Case-Based Adaptation of Argument Graphs with WordNet and Large Language Models

Mirko Lenz[1][0000−0002−7720−0436] and Ralph Bergmann[1,2][0000−0002−5515−7158]

[1] Trier University, Universitätsring 15, 54296 Trier, Germany
info@mirko-lenz.de, bergmann@uni-trier.de
[2] German Research Center for Artificial Intelligence (DFKI),
Branch Trier University, Behringstr. 21, 54296 Trier, Germany
ralph.bergmann@dfki.de

Abstract. Finding information online is hard, even more so once you get into the domain of argumentation. There have been developments around the specialized argumentation machines that incorporate structural features of arguments, but all current approaches share one pitfall: They operate on a corpora of limited sizes. Consequently, it may happen that a user searches for a rather general term like cost increases, but the machine is only able to serve arguments concerned with rent increases. We aim to bridge this gap by introducing approaches to generalize/specialize a found argument using a combination of WordNet and Large Language Models. The techniques are evaluated on a new benchmark dataset with diverse queries using our fully featured implementation. Both the dataset and the code are publicly available on GitHub.

Keywords: argumentation · graphs · adaptation · background knowledge · natural language processing

1 Introduction

Due to the sheer amount on information available on the internet, is has become increasingly harder for users to find exactly what they are looking for. At the same time, traditional search engines like Google purely operate on the textual layer, neglecting any potentially relevant structural information. These issues led to the development of specialized systems optimized for certain tasks—for instance, finding relevant arguments as part of so-called *argumentation machines* [28]. However, one fundamental flaw remains: If a user wants to retrieve information that is not in the corpus indexed by the search engine, it will not be able to provide relevant results.

To better understand this issue, consider a user wanting arguments relevant to the following query:

> Should we put a cap on cost increases of contracts when changing the payer?

Among others, the argumentation machine retrieved the following result from the microtexts corpus [25]:

> Rent prices are already regulated in favour of tenants due to existing laws and the rent index. In view of the high prices for buying flats with existing rent contracts, these are an unattractive investment.

This argument is already quite relevant, but while the query asked about "contract costs", the result is concerned with "rent costs". The relevance would be even higher if the machine was able to infer that "contract costs" is a generalized form of "rent costs" and automatically *adapts* the argument before presenting it to the user. Whereas the pure retrieval of arguments has been solved by multiple works [5, 20, 15, 34, 31], adaptation is rather difficult to solve and—to the best of our knowledge—has not yet been tackled in the literature.

Consequently, our work pursues the following research question: "Given a user-defined query together with arguments retrieved from a larger corpus, can we generalize or specialize the results to better match the user's query and provide a more relevant and useful set of results?" We tackle the *reuse* step that is performed after the *retrieval* in the context of a larger Case-Based Reasoning (CBR) [1] system. The main contributions of this paper are as follows: (i) An approach that extracts the most important keywords of an argument and adapts them using WordNet [22, 17], (ii) multiple approaches leveraging state-of-the-art Large Language Models (LLMs), (iii) a hybrid approach where the identified keywords are adapted using a LLMs and validated with WordNet, (iv) a new benchmark dataset with diverse queries (including reference rankings for a retrieval system and possible adaptation results), and (v) a publicly available implementation[3] that powers an experimental evaluation assessing the impact of the proposed adaptation techniques on a retrieval system.

The presented techniques are in principle applicable to arbitrary types of texts. Some features of our heuristics—for instance, the identification of relevant keywords—use an argument's internal structure to make better informed decisions. We use an established graph-based representation—that is, *argument graphs* (see Section 2 for details)—with a large number of corpora available online. A major part of our efforts will be concerned with *explainability*: Since we alter the semantics of an argument, we argue that the user must have the ability to review any automatic change—otherwise the user's trust may be damaged.

To the best of our knowledge, this is the first paper to apply LLMs to the reuse phase of a CBR system. These models are particularly good at predicting the next word given a specific context which is exactly the kind of task we are trying to solve. We hope to pave the way for further developments in this area—especially Textual Case-Based Reasoning (TCBR) [35].

The remainder of this paper is structured as follows: Section 2 will introduce the most important foundations, followed by a review of related work in Section 3. In Section 4, we describe the three different adaptation approaches. Section 5 provides an experimental results and a discussion of our proposed approaches. Lastly, Section 6 concludes our findings.

[3] https://github.com/recap-utr/arguegen

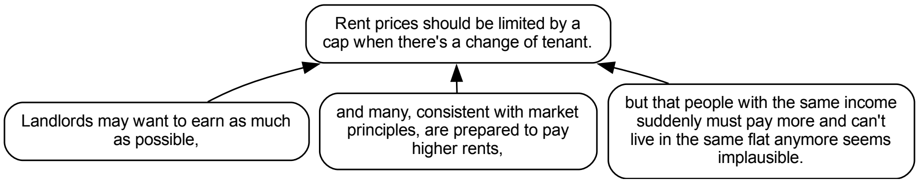

Rent prices should be limited by a cap when there's a change of tenant.
Landlords may want to earn as much as possible,
and many, consistent with market principles, are prepared to pay higher rents,
but that people with the same income suddenly must pay more and can't live in the same flat anymore seems implausible.

Fig. 1: Exemplary argument graph from the microtexts corpus

2 Foundations

In this section, we briefly discuss the foundational concepts and techniques underpinning our proposed approach for argument generalization. To solve this rather difficult task, we combine multiple fields like Computational Argumentation (CA), CBR, and Natural Language Processing (NLP). We will start by introducing the type of data we are using: arguments.

2.1 Argumentation Theory

An *argument* consists of one *claim* together with one or several *premises* that are linked to the claim [24]. The debatable claim can be *attacked* or *supported* by its connected premises [32]. Claims and premises are the smallest self-contained units of argumentation and are also called Argumentative Discourse Units (ADUs). Using these building blocks it becomes possible to construct *argument graphs* to represent larger discourses. Such graphs make it possible for us to integrate structural information into the adaptation process. Consider the example shown in Fig. 1: The blue nodes represent the ADUs and as such store the argumentative content, whereas the arrows represent relationships between them. Most such graph have one *major claim* that defines the overall conclusion of an argument—in our example, this is the root node.

2.2 Automated Reasoning

As mentioned earlier, we follow the overall methodology of CBR to tackle the generalization of argument graphs. The basic assumption here is that similar problems (or *cases*) have similar solutions. By storing previous problems together with their solutions in a *case base*, it is possible to solve new problems with existing knowledge. CBR is a mere problem-solving methodology, meaning that it can be combined with various techniques—for instance, Machine Learning (ML). The branch of Textual Case-Based Reasoning (TCBR) [35] is concerned with cases stored as texts and as such utilizes NLP techniques. Čyras et al. propose Abstract Argumentation for CBR (AA-CBR) [12] and thus combine the Dung framework [16] with CBR. In this paper, we will utilize another reasoning approach—Analogy-Based Reasoning (ABR) [10, 13]—to tackle the *reuse* step of CBR. ABR is based on the assumption that a is to b what c is to d (i.e.,

$a : b :: c : d$) [18]. Thus, by finding the triple (a, b, c), we are able to infer the missing value d [27].

2.3 Background Knowledge

In order to draw sensible inferences with ABR, we need some sort of background knowledge. Due to our focus on generalization/specialization, the lexical database WordNet is a fitting candidate. At its core, WordNet is composed of *lemmas* that are grouped together to so-called *synsets* when multiple lemmas share the same meaning (e.g., the lemmas "price" and "cost" could be grouped this way). Basically, a synset is an n-gram and thus can include compound words. In the following, we will also use the term "concept" when referring to a synset. These synsets are linked via six relationship types, out of which we will only consider *hypernyms* (i.e., generalizations) and *hyponyms* (i.e., specializations). A crucial component of all synsets and lemmas in WordNet is the accompanying Part of Speech (POS) tag, allowing to differentiate between the activity "shop" and the business "shop" [26]. For each synset, WordNet provides one definition along with multiple exemplary real-world uses of the underlying lemmas that may be used as additional contextual information for NLP operations like computing semantic similarities.

2.4 Natural Language Processing

Providing an introduction to all aspects of NLP is out of scope for this paper, we will instead focus on the advanced concept of LLMs here. They use the transformer architecture [33] which also powers models like Bidirectional Encoder Representations from Transformers (BERT) [14]. Compared to plain word embeddings like word2vec [21], transformers use an internal concept of *attention* that allows them to produce contextualized embeddings (i.e., the same word may have different vectors depending on its context). They may be *fine-tuned* on a new dataset, making it possible to apply them on specialized tasks—for instance, Sentence-Transformers (STRF) [29] have been fine-tuned to compute semantic similarities between sentences. Generative Pre-trained Transformer (GPT) models (a type of LLM) use a vastly higher number of internal parameters and are trained on larger corpora, enabling them to show state-of-the-art performance on a variety of tasks even without a fine-tuning step. Instead, they make use of *few-shot learning* (i.e., providing some examples with expected output) and *prompting* to instruct the model [8].

3 Related Work

In our literature research, we did not find prior works on generalizing or specializing argument graphs in a CBR context. Thus, the upcoming section will highlight a selection of contributions to (i) the case-based retrieval of argument graphs and (ii) the adaptation of arguments.

The retrieval of arguments has been covered by many works in recent past—for instance, by the search engines Args.me [34] and ArgumenText [31]. These approaches however deal with simple argumentation structures—that is, they only consider individual ADUs and their stances (pro/con), not complete argument graphs. Bergmann and Lenz investigate the use of CBR methods for retrieving argument graphs based on a structural mapping between user queries and the stored cases [5, 20]. For each node in the user query, a matching node in the case is determined by comparing the embeddings of the node's content. These local similarities are then aggregated to form a global measure incorporating both structural and semantic aspects.

Regarding the adaptation procedure itself, there are two perspectives on this task: (i) Change the structure of the graph or (ii) modify the textual content of the ADUs. A major issue for the former are so-called co-references—for instance, the word "he" might refer to a person described in another node. The latter perspective can be tackled using TCBR—which is concerned with textual adaptation and has its roots in the field of legal reasoning [2, 30]. Bilu and Slonim propose a method to recycle claims for the use in a new domain with the help of Statistical Natural Language Generation (SNLG) [6]. CBR and ABR have been investigated in the area of mediation [3]. The underlying commonsense knowledge has also been applied to the area of argumentation in the past via manual annotation [4]

4 Case-Based Adaptation of Argument Graphs

Having provided the necessary foundations together with relevant works in the field, we will now present our proposed approach for adapting arguments. We will focus solely on *reusing* arguments, leaving the retrieval to the system introduced in [5] (see Section 3 for details). A total of 6 techniques divided in three broader categories will be introduced in this section: (i) two WordNet-based variants, (ii) three LLM-based ones, and (iii) one hybrid one. In addition to the reference implementation in Python used in Section 5, we present a high-level overview of all algorithms using flowcharts with exemplary content. Consistent with the research question (see Section 1), our overall goal is defined as follows: Show that it is possible to increase the relevance of a ranking produced by a retrieval system w.r.t. a given query by generalizing/adapting the found cases.

We consider both *non-interactive* (i.e., the adaptation happens automatically) and *interactive* (i.e., the user initiated the adaptation process) scenarios. In the former, the adaptation is performed automatically after the retrieval without any sort of interaction from the user. In the latter, a user initiates the adaptation process manually for a single case. To tailor the results to their needs, the user provides so-called *adaptation rules* (see Section 4.1) to the system that serve as a starting point for the process. All of our approaches therefore have the ability to honor certain wishes w.r.t. the adaptation.

4.1 Case Representation

Before proceedings with the algorithms, we will briefly establish some common notation: Each case $g \in \text{CB}$ of our case base CB is an argument graph and as such is composed of a set of ADUs $\text{adus}(g) \coloneqq \{a_1, \ldots, a_n\}$. Concepts (i.e., the keywords of the graph's ADUs) $c_i \in C$ are mapped to a set of synsets $\{s_1, \ldots, s_n\} \in S$ with S being all nodes of WordNet. Each such concept c is an n-gram (typically $n \leq 3$) together with a POS tag. For each such concept c, we also store its POS tag. The user-provided query q is an argument graph (just like the stored cases), whereas the rules $\{(c_1, c_2), \ldots\} \coloneqq R$ are source-target tuples of concepts that need to be replaced. The function $\text{vec}(x)$ denotes the vector/embedding of an arbitrary text x.

Based on this representation, we define the function $\text{score}(c)$ to assess the "relevance" of a concept c when (i) filtering relevant concepts to be extracted, (ii) comparing multiple adaptation candidates of a concept, and (iii) determining a sensible order when applying the adaptations. The score is an aggregation of multiple metrics that make use of so-called *related concepts* in the spirit of ABR ($a : b :: c : d$). Let us give you a concrete example: We found out that the concept "landlord" (c) should be adapted know that the general topic of an argument shall be generalized from "rent" (a) to "cost" (b). We now try to find a generalization d of "landlord" that has a high similarity to both "cost" and "landlord". In our paper, such a *similarity function* shall produce a value in $[0, 1]$ when given two concepts c_1, c_2—for instance (i) the semantic similarity between the examples of connected synsets, (ii) the path-based distance of these synsets within WordNet, and (iii) the semantic similarity of the original ADUs. For a complete list, we refer the reader to our reference implementation. A concept's global score is finally defined as the arithmetic mean of all these local measures.

4.2 Adaptation with WordNet

With the case representation introduced, we will now proceed with presenting the first adaptation approach that is built on WordNet. In essence, we extract the most important keywords of an argument and follow their hypernym/hyponym relations to derive appropriate generalizations/specializations. This approach is explainable by default since the reasoning chain is completely known (something which is not the case for all other techniques that we propose). The procedure is depicted in Fig. 2 and discussed in the following paragraphs.

Generate Rules In order to come up with adaptations for individual concepts, we first need to determine how the overall topic of the argument should be changed. In the context of ABR ($a : b :: c : d$) that would mean finding the variables a and b. We generate them by extracting the keywords of the user's query and the argument's major claim with an established extraction algorithm like YAKE [9]. We then determine the shortest paths between the major claim's and the query's keywords and use those pairs having the smallest distance as our adaptation rules. For our previously used example, the resulting rule could

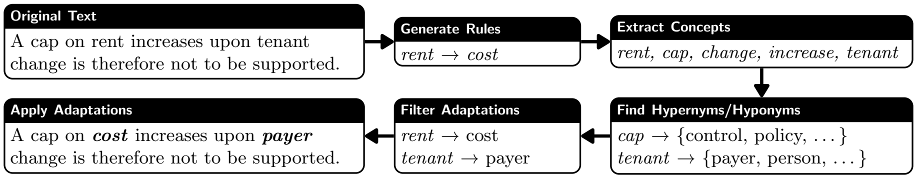

Original Text
A cap on rent increases upon tenant change is therefore not to be supported.
Generate Rules
rent → *cost*
Extract Concepts
rent, cap, change, increase, tenant
Find Hypernyms/Hyponyms
cap → {control, policy, ...}
tenant → {payer, person, ...}
Filter Adaptations
rent → cost
tenant → payer
Apply Adaptations
A cap on ***cost*** increases upon ***payer*** change is therefore not to be supported.

Fig. 2: Overview of the WordNet-based adaptation pipeline with exemplary values.

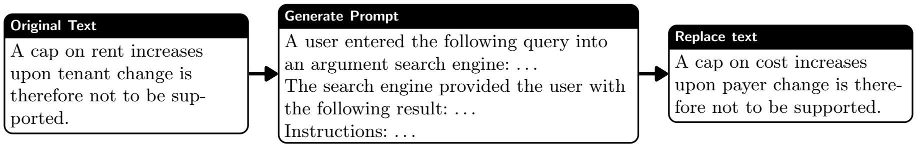
be "rent → cost". Note that we do not need our concept score in this stage. In case the adaptation is performed interactively (i.e., the user provided rules), the system skips this step.

Extract Concepts Now that we know the left part of the ABR equation, we can move to the right part, starting with variable c. Since we determined both a and b through a keyword extraction algorithm like YAKE, we will do the same here to receive a set of *concept candidates*. When for example presented with the rule "rent → cost", one such candidate could be "landlord". We then compute the score of each candidate, enabling us to define a *threshold* that each candidate has to reach. The remaining ones are ordered by their score and optionally selected through a *cut-off*—that is, only the best x candidates are used. After this step, we have a list of *extracted concepts* which can be generalized.

Find Hypernyms/Hyponyms The only variable missing in the ABR equation is d—that is, the generalized/specialized concept. We will first be concerned with finding potential *adaptation candidates* before filtering these based on their computed concept scores. Two variants for fetching those candidates are proposed: (i) Directly use the synsets connected to c (*taxonomy-based*) or (ii) replicate the WordNet paths between a, b with c as the new start node (*analogy-based*). The taxonomy-based one is faster to compute, but has the drawback of needing to compute the score for more concepts (since they are not filtered based on the paths between a and b). Another difference is that the analogy-based variant is much more strict on the potential adaptation targets.

Filter Adaptations We now have multiple candidates for the variable d and are left with selecting the best one. The best synset is chosen based on the corresponding concept score, leaving the task of selecting the correct lemma for the given context. Again, we make use of ABR and select the lemma where the word embedding difference $\text{vec}(c) - \text{vec}(d)$ is closest to $\text{vec}(a) - \text{vec}(b)$.

Apply Adaptations Now that all variables of the equation $a : b :: c : d$ are known, we can finally insert them into the Argument Graph (AG). In an effort to minimize the risk of applying "harmful" adaptations to the argument, we propose an *iterative* technique: Instead of applying all identified adaptations at once, we insert them one after each other and compute the similarity of the case to the



A cap on rent increases upon tenant change is therefore not to be supported.

A user entered the following query into an argument search engine: ...
The search engine provided the user with the following result: ...
Instructions: ...

A cap on cost increases upon payer change is therefore not to be supported.

Fig. 3: Overview of the LLM-based adaptation pipeline with exemplary values.

query after each operation. As soon as the similarity score is identical or even reduced, we stop. The adapted lemmas are correctly inflected to minimize grammatical errors. To minimize the runtime impact, we only consider the semantic similarity of the complete argument and skip the expensive structural matching. This optimization is consistent with the findings of Bergmann and Lenz [5, 20] who report that the semantic retrieval alone produced almost the same ranking.

4.3 Adaptation with Large Language Models

We were mostly concerned with heuristics in the past section. Moving to LLMs, the focus now shifts towards *prompt engineering* as introduced in Section 2. These prompts have different requirements for different LLMs, so we will discuss two paradigms here: (i) edit-based models and (ii) chat-based models. The former ones are a perfect fit for the use-case of adapting arguments since they are specialized in editing texts based on an instruction. The latter family of models has seen an increasing interest by researchers lately and are equally relevant for our work as they are aware of previous responses—an ability that could be valuable for our task. Please note that due to space constraints, we are unable to present the complete prompts and instead show an excerpt of our prompt template in Fig. 3. We refer the interested reader to our implementation for more details.[4]

Edit-Based Models The approach here is relatively simple: As an instruction, we present the model the query together with the case. For each ADU, the task is then to make it more relevant to the query by generalizing the presented snippet.

Chat-Based Models These types of models allow greater degree of freedom. To account for that, we propose two different ways of approaching the adaptation of arguments with chat completions. We either try to (i) replicate the edit-based model and let the chat-based LLM rewrite an ADU's content or (ii) replicate the WordNet approach and let the model predict an adapted text together with the accompanying replacement rules. In other words, the LLM may rewrite the whole text in (i) whereas it should only substitute certain keywords in (ii). This is also why (i) is not explainable whereas (ii) is to a certain degree: There is no guarantee that the predicted texts and rules are consistent with each other, potentially causing user confusion if presented as is.

[4] https://github.com/recap-utr/arguegen

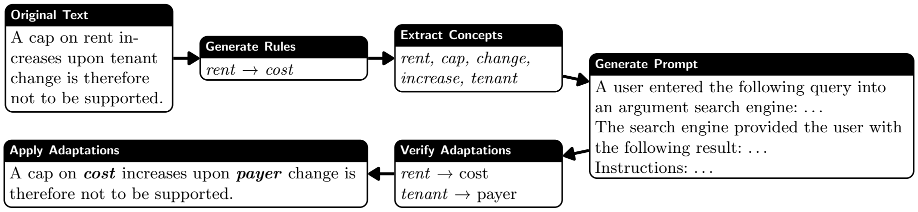

Original Text
A cap on rent increases upon tenant change is therefore not to be supported.
Generate Rules
rent → cost
Extract Concepts
rent, cap, change, increase, tenant
Generate Prompt
A user entered the following query into an argument search engine: ...
The search engine provided the user with the following result: ...
Instructions: ...
Verify Adaptations
rent → cost
tenant → payer
Apply Adaptations
A cap on cost increases upon payer change is therefore not to be supported.

Fig. 4: Overview of the hybrid adaptation pipeline with exemplary values.

For both techniques, we again present the query together with the retrieved argument as context to the model. Only for (ii) we add an instruction that edits should be limited and a list of adaptation rules has to be provided. We then iteratively let the model predict an output for each ADU and add those responses to the context presented to the LLM.

4.4 Hybrid Adaptation

While LLMs often generate text that on first glance seems correct, the results are not guaranteed to be valid in the real world or even consistent with previous responses within the same "conversation". We consequently propose a hybrid adaptation approach that tries to combine the best of both worlds: The word prediction capabilities of LLMs and the consistency checks possible with the taxonomy modeled in WordNet. The process is depicted in Fig. 4.

Compared to the WordNet-based approach seen in Fig. 2, the adaptation and filtering steps are replaced by prompt generation and validation steps. The remaining ones—that is, rule generation, keyword extraction, and application of adaptations—are used unchanged, so we refer to Section 4.2 for details. The new prompt is similar to the one used in Section 4.3: The LLM yet again receives the query and case as context, but this time we do not preset a single ADU to the model, but instead the output of our concept extraction step. In principle, we now have the same output as with the WordNet pipeline, but the predicted adaptations are not guaranteed to be correct. We argue that this attribute is a central aspect when dealing with arguments, so we add an additional *verification* step: For each adaptation rule, we check if it (i) is present in WordNet and (ii) does not exceed a (configurable) path distance threshold from the original concept. Finally, use the *iterative* replacement technique introduced in Section 4.2 to apply the generated adaptation rules. A side-effect of this technique is that it needs fewer predictions by the LLM (one per case here versus one per ADU for Section 4.3) and thus reduces the costs associated with the whole procedure.

5 Experimental Evaluation

The following section provides an experimental evaluation of our proposed approaches for the adaptation of argument graphs. We will define the hypotheses guiding our evaluation, present the experimental setup along with the metrics used, and provide a detailed analysis of the results. Since dealing with arguments often entails a subjective aspect, our evaluation will not solely rely on numbers and supplement the experiments with a detailed review of an exemplary adapted argument. Consequently, our quantitative evaluation in Section 5.2 will be accompanied by a case study in Section 5.3.

Let us start by introducing our working hypotheses that all contribute to answer our research question formulated in Section 1: "Does at least one adaptation approach for argument graphs lead to more relevant/useful results—and if so, which one?"

H1. The generated adaptation rules are a decent approximation the ones crafted by experts.

H2. The similarity of an adapted case w.r.t. to the query is higher than that of the original retrieved case.

H3. By combining the taxonomy of WordNet with the prediction capabilities of a LLM, the hybrid approach performs best.

H1 and H2 aim at making the rather vague notion of "better results" measurable and are complemented by H3 that checks whether the hybrid technique produces the best results w.r.t. those hypotheses. It is worth noting that in our evaluation, we focus solely on the *generalization* aspect of our proposed approach for argument retrieval. While specialization is an equally important task, this focus allows us to discuss the different techniques in more detail. Specialization can be seen as the opposite of generalization, meaning that these two directions may be swapped rather easily.

5.1 Experimental Setup

We wrote a fully working application in Python to enable running our experiments. In an effort to embrace reproducibility, both the software and our dataset (see below) are freely available under the permissive MIT license.[5] Our concept score relies partly on semantic similarity measures (i.e., embeddings). We run the experiments with plain fastText (FT) [7] as well as the contextualized Universal Sentence Encoder (USE) [11] and STRF. For the LLMs, we use the GPT family of models developed by OpenAI that have been popularized through ChatGPT.[6]

Corpus As a data source containing AGs, we use the well-known microtexts corpus, containing a total of 110 graphs with a mean number of five ADUs per graph. It is composed of 23 distinct topics with relatively similar cases. Out of these, we selected nine topics based on their number of cases, leading to 62

[5] https://github.com/recap-utr/arguegen

[6] We used the following models: FT: `en_core_web_lg` from spaCy, USE: `v4`, STRF: `multi-qa-MiniLM-L6-cos-v1`, edit-based LLM: `text-davinci-edit-001`, chat-based LLM: `gpt-3.5-turbo`

graphs. Each of these has been annotated with a *query* for a retrieval system as well as a list of *benchmark adaptations* independently by two experts, resulting in a total number cases of 124. The experts already had experience with AGs and were given detailed annotation guidelines. Both of them received the same cases—making it possible to determine the Inter-Annotator Agreement (IAA)—and were told not to talk about the task with each other. They were also told to order the generalizations based on their importance s.t. the adaptation having priority is at the top. For example, when presented with the argument shown in Fig. 1, one expert created the query "Why should we not put a cap on cost of contracts when changing contractual payer?" Based on the case and the query, they created the three generalizations (i) *rent* → *cost*, (ii) *tenant* → *payer*, and (iii) *flats* → *objects*.

Annotation Reliability We will now determine the IAA between the annotations of the two experts to assess the reliability of the gathered data. Each expert assigned multiple rules to a case, thus we use Krippendorff's α [19] together with Measuring Agreement on Set-Valued Items (MASI) [23] as a weighting method for sets of values. Each rule is a tuple (source, target), meaning that we can differentiate between two *perspectives* of the IAA: (i) Treat identical sources as a perfect agreement (thus ignoring the specified target) or (ii) only treat rules as perfect agreement where both source and target match. Both perspectives yield quite poor agreement scores of (i) 0.20 and (ii) 0.03. Krippendorff recommends to discard all values < 0.667 and as such tells us to not rely on the gathered data. Even if we only investigate the first rule (i.e., the one deemed to be the most important one by the experts), we only receive scores of (i) 0.44 and (ii) 0.08. The main conclusion we can draw from these annotations is the fact that argumentation in itself is highly subjective. As such, a generalization of the system might be sensible even though it does not perfectly resemble the ground truth—affirming the need for a case study.

5.2 Quantitative Results

With the setup explained, we may now proceed and conduct experiments to assess our hypotheses. We will use the standard Information Retrieval (IR) measures Precision P, recall R, and the ranking-aware nDCG. We have already seen in the last section that the IAA of correct sources *and* targets is almost zeros. Consequently, these IR metrics are computed by comparing the *sources* of the system and expert rules—consequently treating the latter as our ground truth. We will additionally report the metrics precision $P@1$ and recall $R@1$ for the first rule as well. Lastly, we measure the similarity improvement $\text{sim}_{\nearrow}$ which is defined as the similarity of the adapted case to the query divided by the similarity of the original case to the query—that is, $\text{sim}_{\text{adapt}} \, / \, \text{sim}_{\text{orig}}$—and indicates whether we succeeded in making the case more relevant to the user's query.

Before presenting our results, we need to discuss the edit-based LLM approaches. As mentioned in Section 4.3, these techniques directly alter the text and provide no trace of the changes which may be problematic for the domain of argumentation where factual correctness is crucial. As part of our experiments, we noticed that most ADU texts are changed completely and at the same time tend to be longer than their unmodified counterparts. We also faced issues when parsing the LLM responses—for instance, texts like the following were predicted: "This segment is not directly relevant to the presented query, but it could be adapted by changing ..." Due to these inconsistencies and the fact that we cannot apply our metrics to them, we need to skip the evaluation of both edit-based models.

We will now proceed to discuss the findings based on the results depicted in Table 1. Among all methods, the chat-based LLM performs worst and even is the only one showing a decrease in semantic similarity. The WordNet approaches show different results depending on the underlying embedding model: FT already predicts a high similarity value between the case and the unmodified case, thus the improvement only changes slightly. It also shows that these plain embeddings are not as well suited as a heuristic to determine adaptation rules—all metrics are lower compared to the contextualized USE or STRF. The analogy-based heuristic delivers worse results than the taxonomy-based on across the board. Moving to the hybrid approach, you may have noticed that we tested two validation setups: *lenient* and *strict*. The lenient one accepts all adaptation rules that are part of WordNet, whereas the strict one only allows concepts in close vicinity to be used as rules. With lenient validation, we observe a higher recall paired with a lower precision. Strict validation leads to the exact opposite situation, meaning that both models are viable depending on the user's preference. The results w.r.t. the most suitable embedding models correlate with our findings of the WordNet approach: contextualized ones yield better metrics.

When analyzing the runtimes of the methods we get mixed results: On average, the WordNet techniques need 5s with FT, 20s with USE and 30s with STRF per case. For the LLM based approaches, measuring the time is rather challenging due to rate limits imposed by OpenAI: once such a limit is reached, an exponential backoff has to be applied. With that in mind, we observed typical runtimes of 50s for the chat-based LLM and 25s for the hybrid technique. While the former performs one request for each ADU of an argument, the latter only requires one request per case, reducing the impact of the rate limits. Overall, the processing time is not optimal for interactive use—only WordNet wth FT could respond within a few seconds.

We will finish the quantitative evaluation by assessing our three hypotheses. To fully accept H1, our models would need to produce perfect precision/recall scores, which is not the case here. At the same time, we have seen that even two human annotators have no strong agreement, making it a difficult decision. Given the fact that the first adaptation rule is correct almost 80% of the time (see $P@1$) for the best performing models, we tend to cautiously accept H1. The situation is easier w.r.t. H2: Both WordNet and the hybrid technique yield more than 20% increase of the similarity score, leading to an acceptance of H2. Now we are only left with H3—is the hybrid approach the best one? The lenient variant has the best scores for $R, \text{nDCG}, \text{sim}_{\nearrow}$, whereas its strict counterpart boasts the

Table 1: Evaluation results the concept-based adaptation approaches

Approach	P	R	P@1	R@1	nDCG	sim$_{\nearrow}$
WordNet (Analogy, FT)	.283	.433	.598	.155	.377	+1.03%
WordNet (Taxonomy, FT)	.304	.448	.620	.165	.400	+1.07%
WordNet (Analogy, USE)	.515	.412	.725	.193	.413	+16.3%
WordNet (Taxonomy, USE)	.517	.426	.717	.191	.420	+16.8%
WordNet (Analogy, STRF)	.519	.473	.770	.207	.450	+20.7%
WordNet (Taxonomy, STRF)	.525	.474	.777	.211	.450	+21.3%
LLM (Chat-based)	.049	.226	.050	.010	.088	−1.64%
Hybrid (Strict, FT)	.501	.357	.644	.170	.363	+0.54%
Hybrid (Lenient, FT)	.304	.437	.664	.178	.410	+0.86%
Hybrid (Strict, USE)	.587	.442	.714	.190	.432	+15.5%
Hybrid (Lenient, USE)	.361	**.544**	.738	.197	**.494**	+**23.5**%
Hybrid (Strict, STRF)	**.590**	.452	**.798**	**.218**	.445	+18.3%
Hybrid (Lenient, STRF)	.353	.528	.738	.200	.483	+**23.5**%

highest values for P, meaning that we tend to accept H3. This is also confirmed by our experience when running the experiments: The hybrid approach seems to be quite robust and in most cases comes up with adaptations that are sensible. The same is not true for the other approaches—they are more likely to predict generalizations that make it harder to understand the argument afterwards.

5.3 Case Study

We have now discussed how the approaches perform w.r.t. the evaluation metrics. To complement these numbers, we will in the following examine concrete adaptation outcomes based on the argument graph depicted in Fig. 1. An expert has created a query together with the corresponding adaptation rules—in fact, this example is part of our benchmark dataset used in the previous section. Please note that due to space constraints, we are unable to show full argument graphs and instead decided to present the beginning of each argument in plain text instead. The concepts changed from the original text are marked in italics.

Query: Should there be a cap on annuity increases for a change of remunerators?

Expert: *Annuity* prices should be limited by a cap when there's a change of *renumerator*. *Property owners* may want to earn as much as possible, and many, consistent with market principles, are prepared to pay higher *annuities*, ...

WordNet (Taxonomy, STRF): Rent prices should be limited by a cap when there's a change of *remunerator*. Landlords may want to earn as much as possible, and many, consistent with market principles, are prepared to pay higher rents, ...

LLM (chat-based): *Annuity increases* should be limited by a cap when there's a change of *remunerators*. *Annuity providers* may *be prepared to offer higher increases* consistent with market principles, ... *Setting a cap on annuity increases in case of a change of remunerators is necessary since sudden changes in income and unaffordable annuity increases are unacceptable.*

Hybrid (Lenient, STRF): Rent costs should be limited by a cap when there's a change of *remunerator*. Landlords may want to earn as much as possible, and many, consistent with *economy* principles, are prepared to pay higher rents, ...

The conclusions one can draw from this example correspond to those of the previous section. The LLM reformulates large parts of the text and even adds new content at the end—an undesired behavior for our use case. The case adapted through WordNet only has one, but correct, adaptation: "tenant → remunerator". In addition to this one, the hybrid approach adapted "market → economy" which is equally sensible even though it is not part of the expert rules. When inspecting the generation process of the hybrid approach, we observed that many more rules were generated—for instance "rent → annuity". These were however not applied since they did not increase the similarity of the case to the query as much as other rule combinations. As such, it may not be the prime goal of an adaptation approach to maximize the similarity through generalization—an aspect that should be further investigated as part of future work.

6 Conclusion and Future Work

In this paper, we successfully designed and implemented an approach for generalizing argument graphs in the context of CBR. With the help of a new benchmark corpus, we demonstrated that the similarity between a retrieved case and a user's query can be increased by more than 20%. This is made possible by combining the taxonomic information obtained from WordNet with the generative powers of recent LLMs. Our tested approaches provide an easy-to-follow trace which changes have been made to an argument which is crucial to gain a user's trust. Revisiting the initial research question—can we increase the relevance of retrieved arguments through adaptation—it is now possible to answer it with a *cautious yes:* Despite needing more working in the future, our results are promising and show the potential even at this early stage. Given the low IAA of our benchmark corpus, using expert rules as a ground truth for this task should be questioned due to the inherent subjectivity involved in the annotation process.

In future work, one may use even more powerful LLMs to perform this task. Additionally, it may be worthwhile to fine-tune a LLM specific to generalizing/specializing argument graphs, the main obstacle here is the lack of adequate training data. Another useful aspect could be the introduction of a structural component to the adaptation—for instance, to remove nodes that are no longer relevant. Lastly, the runtime should be improved to enable the use of our approach in interactive scenarios.

Acknowledgements This work has been funded by the DFG within the project *ReCAP-II* (No. 375342983) as part of the priority program RATIO (SPP-1999) as well as the *Studienstiftung*.

References

1. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Communications **7**(1), 39–59 (1994)
2. Ashley, K.D.: Modelling legal argument: reasoning with cases and hypotheticals. Ph.D. thesis, University of Massachusetts, USA (1988)
3. Baydin, A.G., López de Mántaras, R., Simoff, S., Sierra, C.: CBR with Commonsense Reasoning and Structure Mapping: An Application to Mediation. In: CBR Research and Development. pp. 378–392. Springer (2011)
4. Becker, M., Staniek, M., Nastase, V., Frank, A.: Enriching Argumentative Texts with Implicit Knowledge. In: Natural Language Processing and Information Systems. pp. 84–96. LNCS, Springer (2017)
5. Bergmann, R., Lenz, M., Ollinger, S., Pfister, M.: Similarity Measures for Case-Based Retrieval of Natural Language Argument Graphs in Argumentation Machines. In: The Thirty-Second International Flairs Conference. pp. 329–334. AAAI Press, Florida, USA (19 May 2019)
6. Bilu, Y., Slonim, N.: Claim Synthesis via Predicate Recycling. In: 54th Annual Meeting of the ACL. pp. 525–530. Berlin, Germany (Aug 2016)
7. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching Word Vectors with Subword Information. arXiv (15 Jul 2016)
8. Brown, T.B., et al.: Language Models are Few-Shot Learners. arXiv (28 May 2020)
9. Campos, R., Mangaravite, V., Pasquali, A., Jorge, A.M., Nunes, C., Jatowt, A.: YAKE! Collection-Independent Automatic Keyword Extractor. In: Advances in Information Retrieval. pp. 806–810. Springer (2018)
10. Carbonell, J.G.: Derivational analogy and its role in problem solving. In: Proceedings of the Third AAAI Conference on Artificial Intelligence. p. 64–69. AAAI'83, AAAI Press, Washington, D.C. (22 Aug 1983)
11. Cer, D., Yang, Y., Kong, S.Y., Hua, N., Limtiaco, N., St. John, R., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Sung, Y.H., Strope, B., Kurzweil, R.: Universal Sentence Encoder. arXiv (29 Mar 2018)
12. Cyras, K., Satoh, K., Toni, F.: Abstract argumentation for case-based reasoning. In: Baral, C., Delgrande, J.P., Wolter, F. (eds.) Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference, KR 2016, Cape Town, South Africa, April 25-29, 2016. pp. 549–552. AAAI Press (2016), http://www.aaai.org/ocs/index.php/KR/KR16/paper/view/12879
13. Defourneaux, G., Peltier, N.: Analogy and Abduction in Automated Deduction. In: Proc. IJCAI-97. p. 216–221. Morgan Kaufmann (1997)
14. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv (11 Oct 2018)
15. Dumani, L.: Good Premises Retrieval via a Two-Stage Argument Retrieval Model. undefined (2019)
16. Dung, P.M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artificial intelligence **77**(2), 321–357 (1 Sep 1995)
17. Fellbaum, C. (ed.): Wordnet: An Electronic Lexical Database. Language, Speech and Communication, The MIT Press, Cambridge, MA (9 May 1998)
18. Hesse, M.: On Defining Analogy. Proc. Aristotelian Soc. **60**, 79–100 (1959)
19. Krippendorff, K.: Reliability in Content Analysis: Some Common Misconceptions and Recommendations. Human communication research **30**(3), 411–433 (1 Jul 2004)

20. Lenz, M., Ollinger, S., Sahitaj, P., Bergmann, R.: Semantic Textual Similarity Measures for Case-Based Retrieval of Argument Graphs. In: CBR Research and Development. pp. 219–234. LNCS, Springer (2019)
21. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient Estimation of Word Representations in Vector Space. arXiv (16 Jan 2013)
22. Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.: WordNet: An online lexical database. In: International Journal of Lexicography (1990)
23. Passonneau, R.: Measuring Agreement on Set-valued Items (MASI) for Semantic and Pragmatic Annotation. In: Proceedings of LREC. ELRA, Genoa, Italy (May 2006)
24. Peldszus, A., Stede, M.: From Argument Diagrams to Argumentation Mining in Texts: A Survey. IJCINI **7**(1), 1–31 (2013)
25. Peldszus, A., Stede, M.: An annotated corpus of argumentative microtexts. In: Proceedings of the 1st European Conference on Argumentation. vol. 2, pp. 801–815. College Publications, Lisbon, Portugal (2015)
26. Petrov, S., Das, D., McDonald, R.: A Universal Part-of-Speech Tagset. In: Proceedings of LREC. pp. 2089–2096. ELRA, Istanbul, Turkey (May 2012)
27. Prade, H., Richard, G.: Analogical Proportions and Analogical Reasoning - An Introduction. In: CBR Research and Development. pp. 16–32. LNCS, Springer International Publishing (2017)
28. Reed, C., Norman, T.J.: Argumentation Machines: New Frontiers in Argument and Computation, Argumentation Library, vol. 9. Springer Science & Business Media, Dordrecht (9 Mar 2013)
29. Reimers, N., Gurevych, I.: Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In: Proceedings of EMNLP-IJCNLP. pp. 3982–3992. ACL, Hong Kong, China (Nov 2019)
30. Rissland, E.L., Ashley, K.D., Karl Branting, L.: Case-based reasoning and law. Knowledge Engineering Review **20**(3), 293–298 (Sep 2005)
31. Stab, C., Daxenberger, J., Stahlhut, C., Miller, T., Schiller, B., Tauchmann, C., Eger, S., Gurevych, I.: ArgumenText: Searching for Arguments in Heterogeneous Sources. In: Proceedings of ACL. pp. 21–25. ACL, New Orleans, Louisiana (Jun 2018)
32. Stab, C., Gurevych, I.: Identifying Argumentative Discourse Structures in Persuasive Essays. In: Proceedings EMNLP. pp. 46–56. ACL, Doha, Qatar (Oct 2014)
33. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention Is All You Need. arXiv (12 Jun 2017)
34. Wachsmuth, H., Potthast, M., Al Khatib, K., Ajjour, Y., Puschmann, J., Qu, J., Dorsch, J., Morari, V., Bevendorff, J., Stein, B.: Building an argument search engine for the web. In: Proceedings of the 4th Workshop on Argument Mining. p. 49–59. ACL, Stroudsburg, PA, USA (2017)
35. Weber, R.O., Ashley, K.D., Brüninghaus, S.: Textual case-based reasoning. Knowledge Engineering Review **20**(3), 255–260 (Sep 2005)