and 21% for testing. We compare several models: Encoder-based Transformers and Generative Transformers in a few-shot setup and fined-tuned on the target dataset. As with previous approaches, we compare with the best-performing

[8] https://webis.de/research/chatnoir.html

[9] https://github.com/stanford-futuredata/ColBERT

Model	Precision	Recall	F1	Params
Bondarenko et al. (2022a) (albert/albert-large-v1)	0.9250±0.0104	0.9116±0.0090	0.9179±0.0006	17M
distilbert/distilbert-base-uncased-finetuned-sst-2-english	0.9244±0.0113	0.9131±0.0094	0.9186±0.0051	67M
prajjwal1/bert-tiny	0.9235±0.0099	0.8759±0.0027	0.8990±0.0049	4M
meta-llama/Llama-2-7b-chat-hf (90-shot)	0.8592±0.0210	0.2774±0.1077	0.3065±0.1461	7B
lmsys/vicuna-7b-v1.5 (90-shot)	0.8917±0.0108	0.9024±0.0120	0.8824±0.0037	7B
meta-llama/Llama-2-7b-chat-hf (LoRA)	0.9762±0.0040	0.9762±0.0041	0.9762±0.0041	7B+33M
lmsys/vicuna-7b-v1.5 (LoRA)	**0.9777±0.0001**	**0.9777±0.0002**	**0.9777±0.0001**	7B+33M

Table 2: Results for Comparative Question Identification on the Webis-2022 dataset (Bondarenko et al., 2022a). Other models and results are found in Table 8 in the Appendix.

Sentence	which was the first national park
Tagged sentence	[ID-0]which [ID-1]was [ID-2]the [ID-3]first [ID-4]national [ID-5]park
Target	[ID-0][O] [ID-1][O] [ID-2][O] [ID-3][PRED] [ID-4][OBJ] [ID-5][OBJ]

Table 3: Example of data format for Object and Aspect Labeling as a text-to-text problem for LLMs.

model from the paper (ALBERT-large[10]) the same hyperparameters stated in the paper.

In Table 2, we showcase two models selected from among the multiple ones we tested. DistilBERT (Sanh et al., 2020)[11], which was previously fine-tuned on the Sentiment Analysis dataset, delivers the best results. On the other hand, BERT-tiny[12] (Bhargava et al., 2021; Turc et al., 2019) offers the lowest number of parameters, making it efficient for use in a demonstration system. We also performed the hyperparameters search for both models [13]. As for the generative LLMs (Llama and Vicuna), we test them in both few-shot and fine-tuning setups. For few-shot experiments, we tested different numbers of examples with step $= 1$ up to $N = 10$, with step $= 5$ up to $N = 50$, and with step $= 10$ up to $N = 100$. We report the score with the best step $N = 90$. Multiple runs for this setup were done with sampling different examples for few-shot.

The results show that generative LLMs deliver the worst results in the few-shot setup and the best results with fine-tuning against all other approaches. Moreover, larger models yield better results. Therefore, the choice of the model for this step would highly depend on the deployment server capacity.

Error analysis shows that the misclassification happened with a very low frequency of around 1% of the time, and the majority were false negatives. It was noticed that some questions in the dataset could've been mislabelled, some examples can be seen in Appendix B, therefore, a review of the dataset would be beneficial.

[10] https://huggingface.co/albert-large-v1

[11] https://huggingface.co/distilbert

[12] https://huggingface.co/prajjwal1/bert-tiny

[13] Best hyperparameters found for both DistilBERT and BERT-tiny: learning rate $= 7.53539e-05$, epochs $= 6$, batch size $= 12$, max seq length $= 315$

4.2. Object and Aspect Labeling

The subsequent step in the pipeline involves the extraction of the objects and aspects from the identified comparative questions for comparison. To do that, we evaluate both the Encoder and Generative Transformers of the Webis-2022 dataset for sequence tagging. Following a similar approach to the previous step, we split the dataset into the train (70%), validation (9%), and test (21%) parts, publish it, and replicate a RoBERTa-large model using the hyperparameters specified in the paper Bondarenko et al. (2022a).

At this step, we experiment with 14 different Encoder-based models. The performance of the top four models on the split dataset is presented in Table 4. Due to resource constraints, we perform cross-validation on the best three models demonstrating the results in Table 14.

Large Language Models —LLama-2 and Vicuna — are both tested in the generative setup only (for few-shot and fine-tuning). To do that, we can convert the sequence labeling problem into a text-to-text problem, as suggested by (Raman et al., 2022). In this approach, we incorporate additional tokens to guide the language model in labeling each token. To indicate the token number in a sentence, we introduce a special token format *[ID-N]*, as demonstrated in Table 3. Additionally, for the few-shot version, we include three examples of input and the corresponding expected output labels in the language model's instruction. In addition, we add the following basic instruction for each version of the used model:

(1) You are a helpful assistant for sequence labeling with the following labels: OBJ - Object, ASP - Aspect, PRED - Predicate and O - none.

We can see that the Encoder-based models ex-

Model	F1-OBJ	F1-ASP	F1-PRED	F1-Mean	Params
Bondarenko et al. (2022a) (FacebookAI/roberta-large)	0.7946±0.0073	0.6433±0.0031	0.9406±0.0019	0.8249±0.0037	355M
FacebookAI/roberta-base	0.7696±0.0052	0.6078±0.0121	0.9446±0.0006	0.8078±0.0041	125M
microsoft/deberta-v3-base	0.7998±0.0061	0.6808±0.0029	0.9524±0.0042	0.8370±0.0049	184M
microsoft/deberta-v3-large	**0.8290±0.0077**	**0.6809±0.0018**	**0.9604±0.0009**	**0.8545±0.0032**	434M
google-bert/bert-base-uncased	0.7337±0.0058	0.5851±0.0075	0.9348±0.0079	0.7832±0.0025	109M
meta-llama/Llama-2-7b-chat-hf (3-shot)	0.2008±0.0641	0.0000±0.0000	0.6364±0.0120	0.3655±0.0320	7B
lmsys/vicuna-7b-v1.5 (3-shot)	0.0880±0.0705	0.0345±0.02753	0.3402±0.1366	0.1763±0.0712	7B
meta-llama/Llama-2-7b-chat-hf (generative setup)	0.4903±0.1084	0.3843±0.0615	0.7493±0.1081	0.5413±0.1533	7B
lmsys/vicuna-7b-v1.5 (generative setup)	0.4684±0.0815	0.3225±0.0802	0.6268±0.1519	0.4725±0.1243	7B

Table 4: Results for Object and Aspect Labeling on the Webis-2022 dataset (Bondarenko et al., 2022a). Other models and their results can be found in Table 9 in the Appendix. The used hyperparameters are mentioned in Table 11 in the Appendix.

Model	F1-BETTER	F1-WORSE	F1-NONE	F1-Mean	Params
Panchenko et al. (2019)	0.75	0.43	0.92	0.85	UNK
Ma et al. (2020)	0.7821	0.5872	0.9298	0.8743	UNK
google-bert/bert-base-uncased	0.8999±0.0078	0.7426±0.0254	0.9636±0.0038	0.8807±0.0088	109M
microsoft/deberta-v3-large	**0.9172±0.0157**	**0.8303±0.0440**	**0.9744±0.0036**	**0.9106±0.0065**	434M
meta-llama/Llama-2-7b-chat-hf (30-shot)	0.3636±0.0100	0.2170±0.0482	0.4406±0.0600	0.4075±0.0428	7B
lmsys/vicuna-7b-v1.5 (20-shot)	0.3343±0.0546	0.7740±0.0677	0.2097±0.0465	0.6428±0.0403	7B
meta-llama/Llama-2-7b-chat-hf (LoRA)	0.8473±0.0055	0.7143±0.0021	0.9426±0.0056	0.9073±0.0056	7B+33M
lmsys/vicuna-7b-v1.5 (LoRA)	0.8597±0.0056	0.7140±0.0067	0.9216±0.0056	0.9043±0.0002	7B+33M

Table 5: Results for Stance Classification dataset (Panchenko et al., 2019). Other models and their results can be found in Table 10 in the Appendix. Hyperparameters are presented in Table 12 in the Appendix.

hibit superior performance, surpassing even the titans with 7B parameters. The expected poor results for the generative models confirm their ineffectiveness for the sequence labeling task. Among all the Encoder-based models evaluated, DeBERTa-v3-large (He et al., 2023) outperformed all others across all metrics. However, it is worth mentioning that this model also has the highest number of parameters among the Encoder-based models. Furthermore, despite DeBERTa-v3-base (He et al., 2023) being significantly smaller than RoBERTa-large, on average, it outperforms other models and even the RoBERTa-large proposed in (Bondarenko et al., 2022a) for this task.

Therefore, based on our evaluations, DeBERTa-v3-large (He et al., 2023) emerges as the most effective model, demonstrating superior task performance. In case of limited server resources, smaller models like DeBERTa-v3-base (He et al., 2023) also prove to be quite efficient.

After the in-depth analysis of the best model, the primary weaknesses were identified. For example, the model frequently misclassifies non-entity tokens as objects or aspects, in 26% of the time, and incorrectly identifies the beginning of object entities in 21% of the time. Furthermore, the model often confuses the beginning of aspect entities with object entities or non-entity tokens, a problem that arises in 20% of cases. This could potentially be improved by training the system on a larger and more balanced dataset. The examples for each error class are presented in Appendix B.

4.3. Sentence Retrieving

After extracting objects and aspects, we look for their matches in the Common Crawl text. It is also important to mention, that for the final pipeline, we make use not only of the extracted objects but also of predicates, as both of the classes could be relevant for the search. We use Elastic Search[14] full-text index of a pre-processed (lemmatized) corpus containing 14.3 billion English sentences from the Common Crawl. To retrieve arguments, the index is queried for sentences matching the input objects and optionally aspects. The output is further filtered in terms of relevance for comparison.

4.4. Stance Classification

Once sentences with matching objects and aspects are extracted, we need to classify and rank them to get top-K arguments supporting each object. We train and test our models on the dataset from (Panchenko et al., 2019), using the split from the authors. We do not make use of Webis-2022, as its excerpts are larger. The dataset contains texts categorized into three distinct classes: *"BETTER"*, indicating that the first object mentioned is superior to the second; *"WORSE"*, signifying that the first object is inferior to the second; and *"NONE"*, denoting the absence of a direct comparison.

We test five models and present the best-performing two in Table 5. Encoder-based models are given both objects and a sentence separated

[14] https://www.elastic.co

Model	ROUGE-1	ROUGE-2	BERT-Score	Params
CTRL Which-better-x-y-for-z (Chekalina et al., 2021)	**0.2454**	0.0200	0.8214	1.63B
CAM bullet points (Chekalina et al., 2021)	0.2298	**0.0328**	0.8201	-
facebook/bart-large-cnn	0.1855±0.0047	0.0161±0.0010	**0.8270±0.0101**	406M
sshleifer/distilbart-cnn-6-6	0.1947±0.0021	0.0171±0.0008	0.8260±0.0105	230M
meta-llama/Llama-2-7b-chat-hf (no args)	0.1623±0.0009	0.0227±0.0005	0.8070±0.0135	7B
lmsys/vicuna-7b-v1.5 (no args)	0.1930±0.0028	0.0196±0.0010	0.8136±0.0143	7B
meta-llama/Llama-2-7b-chat-hf (2-shot)	0.1778±0.0050	0.0168±0.0021	0.8048±0.0127	7B
lmsys/vicuna-7b-v1.5 (2-shot)	0.1857±0.0026	0.0204±0.0015	0.8065±0.0164	7B
gpt-3.5-turbo (no args)	0.1658±0.0015	0.0200±0.0007	0.8125±0.0112	154B
gpt-3.5-turbo (2-shot)	0.1998±0.0023	0.0210±0.0009	0.8125±0.0101	154B

Table 6: Results for Summarization on Yahoo!Answer dataset (Chekalina et al., 2021). The results are compared against the answer marked as the "Best Answer" on the platform.

with *"[SEP]"* symbol. The example of the input is presented below:

(2) Lisp [SEP] Java [SEP] Common Lisp is a bit less strongly functionally-oriented, but it still supports it better than, say, C or Java, by having first-class functions and closures.

Llama-2 and Vicuna are trained using LoRA in *bf16* mode. We compile the input sentence for those models using the following template: Object 1: {object1}; Object 2: {object2}; {sentence}. Here, "{object1}" and "{object2}" are objects for comparison and "{sentence}" is the sentence which stance should be identified. An example with such a template is presented below:

(3) Object 1: Lisp; Object 2: Java; Common Lisp is a bit less strongly functionally-oriented, but it still supports it better than, say, C or Java, by having first-class functions and closures.

The models undergo fine-tuning with a classification head featuring three output neurons corresponding to the *"BETTER"*, *"WORSE"*, and *"NONE"* classes, respectively. The final prediction is derived using the softmax activation function.

From the results, we can see that DeBERTa-v3-large is the best-performing model for multi-class classification. Llama-2 and Vicuna used with LoRA demonstrate almost equal performance. Nevertheless, they contain many more parameters compared to Encoder-based models, which makes them infeasible to use.

For clarity, when aggregating arguments for each object, we consider both 'BETTER' and 'WORSE' classifications. Specifically, we tally "BETTER" arguments where the object in question is the first item as well as "WORSE" arguments where the object is the second item in the sentence. For example, when comparing Python with Java, both "Python is better than Java" (classified as "BETTER") and "Java is worse than Python" (classified as "WORSE") contribute to Python's favorability.

The in-depth analysis of the best model shows that the most common type of error is confusing the label "BETTER" with the label "NONE" (28.4%). The second is when the actual label is "NONE" and the predicted label is "WORSE" (19.7%). The model performs best at detecting the "NONE" label which is the majority of samples in the training set. Training the system on a more balanced dataset will likely improve these results. Some examples of each error type are presented in Appendix B.

4.5. Summarization

Following Chekalina et al. (2021) we choose several models for summarization and evaluate them on the Yahoo!Answers dataset in an unsupervised manner. We choose two BART models (Facebook BART-large (Lewis et al., 2020) and DistillBART[15] both fine-tuned on CNN). For these two models, we apply the following prompt:

(4) Summarize: {arguments}.

Here, "{arguments}" refers to 10 enumerated arguments from a prior step, separated by a new line.

However, our main goal for this subtask is to make use of the Generative Transformers, as they have shown strong capabilities in question answering, text summarization, and text generation in general. For Generative LLMs, we compare LLaMA-2 and Vicuna again, with 7B of parameters. In addition to that, we also run experiments with GPT-3.5 turbo from OpenAI. We try three setups: instruction with no arguments to compare objects using the model's knowledge, a zero- and a few-shot shot setup with retrieved arguments. Input templates can be seen below:

(5) Write a comparison summary of "{object1}" and "{object2}".

(6) [INST] «SYS» You are a helpful assistant. «/SYS»
Write a comparison summary of "{object1}"

[15] https://huggingface.co/sshleifer/distilbart-cnn-6-6

model name	summary
Yahoo.Best Answer	How can you even ask this question yet? Only the Xbox 360 is out at the moment and that hasn't even been tested by gamers enough to see truly how good or rubbish it is. You need to ask the question again when all three systems are out!
CTRL:does-x-better-y-for-z	I think that the answer to this question depends on what you mean by "better". For me, it comes down to two things: 1) how much of a difference does it make in terms of performance and 2) whether or not it's worth paying more for it. If you're looking at buying an Xbox One, then yes, I'd say go ahead and get one. But if you're just thinking about getting one, then no, don't bother. The only reason why I'm even considering one now is because my 360 has been acting up lately (again, probably due to overheating). And while I've had no problems with it, there are definitely things that could be improved upon. So far, though, there's nothing really compelling me to buy another Xbox. In fact, after playing around with some other consoles over the last few days, I can't see myself ever wanting to. Maybe I'll wait until they have something new out before making any decisions. What do you think? Are you planning on picking up an Xbox One? Let us know your thoughts in the comments.
facebook/bart-large-cnn	Microsoft has a much better (easier to use) SDK for games than Sony does. Microsoft made these updates go much faster, smoother, and less frequent than Sony ever did with the PlayStation 3. Microsoft had better games at launch than Sony and will also in 2014.
gpt-3.5-turbo (2-shot)	Microsoft and Sony are both major players in the technology industry, but they have their differences in terms of performance and capabilities. According to some arguments, Microsoft is considered to be faster and superior to Sony [1]. They have a better SDK for games [3], and their updates are smoother and less frequent [2]. Microsoft is also praised for their superior pacing [4] and better customer service [8]. On the other hand, Sony is regarded as a superior hardware manufacturer [11][12], and they are working harder for gamers [10]. Some argue that Sony is better at designing hardware [12][14] and has a superior position [9]. However, there are also arguments stating that Microsoft is inferior to Sony in terms of gaming [13][16], while others believe that Sony is better than Microsoft in every way [17]. Overall, personal preferences and experiences play a significant role in deciding which brand is better. Both companies have their strengths and weaknesses, and customers should consider their own needs and preferences when choosing between Microsoft and Sony. Arguments used: 1, 2, 3, 4, 8, 9, 10, 11, 12, 13, 14, 16, 17.
lmsys/vicuna-7b-v1.5 (no args)	Microsoft and Sony are two of the largest and most well-known technology companies in the world. While they both produce a wide range of products, including computers, gaming consoles, and mobile devices, there are some key differences between the two companies. One of the main differences between Microsoft and Sony is the focus of their businesses. Microsoft is primarily a software company, with a focus on developing and selling operating systems, productivity software, and other software products. Sony, on the other hand, is a diversified company that produces a wide range of products, including electronics, gaming consoles, movies, and music. Another key difference between the two companies is their approach to innovation. Microsoft is known for its focus on research and development, and has been a leader in the development of new technologies, such as the Windows operating system and the Xbox gaming console. Sony, on the other hand, is known for its focus on creating high-quality, innovative products, and has been a leader in the development of new technologies, such as the PlayStation gaming console and the Bravia line of televisions. Overall, while Microsoft and Sony both produce a wide range of technology products, they have some key differences in terms of their focus and approach to innovation. Microsoft is primarily a software company, while Sony is a diversified company that produces a wide range of products. Additionally, Microsoft is known for its focus on research and development, while Sony is known for its focus on creating high-quality, innovative products.
lmsys/vicuna-7b-v1.5 (2-shot)	Microsoft and Sony are two major companies in the technology industry, with a significant presence in the gaming market. Some argue that Microsoft is better and faster than Sony, with updates going smoother and less frequent [2]. Microsoft is also considered to have a better SDK for games [3], and their conference was thought to have better pacing [4]. Additionally, some believe that Microsoft has a superior position over Sony in terms of software tools [9]. However, others argue that Sony is a superior hardware manufacturer, much better than both Nintendo and Microsoft [11]. The PS4 is physically superior to the Xbox One, with better design [12]. Sony is also believed to be working harder for gamers than Microsoft and Nintendo [10]. In terms of gaming, some argue that Microsoft is inferior to Sony [13], and that Sony is superior to Microsoft in every way possible [17]. Sony was also considered to be a bit smarter than Microsoft in terms of their approach to gaming [18]. Ultimately, the preference between Microsoft and Sony comes down to personal opinions and experiences. Some may prefer Microsoft for its software tools and updates, while others may prefer Sony for its hardware design and gaming experience. Arguments used: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20

Table 7: Examples of the summaries for comparing "Microsoft" and "Sony": best answer from Yahoo!Answers, summaries generated with best-performing approach (Chekalina et al., 2021) and summaries generated with LLMs (Vicuna and GPT-3.5-turbo) in zero- and two-shot setups.

> and "{object2}". Summarize only relevant arguments from the list. After the summary, list the argument numbers you used below the text. Put citations in brackets inside the text. Do not even mention arguments that are not relevant to "{object1}" and "{object2}".
> {arguments}
> Answer:

Moreover, in order to control the model's hallucination, we required the output to contain citations of the arguments inside the generated text.

The results, presented in Table 6 are controversial. From what we can see, that approach by Chekalina et al. (2021) produces texts that are closest to the best answers from Yahoo!Answers. Interestingly, LlaMA-2 and GPT-3.5-turbo without arguments demonstrate very low results, which are only slightly improved when providing a list of arguments to rely on. Apparently, LLMs provide comparisons with more generalization than users

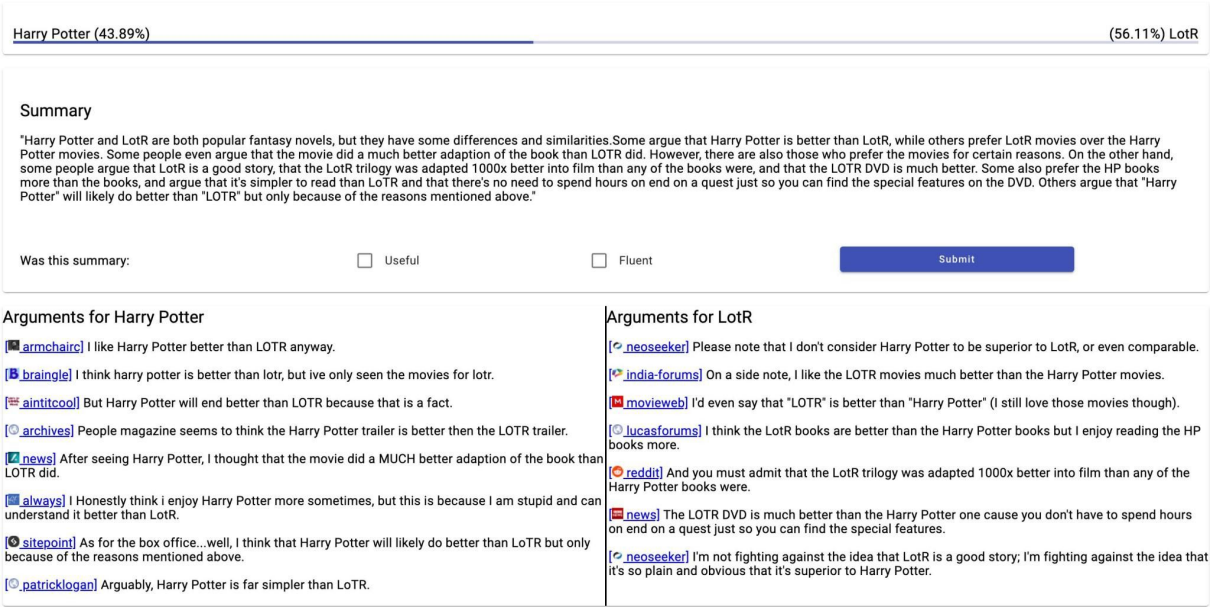

Figure 3: CAM 2.0 output for the question "What is better: Harry Potter or LotR?".

of Yahoo!Answers that usually share their own experience. This can be easily seen from Table 7 (and from further qualitative analysis): texts generated by LLMs are more abstract when compared to the best answers. Moreover, according to the user study in the previous paper (Chekalina et al., 2021), only 62% of answers do answer the questions completely and 86% of answers are fluent. As a result, with this evaluation pipeline, we measure not only the ability of the model to do a comparative summarization, but also "how similar the retrieved arguments are to the best answer from Yahoo!Answers" which is not exactly the purpose of our evaluation. Yahoo!Answers does not cover all aspects of comparison and might not mention arguments found in Common Crawl at all.

To evaluate the ability of the model to summarize the provided arguments, we make use of in-text argument citations. We ask three annotators who have expertise in computational linguistics to find the part of the text where the argument is cited and check whether the argument is cited correctly. There are three possible answers: (1) yes, the argument is relevant for the text and is cited correctly; (-1) no, the argument is relevant, but states the opposite; (0) no, the argument is irrelevant for this sentence. The annotations were made for 50 summaries from the Vicuna model, as it correctly reproduces the required structure of summary in comparison to LLaMA-2 which tends to copy-paste arguments instead of generating a coherent text. We achieve a very good inter-annotator agreement: Krippendorf's alpha is 0.794.

However, the results of the model citation injection are not very promising. First, we calculated the strong precision, which takes into account only label *"1"* and the precision score is **0.5**. If we consider *"-1"* labels as well, the score becomes higher and reaches **0.64**. We can see that the model does not show a strong ability to produce citations for the arguments used and such a dataset for model fine-tuning would be extremely beneficial. Error analysis shows that the main types of errors could be split in three main classes: "irrelevant arguments", "opposite arguments", "poor-quality argument". The examples are presented in Appendix B.

5. System Demonstration

We also created a full pipeline demonstration system, available online.[16] Each step of the system contains the most efficient approach introduced in the subsections above: we applied *vicuna-7b-v1.5* for Comparative Question Identification and *microsoft/deberta-v3-large* for Object and Aspect Identification and Stance classification.

The user interface consists of an input form for natural questions and an answer presentation component, which in turn consists of 5 smaller components. A component indicating whether the question is comparative or not, an object and aspect input component, a component with arguments, a final bar score, and a summary component. First of all, a user types a question that is classified as *Comparative* or *Not Comparative* and the class is displayed to the user, e.g. as in Figure 4. If the question is identified as comparative, the processing continues. Otherwise, it displays a message that the question is not comparative. If the question

[16] https://cam-v2.ltdemos.informatik.uni-hamburg.de

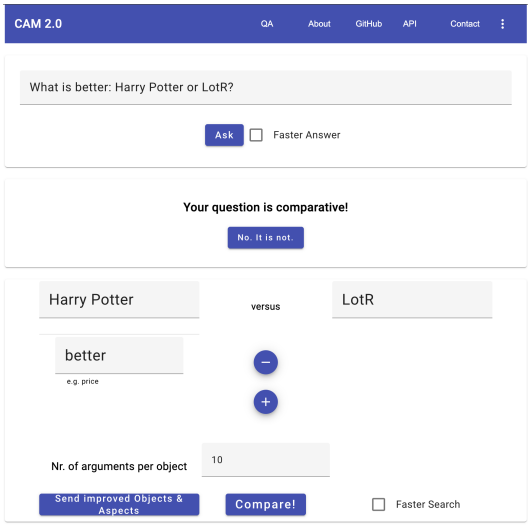

Figure 4: CAM 2.0 input form for natural language questions.

contains obscenity, the system will notify the user and will ask him to reformulate the question.

If the user does not agree with the classifier result, they can mark the wrong detection and continue or discontinue the pipeline. Such cases are stored for further investigation and improving the classifier. If the question is incorrectly classified, the user can provide feedback to the system using a designated button. After that, the misclassified sentence will be stored in the corresponding table in the PostgreSQL database[17] for further model improvement. The pipeline will be discontinued or continued from this point, depending on the question type.

The next component is divided into three parts, which can be seen in Figure 4. On the top, the user can see the extracted objects from the input question. By default, they are extracted from the input sentence using the object and aspect labeling model. Those objects can be manually corrected and sent as feedback to the system using a designated button. In the middle, the extracted aspects can be easily edited or added.

The answer presentation component is displayed in three different formats: a score bar, two columns for each object within the arguments, and a summary of the top-K sentences from each side. An example of such output is shown in Figure 3. The overall score distribution allows the user to grasp a general impression of the entered comparative question. To calculate it, we combine the Elastic Search scores of the retrieved and classified sentences for each object separately. We also display arguments for each object as a clickable link, which forwards to the source which contains the complete argument. Those arguments are also ranked in accordance with the Elastic Search score. A summary is generated using all the displayed arguments. and made visible to the user for an easy read.

[17] https://www.postgresql.org

6. User Study

To evaluate the whole pipeline, we select 50 questions with the objects labeled from the Touché at CLEF competition in 2022 (Bondarenko et al., 2022b) and ask those questions to our system. First of all, **100%** of those questions were classified as comparative. Second, our pipeline gained an F1-score of **1.0** for object labeling, correctly extracting all objects in the questions given. After that, we selected 28 questions with at least five arguments supporting each object and asked three annotators to read summaries and evaluate the quality of summaries. We removed argument citation from the generated text so that its quality does not affect the user's impression of the generated text. Four annotators were suggested to annotate the output of the model against two criteria: (i) whether the answer is helpful ("Does it help to make your conclusion about the objects?") and (ii) how fluently it is written. Each of the four annotators agreed on the same 22 summaries (**78.6%**) being helpful and the same 18 summaries (**64.3%**) being fluent. Although employing only four annotators may not suffice for drawing definitive conclusions about the system, the primary objective of our user study was to conduct an initial manual evaluation of the summary quality. We aimed to assess whether the generated summaries are both well-crafted and practically useful in facilitating decision-making. Figure 3 also includes checkboxes displayed post-output to gather additional user feedback.

7. Conclusion

This paper presents *CAM 2.0* — an end-to-end (full pipeline) system for Comparative Question Answering. It comprises several steps: comparative question identification, object and aspect identification, argument retrieval system, stance classification, and comparative summarization. We compare several Transformer Encoders and Generative Transformers for each classification and sequence labeling task to compare models in terms of space usage and efficiency. User study indicates that medium-sized Transformer Encoders deliver strong performance. Furthermore, fine-tuned versions of LLaMA-2 and Vicuna yield near-optimal quality, setting new state-of-the-art benchmarks. Regarding summary generation, while in-text argument referencing could improve, Generative Transformers already produce coherent texts that effectively aggregate answers. For future research, we aim to enhance the generation process and add new languages to the pipeline.

Limitations

We find the main limitation of our work as follows:

- Nowadays, dozens of large pre-trained generative models exist and we report results only on a few of them. It may be that some other base models used could further push the results. However, our goal was to show an example of how similar models are and not perform an exhaustive search of all models.

- As outlined in Section 3, multiple datasets are available to support various stages of the pipeline. We do not present the results for all possible datasets in the paper. However, we compute all models tested in the paper on our benchmark.

- We did not test the multilingual setting of our approach, which is possible if multilingual versions of sequence-to-sequence models are used, such as mT5 or mBERT. This is an important additional experiment to further validation of the method explored in our work.

- We also acknowledge the importance of filtering inappropriate content, and our texts for retrieval are mostly already filtered and we could add some word-based filters. Despite applying an obscenity filter for the input questions, we still need to pay more attention to the problem in future work, even if it is not the primary task of our research.

Ethics Consideration

In our work, we employ large-scale neural models like LLaMA-2 and Vicuna, which have been pre-trained on a diverse corpus that includes user-generated content. While authors of the models made an effort to filter toxic or biased content, the model itself still can contain certain biases, and as a consequence outputs of our methods may render such biases. Methodologically, however, it is straightforward to apply our techniques to other pre-trained models that have been debiased in the required manner.

Another ethical concern might be in the questions the users might ask that contain inappropriate (obscene / toxic / insulting) objects for comparison. Unfortunately, our system does not filter such questions automatically, however, there is a very low chance that sentences containing such objects will be found in the Common Crawl. Nevertheless, we plan to add filters in the next version of the system.

Acknowledgements

This work was 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).

Bibliographical References

Tinsaye Abye, Tilmann Sager, and Anna Juliane Triebel. 2020. An open-domain web search engine for answering comparative questions notebook for the touché lab on argument retrieval at clef 2020.

Christopher Akiki, Maik Fröbe, Matthias Hagen, and Martin Potthast. 2021. Learning to rank arguments with feature selection notebook for the touché lab on argument retrieval at clef 2021. *Touché Lab on Argument Retrieval at CLEF 2021*.

Alaa Alhamzeh, Mohamed Bouhaouel, Előd Egyed-Zsigmond, and Jelena Mitrović. 2021. Distilbert-based argumentation retrieval for answering comparative questions notebook for the touché lab on argument retrieval at clef 2021. *Touché Lab on Argument Retrieval at CLEF 2021*.

Niclas Arnhold, Philipp Rösner, and Tobias Xylander. 2022. Quality-aware argument re-ranking for comparative questions notebook for the touché lab on argument retrieval at clef 2022. *ouché Lab on Argument Retrieval at CLEF 2022*.

Meriem Beloucif, Seid Muhie Yimam, Steffen Stahlhacke, and Chris Biemann. 2022. Elvis vs. M. Jackson: Who has more albums? classification and identification of elements in comparative questions. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 3771–3779, Marseille, France. European Language Resources Association.

Prajjwal Bhargava. 2021. prajjwal1/bert-tiny.

Prajjwal Bhargava, Aleksandr Drozd, and Anna Rogers. 2021. Generalization in nli: Ways (not) to go beyond simple heuristics.

Alexander Bondarenko, Yamen Ajjour, Valentin Dittmar, Niklas Homann, Pavel Braslavski, and Matthias Hagen. 2022a. Towards understanding and answering comparative questions. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, WSDM ’22, page 66–74, New York, NY, USA. Association for Computing Machinery.

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 *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, pages 52–60. ACM.

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 *Experimental IR Meets Multilinguality, Multimodality, and Interaction. 11th International Conference of the CLEF Association (CLEF 2020)*, volume 12260 of *Lecture Notes in Computer Science*, pages 384–395, Berlin Heidelberg New York. Springer.

Alexander Bondarenko, Maik Fröbe, Johannes Kiesel, Shahbaz Syed, Timon Gurcke, Meriem Beloucif, Alexander Panchenko, Chris Biemann, Benno Stein, Henning Wachsmuth, Martin Potthast, and Matthias Hagen. 2022b. Overview of Touché 2022: Argument Retrieval. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction. 13th International Conference of the CLEF Association (CLEF 2022)*, Lecture Notes in Computer Science, Berlin Heidelberg New York. Springer.

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 *Experimental IR Meets Multilinguality, Multimodality, and Interaction. 12th International Conference of the CLEF Association (CLEF 2021)*, volume 12880 of *Lecture Notes in Computer Science*, pages 450–467, Berlin Heidelberg New York. Springer.

Viktoriia Chekalina, Alexander Bondarenko, Chris Biemann, Meriem Beloucif, Varvara Logacheva, and Alexander Panchenko. 2021. Which is better for deep learning: Python or MATLAB? answering comparative questions in natural language. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 302–311, Online. Association for Computational Linguistics.

Viktoriia Chekalina and Alexander Panchenko. 2022. Retrieving comparative arguments using deep language models notebook for the touché lab on argument retrieval at clef 2022. *Touché Lab on Argument Retrieval at CLEF 2022*.

Artem Chernodub, Oleksiy Oliynyk, Philipp Heidenreich, Alexander Bondarenko, Matthias Hagen, Chris Biemann, and Alexander Panchenko. 2019. TARGER: Neural argument mining at your fingertips. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 195–200, Florence, Italy. Association for Computational Linguistics.

Alessandro Chimetto, Davide Peressoni, Enrico Sabbatini, Giovanni Tommasin, Marco Varotto, Alessio Zanardelli, and Nicola Ferro. 2022. Seupd@clef: Team hextech on argument retrieval for comparative questions. the importance of adjectives in documents quality evaluation notebook for the touché lab on argument retrieval at clef 2022. *Touché Lab on Argument Retrieval at CLEF 2022*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2023. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: decoding-enhanced bert with disentangled attention. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

HF Canonical Model Maintainers. 2022. distilbert-base-uncased-finetuned-sst-2-english (revision bfdd146).

Johannes Huck. 2020. Development of a search engine to answer comparative queries notebook for the touché lab on argument retrieval at clef 2020. *Touché Lab on Argument Retrieval at CLEF 2020*.

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.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Shasha Li, Chin-Yew Lin, Young-In Song, and Zhoujun Li. 2010. Comparable entity mining from

comparative questions. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 650–658, Uppsala, Sweden. Association for Computational Linguistics.

Nianzu Ma, Sahisnu Mazumder, Hao Wang, and Bing Liu. 2020. Entity-aware dependency-based deep graph attention network for comparative preference classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 5782–5788. 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.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Karthik Raman, Iftekhar Naim, Jiecao Chen, Kazuma Hashimoto, Kiran Yalasangi, and Krishna Srinivasan. 2022. Transforming sequence tagging into A seq2seq task. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 11856–11874. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

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, Glasgow, Scotland, UK, March 10-14, 2019*, pages 361–365. ACM.

Priyanka Sen, Alham Fikri Aji, and Amir Saffari. 2022. Mintaka: A complex, natural, and multilingual dataset for end-to-end question answering. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1604–1619, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Bjarne Sievers. 2020. Question answering for comparative questions with gpt-2 notebook for touché at clef 2020. *Touché Lab on Argument Retrieval at CLEF 2020*.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: The impact of student initialization on knowledge distillation. *CoRR*, abs/1908.08962.

Mengxia Yu, Zhihan Zhang, Wenhao Yu, and Meng Jiang. 2023. Pre-training language models for comparative reasoning. *CoRR*, abs/2305.14457.

A. Extended Experiment Results

In this section, we present the extended experiment results with a wide variety of Transformer models and present the results for each task separately. Tables 8 to 10 present the scores on the fixed test sets, while 13-14 present the cross-validation results on the original datasets. Table 11 list the hyperparameters for each model for the Object and Aspect Identification task.

Model	Precision	Recall	F1
Bondarenko et al. (2022a) (albert/albert-large-v1)	**0.9250±0.0104**	0.9116±0.0090	0.9179±0.0006
distilbert/distilbert-base-uncased-finetuned-sst-2-english	0.9244±0.0113	0.9131±0.0094	0.9186±0.0051
prajjwal1/bert-tiny	0.9235±0.0099	0.8759±0.0027	0.8990±0.0049
distilbert-base-uncased	0.9090±0.0057	0.9345±0.0058	0.9216±0.0037
FacebookAI/roberta-base	0.8903±0.0116	**0.9498±0.0122**	0.9190±0.0024
microsoft/deberta-base	0.9012±0.0125	0.9439±0.0059	**0.9219±0.0041**

Table 8: Results for Comparative Question Identification on the Webis-2022 dataset (Bondarenko et al., 2022a), all models using Transformers.

Model	F1-OBJ	F1-ASP	F1-PRED	F1-Mean	Params
Bondarenko et al. (2022a) (FacebookAI/roberta-large)	0.7946±0.0073	0.6433±0.0031	0.9406±0.0019	0.8249±0.0037	355M
FacebookAI/roberta-base	0.7696±0.0052	0.6078±0.0121	0.9446±0.0006	0.8078±0.0041	125M
Jean-Baptiste/roberta-large-ner-english	0.7682±0.0169	0.6464±0.0166	0.9370±0.0050	0.8107±0.0128	355M
albert/albert-base-v2	0.7802±0.0075	0.5935±0.0308	0.9389±0.0047	0.8077±0.0099	12M
microsoft/deberta-v3-base	0.7998±0.0061	0.6808±0.0029	0.9524±0.0042	0.8370±0.0049	184M
microsoft/deberta-v3-large	**0.8290±0.0077**	**0.6809±0.0018**	**0.9604±0.0009**	**0.8545±0.0032**	434M
distilbert/distilbert-base-uncased	0.6921±0.0059	0.5609±0.0177	0.9240±0.0029	0.7537±0.0024	67M
Davlan/distilbert-base-multilingual-cased-ner-hrl	0.6595±0.0179	0.5009±0.0199	0.9045±0.0027	0.7216±0.0112	135M
google-bert/bert-base-uncased	0.7337±0.0058	0.5851±0.0075	0.9348±0.0079	0.7832±0.0025	109M
dslim/bert-base-NER-uncased	0.7353±0.0051	0.5954±0.0024	0.9385±0.0079	0.7866±0.0034	109M
prajjwal1/bert-medium	0.7375±0.0031	0.5750±0.0070	0.9337±0.0036	0.7830±0.0010	42M
prajjwal1/bert-small	0.7218±0.0070	0.5790±0.0085	0.9265±0.0029	0.7725±0.0034	29M
prajjwal1/bert-mini	0.6687±0.0085	0.5148±0.0171	0.9234±0.0005	0.7359±0.0053	12M
prajjwal1/bert-tiny	0.5458±0.0028	0.3875±0.0108	0.8889±0.0064	0.6432±0.0012	5M

Table 9: Encoder-based Transformer Models Results for Object and Aspect Labeling on the Webis-2022 dataset (Bondarenko et al., 2022a).

Model	F1-BETTER	F1-WORSE	F1-NONE	F1-Mean	Params
microsoft/deberta-v3-base	0.8952±0.0226	0.8101±0.0578	0.9685±0.0067	0.8948±0.0134	184M
microsoft/deberta-v3-large	0.9172±0.0157	**0.8303±0.0440**	**0.9744±0.0036**	**0.9106±0.0065**	434M
FacebookAI/roberta-base	0.9195±0.0065	0.8113±0.0341	0.9670±0.0038	0.8998±0.0034	125M
FacebookAI/roberta-large	**0.9216±0.0075**	0.8127±0.0472	0.9702±0.0054	0.9047±0.0022	355M
google-bert/bert-base-cased	0.8999±0.0078	0.7426±0.0254	0.9636±0.0038	0.8807±0.0088	109M

Table 10: Encoder-based Transformer Models Results for Stance Classification dataset (Panchenko et al., 2019).

Model	train_batch_size	num_train_epochs	weight_decay	warmup_steps	learning_rate
microsoft/deberta-v3-base	16	8	0.01	100	0.00007
microsoft/deberta-v3-large	16	11	0.01	100	0.00005
FacebookAI/roberta-base	16	8	0.0001	200	0.0001
FacebookAI/roberta-large	8	10	0.001	100	0.00002
Jean-Baptiste/roberta-large-ner-english	16	8	0.001	200	0.00005
albert/albert-base-v2	16	8	0.1	400	0.00007
google-bert/bert-base-uncased	16	7	0.1	300	0.00005
dslim/bert-base-NER-uncased	8	8	0.001	300	0.0001
distilbert/distilbert-base-uncased	16	8	0.001	100	0.0001
Davlan/distilbert-base-multilingual-cased-ner-hrl	16	8	0.001	100	0.0001
prajjwal1/bert-medium	16	10	0.0001	100	0.0001
prajjwal1/bert-small	16	8	0.001	100	0.00007
prajjwal1/bert-mini	16	8	0.0001	100	0.0001
prajjwal1/bert-tiny	8	8	0.001	100	0.00007

Table 11: Hyperparameters of Encoder-based Transformer Models for Object and Aspect Labeling.

Model	train_batch_size	num_train_epochs	weight_decay	warmup_steps	learning_rate
microsoft/deberta-v3-base	16	7	0.01	400	0.00005
microsoft/deberta-v3-large	16	13	0.1	100	0.00003
FacebookAI/roberta-base	16	8	0.001	100	0.00007
FacebookAI/roberta-large	16	11	0.0001	300	0.00003
google-bert/bert-base-uncased	16	8	0.1	300	0.00007

Table 12: Hyperparameters of Encoder-based Transformer Models for Stance Classification.

Model	Precision	Recall	F1	Params
	Bondarenko et al. (2022a)			
rules	-	0.54	0.70	-
Aggregated	0.89	0.71	0.83	-
ALBERT-large	**0.95**	0.87	0.91	17M
lmsys/vicuna-7b-v1.5 (LoRA)	0.94	**0.92**	**0.93**	7B+33M

Table 13: Cross-validated results (10-folds) of the best model for Comparative Question Identification on the Webis-2022 dataset (Bondarenko et al., 2022a).

Model	F1-OBJ	F1-ASP	F1-PRED	F1-Mean	Params
		Bondarenko et al. (2022a)			
BiLSTM (paper)	0.82	0.52	0.85	-	-
RoBERTa-large (paper)	*0.93*	*0.80*	*0.98*	-	*355M*
RoBERTa-large (re-run from repo)	**0.8521±0.0252**	0.7024±0.0285	0.9668±0.0083	0.8689±0.0164	355M
microsoft/deberta-v3-base	0.8511±0.0170	0.7083±0.0356	**0.9700±0.0071**	0.8704±0.0118	184M
microsoft/deberta-v3-large	0.8502±0.0168	**0.7188±0.0257**	0.9683±0.0066	**0.8711±0.0079**	434M

Table 14: Cross-validated results (10-folds) of the best model for Object and Aspect Labeling on the Webis-2022 dataset (Bondarenko et al., 2022a). However, the results presented in (Bondarenko et al., 2022a) cannot be reproduced with the published code. Moreover, the evaluation methods are neither explicitly described in the paper, nor in the published code. This limits the transparency and reproducibility.

B. Error Analysis

The following Section comprises examples for each task. Comparative Question Identification most probably have errors in the original dataset. When considering Object and Aspect Labeling, Stance Classification, and Summarization tasks, we demonstrate and explain the most common error types.

B.1. Comparative Question Identification

Most questions that were misclassified by the best model, most probably have the incorrect labels in the initial dataset. Here are some examples:

- This is not a comparative question, it is asking for a scientific explanation

  "Is there an evolutionary advantage to having eyebrows?"

- This is not a comparative question but one looking for an explanation of a factual difference.

  "Why does turkey have darker, more flavorful dark meat than chicken?"

- The question is asking for a recommendation, rather than drawing a comparison.

  "Where can I get a very professional and reliable envelope printing service in Sydney?"

- The question is asking for a list of entities, but nothing is compared.

  "Who were the major colonial powers involved in Caribbean culture?"

- This is asking for a single most exported product, not for a comparison.

  "What was the primary export product of Eastern Europe to West?"

B.2. Object and Aspect Labeling

The following error types were identified:

1. The model (26%) often misclassified the non-entity token ('O') as an object ('B-OBJ' or 'I-OBJ') or aspect ('B-ASP' or 'I-ASP').
    - The word 'to' ('O') is incorrectly classified as 'I-OBJ':

      What is a good book to start learning about logic?
2. The model frequently (20%) confuses the following aspect entity ('I-ASP') with a following object entity ('I-OBJ') or a non-entity token ('O'). The other way around is true for the following object entity token ('I-OBJ') in 21% of cases.
    - The sequence "chess to a child" ('I-ASP') was misclassified as 'I-OBJ':

      What is the best way and right age to introduce chess to a child?
    - The sequence "self esteem and confidence" ('I-OBJ') was misclassified as 'I-ASP':

      Which is the best book for building self esteem and confidence?
    - The sequence "my porn videos" ('I-OBJ') was misclassified as 'I-ASP':

      What is the best site to sell my porn videos?
3. The model sometimes (8%) mistakes the beginning of an aspect entity ('B-ASP') with an object entity ('B-OBJ' or 'I-OBJ') or a non-entity token ('O').
    - The word "surreal" ('B-ASP') was misclassified as 'B-OBJ':

      What are some of the most surreal places in Germany?
4. The model struggles (6%) to correctly identify the beginning of an object entity ('B-OBJ') by predicting that it is a non-entity token ('O') or a following object entity token ('I-OBJ').
    - The word "be" ('B-OBJ') was misclassified as 'O':

      Is it better to be a famous rich person or an anonymous rich person?

B.3. Stance Classification

The analysis of the model's errors reveals four main types of misclassifications:

1. The correct label is "NONE" (neutral), but the model favors the first object:

   "Ferrari and Renault both have their strengths in the car industry"
    - The model misclassified the sentence favoring Ferrari (object 1), when in fact the sentence is neutral.
2. The correct label is "BETTER" (favoring the first object), but the model predicts "WORSE" (favoring the second object):

   "Microsoft has a larger market share than Sony"
    - The model incorrectly predicted that the sentence favors Sony, while it favors Microsoft.
3. The correct label is "WORSE" (favoring the second object), but the model predicts "NONE":

   "Toyota cars are not as luxurious as Ford cars."
    - The sentence was wrongly classified as neutral by the model, even though it favors Ford.
4. Actual label is "WORSE" (favoring the second object), but the model predicts "BETTER" (favoring the first object):

   "Ruby is not as efficient as Perl in text processing"
    - The model misclassified as favoring Ruby, while it favors Perl.

B.4. Summarization

When analyzing the output of the generated summaries, the following error types of the citations were taken into account:

1. The citation mentioned in the summary supports the opposite object than the one mentioned in the argument.

   Argument: *"Softball is much harder than baseball."*

   Summary excerpt: *"While some argue that baseball is harder than softball ..."*

2. The argument cited in the summary is irrelevant for the supported text part.

   Argument: *"Sony is slower than Microsoft."*

   Summary excerpt: *"They also have a better customer service reputation."*

3. The argument cited in the summary is irrelevant for the object comparison in general (the quality of the input argument was not good).

   Argument: *"It's nicer than soya, and will even make a decent hot chocolate."*

   Summary excerpt: *"While some people prefer chocolate over tea ..."*