Quantitative Argument Summarization and Beyond: Cross-Domain Key Point Analysis

Roy Bar-Haim **Yoav Kantor*** **Lilach Eden** **Roni Friedman**
Dan Lahav **Noam Slonim**
IBM Research
{roybar,yoavka,lilache,roni.friedman-melamed,noams}@il.ibm.com
dan.lahav@ibm.com

Abstract

When summarizing a collection of views, arguments or opinions on some topic, it is often desirable not only to extract the most salient points, but also to quantify their prevalence. Work on multi-document summarization has traditionally focused on creating textual summaries, which lack this quantitative aspect. Recent work has proposed to summarize arguments by mapping them to a small set of expert-generated *key points*, where the salience of each key point corresponds to the number of its matching arguments. The current work advances *key point analysis* in two important respects: first, we develop a method for automatic extraction of key points, which enables fully automatic analysis, and is shown to achieve performance comparable to a human expert. Second, we demonstrate that the applicability of key point analysis goes well beyond argumentation data. Using models trained on publicly available argumentation datasets, we achieve promising results in two additional domains: municipal surveys and user reviews. An additional contribution is an in-depth evaluation of argument-to-key point matching models, where we substantially outperform previous results.

1 Introduction

The need for summarizing views, arguments and opinions on a given topic is common to many text analytics applications, across a variety of domains. Some prominent examples for this type of data are responses to open-ended questions in surveys, user reviews on products and services, and posts in online discussion forums. We will hereafter refer to such utterances that express an opinion, view, argument, ask, or suggestion, collectively as *comments*.

Compressing such textual collections into short summaries relies on their inherent *redundancy*. The goal of Multi-Document Summarization (MDS) algorithms is to create short textual summaries from document clusters sharing the same topic. These summaries aim to capture most of the relevant information in the input clusters, while removing redundancies. However, in many cases we would also like to *quantify* the prevalence of each of the points included in the summary. For example, when analyzing the responses of a municipal survey, it would be desirable to let the policy makers know that the point *"The city needs better public transportation"* in the summary matches 8% of the comments, while the points *"Please consider increasing the number of parks, walking and biking trails."* and *"electric rates are too high"* match 4% and 2% of the comments, respectively. The users may also want to drill down to view the comments that were mapped to a specific point in the summary.

Recently, Bar-Haim et al. (2020) proposed *key point analysis* as a summarization framework that meets the above desiderata, in the context of argument summarization. Given a collection of arguments on some topic, their approach aims to match each argument to a short list of *key points*, defined as high-level arguments. In their work, key points were manually composed by an expert, while the matching of arguments to key points was done automatically.

The current work promotes this line of research in two important respects. First, we develop a method for automatic key point extraction (Section 3), allowing fully automatic key point analysis. Our method first selects short, high quality comments as *key point candidates*. It then leverages previous work on argument-to-key-point matching to select a subset of the candidates that achieve high coverage of the data. We show that this relatively simple approach for key point extraction achieves

*First two authors equally contributed to this work.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 39–49,
November 16–20, 2020.

results on argumentation data that are on par with human experts.

The second major contribution of this work is demonstrating the applicability of key point analysis in additional domains beyond argumentation. We report promising results on two datasets: municipal surveys and user reviews. Remarkably, the results are achieved using the same argument matching and argument quality models that were trained on argumentation data, and require only minimal parameter tuning, but no domain-specific labeled data.

An additional contribution is an extensive comparison of pre-trained Transformer models for argument matching, in terms of both accuracy and run time, which results in substantial improvement over the best results reported by Bar-Haim et al. (Section 2).

2 Matching Comments to Key Points

The goal of key point analysis is to extract key points and to match comments to these key points. As mentioned in the previous section (and will be further detailed in the next section), our key point selection algorithm is also based on matching comments to key points, making it a critical component in our system.

We build on the work of Bar-Haim et al. (2020), who developed a large-scale labeled dataset for the task of matching arguments to key points. The dataset, termed *ArgKP*, contains about 24K *(argument, key point)* pairs, for 28 controversial topics. Each of the pairs is labeled as matching/non-matching. Given a set of key points for a topic, an argument could be matched to one or more key points, or to none of them. The arguments in this dataset are a subset of a larger dataset, the *IBM-ArgQ-Rank-30kArgs* dataset, which contains 71 topics, with stance and argument quality annotations for each argument (Gretz et al., 2020).

Bar-Haim et al. only experimented with BERT (Devlin et al., 2019) as a supervised model for argument matching, which they trained on the ArgKP dataset. We aimed to improve their results by testing several more recent transformer-based pre-trained models that were shown to substantially outperform BERT on various tasks (Wang et al., 2018), and in particular on the related task of Recognising Textual Entailment (RTE). We used the HuggingFace transformers framework and fine-tuned four different models: bert-large-uncased (Devlin et al., 2019) (BERT), xlnet-large-cased (Yang et al., 2019) (XLNet), roberta-large (Liu et al., 2019) (RoBERTa) and albert-xxlarge-v1 (Lan et al., 2020) (ALBERT).[1]

We ran 4-fold cross-validation on the ArgKP dataset, where each fold had a train set of 17 topics, development set (dev-set) of 4 topics and test set of 7 topics. The learning rate for each model was tuned based on the final training loss in one of the splits. This learning rate was then used in all four splits. The selected learning rates were 2e-5 for BERT, 7e-6 for XLNet, 5e-6 for RoBERTa and 1e-5 for ALBERT. For choosing the number of epochs, we trained each model with 3 epochs and 9 epochs and selected the one that performed better on the dev-set. All models were better when trained for 9 epochs, except BERT that was better when trained for 3 epochs.

The evaluation results for these models with the above parameters are shown in Table 1. First, we ran inference with each model over all the (argument, key point) pairs in the dev-set and test-set. We then evaluated the following *selection policies* defined by Bar-Haim et al. A selection policy defines how to match an argument to one or more key points, based on the classifier's match score for each key point (kp), and a given threshold t:

- The *threshold (TH)* policy matches the argument to all the kps with match score $> t$.
- The *best match (BM)* policy matches the argument to the kp with the highest match score.
- The *best match+threshold (BM+TH)* policy matches the argument to the kp with the highest match score, if the match score $> t$.

For each fold, we selected the threshold t that maximizes the F1 score over the dev-set.

The model that achieves the best F1 score is ALBERT with an F1 score of 0.809. RoBERTa is second best with an F1 score of 0.773. However, inference time of RoBERTA is about 6 times faster than ALBERT (run times for each model are detailed in Appendix A). Taking run time into account, we decided, for practical reasons, to use the RoBERTa model in the rest of the experiments. We apply this model to arguments, as well as other types of comments in different domains. Notably, both ALBERT and RoBERTa substantially outperform BERT, which only reaches F1 score of 0.721

[1] https://github.com/huggingface/transformers

Model	Selection Policy	Accuracy	Precision	Recall	F1
BERT	TH	0.867	0.677	0.700	0.685
	BM	0.879	0.705	0.716	0.710
	BM+TH	0.893	0.788	0.665	0.721
XLNet	TH	0.897	0.750	0.759	0.752
	BM	0.894	0.743	0.751	0.747
	BM+TH	0.908	0.834	0.709	0.765
RoBERTa	TH	0.897	0.731	**0.803**	0.765
	BM	0.895	0.745	0.753	0.749
	BM+TH	0.913	0.849	0.711	0.773
ALBERT	TH	0.909	0.779	0.794	0.784
	BM	0.908	0.778	0.785	0.780
	BM+TH	**0.926**	**0.877**	0.751	**0.809**

Table 1: Argument-to-Key Point matching results on the ArgKP dataset.

(similar to the F1 of 0.713, reported for BERT by Bar-Haim et al.).

3 Key Point Extraction

In addition to the matching of comments to given key points, we wish to extract the key points automatically from the set of comments, to enable fully-automatic key point analysis. Extraction is performed in two steps: first, a set of key point candidates is selected from the comments and second, the most salient candidates are selected as key points.

3.1 Candidate Extraction

Our approach assumes that the desired key points can be found among the given comments. We start by collecting concise, high quality candidates. We consider only single sentences, and filter out sentences whose length exceed a certain number of tokens. In order to ensure the high quality and argumentative nature of the selected comments, we use the publicly available *IBM-ArgQ-Rank-30kArgs* dataset of Gretz et al. (2020), which consists of around 30k arguments annotated for point-wise quality to train an argument quality ranking model.

We then use this model to compute the argument quality score of each comment, and include only high quality candidates. In addition, we filter out sentences starting with pronouns in order to keep the key points self-contained.

3.2 Key Point Selection

After the set of candidates is extracted, we use the matching model described in Section 2 to obtain a match score between each comment and candidate, and between each pair of candidates.

First, to achieve high coverage of the selected key points, we match comments to candidates by applying the BM+TH selection policy using the matching model and a threshold t, and sort the candidates in descending order according to their coverage, i.e., the number of matched comments. Second, in order to avoid redundancy among the selected key points, we traverse the candidates and remove from the list each candidate whose matching score with a higher-ranked candidate exceeded the threshold.[2] The removed candidates and their matched comments are then matched to the remaining candidates. Finally, the candidates are resorted to form a ranked list of top key points.

The pseudo-code of the algorithm can be found in Appendix B.

4 Experiments

4.1 Evaluaton Method

Let D be a dataset, T the set of topics[3] in D, C_t the set of comments for a topic $t \in T$, and K_t the set of key points extracted for t. Key point analysis finds for each $t \in T$ a set of key points K_t and a mapping from a subset of C_t to K_t. We define *precision* as the fraction of mapped comments for which the mapping was correct, and *coverage* as the fraction of mapped comments out of all the comments.

Our goal is to achieve both high precision and high coverage, however there is typically a tradeoff between the two. This tradeoff can be controlled by setting a threshold on the match score, and applying the BM+TH selection policy to match only a subset of the comments to the key points.

We explore this tradeoff by measuring the precision for different levels of coverage. The precision

[2] As the match scoring function is not symmetric, we compute the match score in both directions and take the average.

[3] Topics may be debate motions in argumentation data, products in user reviews, etc.

at coverage c is defined as the maximal precision such that the coverage is at least c (which can be found by searching over possible threshold values). We measure precision at coverage levels of $0.2, 0.4..., 1.0$.

All the configurations in the following experiments use the matching model that was selected as described in Section 2, and differ only in the set of key points K_t generated for each of the topics $t \in T$ in the dataset D.

The evaluation of each configuration is performed as follows:

1. For each $t \in T, c \in C_t$ we map c to its best matching key point k in K_t, with matching score s.
2. We randomly select from the dataset 500 comments with uniform distribution over the topics. For each sampled comment, we add the tuple $[(c, k), s]$ to our sample.
3. The (c, k) pairs are manually labeled as matched/unmatched (cf. Section 4.4).
4. Based on the manual labeling of the sample, we measure precision at coverage levels of $0.2, 0.4..., 1.0$.

4.2 Datasets

We test our key point analysis method on three datasets: *Arguments*, *Survey* and *Reviews*.

Arguments Dataset. The *IBM-ArgQ-Rank-30kArgs* dataset (Gretz et al., 2020) contains 30k arguments actively collected for and against 71 debatable topics, such as *"Homeopathy brings more harm than good"*[4]. Arguments in the dataset have strict length limitations. Each argument is annotated for its stance towards the topic it discusses and for its quality. As previously mentioned, *ArgKP* was created based on part of this dataset (28/71 topics).

Survey Dataset. Open-ended comments provided by respondents to the *Austin Community Survey*, which took place in 2016 and 2017[5]. Comments were written in response to the following question: *"If there was ONE thing you could share with the Mayor regarding the City of Austin (any comment, suggestion, etc.), what would it be?"*. These comments are raw and unedited, and sometimes contain a few sentences each. The dataset contains $3,188$ comments. Since over 90% of the arguments in our training data are single sentences, and in order to avoid sentences with missing context, only the first sentence of each comment was included in our set.[6]

Reviews Dataset. The *Opinosis* dataset (Ganesan et al., 2010) contains sentences extracted from user reviews on a given topic[7]. Each topic is a combination of product name and review aspect, such as *sound quality of ipod nano*. The dataset contains 51 topics and $7,086$ review sentences, obtained from Tripadvisor (hotels), Edmunds.com (cars) and Amazon.com (various electronics).

4.3 Experimental Setup

Data Splits and Model Training. We used the 28 topics of the ArgKP dataset as training set (24 topics) and development set (4 topics) for the comment matching classifier, which used the selected model as described in Section 2. This model was applied to all three datasets. The remaining 43 topics in the Arguments dataset were used as the test set. Following Bar-Haim et al., we perform key point analysis per topic+stance, 86 pairs in total.

We trained two versions of the argument quality classifier[8]. One was applied to the Arguments test set, so it was only trained on the 24 training topics, with the 4 development topics serving as a development (dev) set. For the Survey and Reviews datasets, we trained a second model on all the available 71 topics.

We did not have training data for the Survey and Reviews datasets. However, we split each of them into test/dev sets, and used the dev set for experimentation and manual parameter tuning. The Survey dataset was split into 314 dev and 2,840 test comments (after comments filtering, as described below). The Reviews dataset was split into 10 dev and 41 test topics.

Filtering and Parameter Tuning. We applied the following filters to each of the datasets. First, comments with non-ascii characters, less than 10

[4] https://www.research.ibm.com/haifa/dept/vst/debating_data.shtml

[5] https://data.world/cityofaustin/mf9f-kvkk

[6] About 50% of the comments contain a single sentence, and in many of the multi-sentence comments, the first sentence captures the main point of the comment.

[7] https://github.com/kavgan/opinosis-summarization/blob/master/OpinosisDataset1.0_0.zip

[8] Replicating the $BERT\_FT_{topic}$ model of Gretz et al. (2020).

characters, under 4 or over 30 tokens (excluding punctuation marks) were removed. In the Arguments dataset, we also removed 10% of the comments that had the lowest quality, as predicted by the argument quality classifier. We did not apply this filter to the other datasets, as we found the quality predictions to be less indicative for their comments. Table 2 lists the number of topics and comments in the three datasets, before and after filtering.

When selecting key point candidates, we aimed to extract about 20% of the shorter, higher quality comments. Since the datasets vary in their characteristics, we adjusted the thresholds for each of them using the respective dev-set. We selected candidates of up to 12 tokens in Arguments and Reviews, and 10 for Survey. The argument quality thresholds were 0.7, 0.4 and 0.35 for Arguments, Survey and Reviews, respectively.

Finally, the key point selection algorithm requires a matching threshold (parameter t in Section 3.2). We tuned this parameter on the dev set of the Arguments dataset, and selected the threshold that maximized the F1, using the BM+TH selection policy. The best threshold (0.856), was used for both the Arguments and Surveys datasets, where key points were extracted for broad topics. The Reviews dataset, however, required finer granularity, as topics were specific aspects of products[9]. Therefore, its threshold was manually set to 0.999 after a few iterations of running the algorithm on the dev set and reviewing the results.

4.4 Human Evaluation

Annotation Process. Using the Appen crowd labeling platform[10], we annotated pairs of comments and key points for match. The instructions stated that *"A key point matches a comment if it captures the gist of the comment, or is directly supported by a point made in the comment"*. In addition to this binary choice, there was also an option to indicate that either key point or comment were not clear (which we considered as *no match* in our assessment). Each comment and key point pair was annotated by 7 crowd annotators. There were three variants of this task:

- Argumentative data - which presented the topic as the context for each comment and key point pair. It also included an additional question regarding the stance of the comment towards the topic, which we used for quality control.
- Survey data - which mentioned the general context in which the comments were written (a community survey about the city of Austin).
- Product review data - which presented product and review aspect as the context for each comment and key point pair.

For each variant, examples matching the type of data labeled were offered in the guidelines.

We employed the following measures to ensure the annotations quality:

- Annotator-κ score - a score measuring inter annotator agreement, averaging all pair-wise Cohen's Kappa for a given annotator, for any annotator sharing at least 50 judgements with at least 5 other annotators, as introduced in Toledo et al. (2019). Judgements of annotators with annotator-$\kappa < 0.1$ were ignored.
- Selected group of trusted annotators - access to the task was limited to a group of annotators with trusted quality, based on previous tasks that were performed for our team, as in Gretz et al. (2020).
- Hidden test questions - for the tasks on argumentative data, stance questions functioned as hidden test questions. As they are based on the *IBM-ArgQ-Rank-30kArgs* dataset, their stance was known. Annotators choosing the wrong stance in more than 15% of their annotations, were ignored.

We consider a pair as a match if it was labeled as a match by more than 50% of the annotators.

Annotations Consistency. Fleiss' Kappa for the match question on this task was 0.38. In the Arguments dataset, where stance was also labeled, stance Fleiss' Kappa was 0.86. Both were calculated prior to any filtering performed on the results.

Previous work has shown for a variety of NLP annotation tasks that while individual crowd annotations have lower quality than expert annotations, expert-level annotation quality can be achieved by aggregating over sufficient number of crowd annotations (Snow et al., 2008). Therefore, crowd annotation quality should be assessed primarily by considering the final, aggregated label.

To this end, we tested the consistency of the la-

[9] When using threshold 0.856, around 90% of the comments for most topics were clustered under a single key point.

[10] https://appen.com/

	Train		Dev		Test	
Dataset	# Topics	Comments Before / After filtering	# Topics	Comments Before / After filtering	# Topics	Comments Before / After filtering
Arguments	24	10,324/10,324	4	1,775/1,599	43	18,398/16,488
Survey	-	-	1	314/272	1	2,840/2,425
Reviews	-	-	10	1,208/1,094	41	5,878/4,845

Table 2: Number of topics and comments per dataset

beled results over different sets of annotators as follows: 300 random comment-key point pairs were selected from the Arguments dataset[11]. Each pair was annotated by 14 different annotators. Annotations for each pair were randomly split to two sets, such that each pair in each set had 7 annotations. After processing each set to produce majority labels, Cohen's Kappa obtained between the pair labels of each set was 0.63.

4.5 Results and Discussion

The results for the three datasets are summarized in Table 3. Fully automatic key point analysis is shown to perform well on the Arguments test set: precision of 0.752 and 0.792 when matching all the comments to 5 and 10 key points, respectively. When matching 60% of the comments, we achieve precision above 90%. Table 4 shows an example for key points generated for one of the topic+stance pairs in the Arguments datasets, and their distribution over the comments for that topic and stance.

We also compared our automatic key point extraction to the approach taken by (Bar-Haim et al., 2020), where key points were manually created by a debate expert. Following Bar-Haim et al., the expert composed 7 key points per topic+stance, based on his domain knowledge, and without being exposed to the comments. A total of 70 key points were composed, for 10 randomly-sampled topic+stance pairs from the test set. Comparing the results for these key points with our automatic results for the same number of key points shows that we were able to achieve similar precision (0.696 vs. 0.708) over all the comments (coverage of 100%). The precision for coverage of 80% is also comparable (0.8 vs. 0808). For lower coverage rates, the precision for the manual key points is higher.

To evaluate the similarity between our automatically extracted key points and the ones generated by the human expert, we attempted to match each automatic key point to an associated manually composed key point. Out of the 70 KPs, 10 were classified as *Matching* - the key points are essentially the same; 32 were *Related* - the key points reflect a similar point or one key point is entailed by the other; 16 were *Remote* - the key points are connected but there exists a distinct change that makes them different in essence, and only 12 were unrelated. These results suggest that the automatic process was able, to a large extent, to mimic the analysis of a human expert. We also found that manually composed key points tend to be more abstract, and in some cases a single manual point matched several more specific automatically extracted points.

Remarkably, our method, which makes use of models trained on argumentation data, performs reasonably well also when applied to survey and user reviews data. Presumably, the comments in these datasets also contain argumentation, which allows to transfer the knowledge learned from the argumentation dataset to these domains. The argumentation in the Arguments dataset is more explicit, though, as the contributors to this dataset were asked to provide pro and con arguments for the given controversial topics.

For the Survey dataset, we achieve precision of 0.763 when matching 60% of the comments in the labeled sample to 20 key points[12]. Table 5 shows KP analysis results for this coverage rate, including the extracted key points, their distribution, comment matching precision per key point, and the top two matching comments for each key point. While the extracted key points are largely concise and to the point, the results could be further improved with some manual post-processing. For example, the first KP can be rephrased as *"Reduce traffic congestion"*, removing the extra part about a monorail system, which is not mentioned in the top comments. We can then re-match the comments to the revised KPs, and the process can iterate, until both

[11] This dataset had the lowest Fleiss kappa of the three - 0.34. Survey dataset kappa was 0.41 and Reviews dataset kappa was 0.37

[12] We used here a larger number of key points since, unlike the other two datasets, the Survey test set contains a single topic with more than 2,400 comments.

		Precision						
	Dataset	Arguments (All)		Arguments (Subset)		Survey	Reviews	
	Configuration	Auto 5 KPs	Auto 10 KPs	Auto 7 KPs	Expert 7 KPs	Auto 20 KPs	Auto 2 KPs	Gold 2 KPs
Coverage	0.2	0.911	0.933	0.843	0.948	0.873	0.814	0.811
	0.4	0.911	0.932	0.843	0.948	0.824	0.796	0.770
	0.6	0.906	0.915	0.837	0.905	0.763	0.731	0.642
	0.8	0.854	0.883	0.800	0.808	0.638	0.670	0.544
	1.0	0.752	0.792	0.696	0.708	0.514	0.568	0.454

Table 3: Results for the Arguments, Survey, and Reviews datasets.

Key Point	**%**
People who have three minor offences are unfairly punished.	30%
The three strike law has not proved effective in reducing criminality	15%
The three strike law prohibits reform of offenders.	12%
Many people could pay long sentences for nonviolent crimes	12%
The three-strikes law has resulted in overcrowded prisons	8%
The 3 strikes law doesn't allow judicial discretion in sentencing	7%
The three-strikes law costs tax payers too much money.	6%
The three-strikes law is inequitable and targets men of color.	5%
The three strike law is too strict for some offenders	5%

Table 4: Top key points and their coverage for the topic *"We should abolish the three-strikes laws"* and *Pro* stance from the Arguments dataset, when generating up to 10 key points using the selection algorithm. After generating the key points list, each of the 267 comments is matched to a key point using the BM selection policy.

coverage and precision are satisfactory.

The precision over all the comments was 0.514. We note that key point analysis can be effectively applied even if the matching precision is not very high. For example, suppose that 10% of the comments were matched to a certain key point with precision of only 50%. This means that in practice, 5% of the comments do match this key point, so it is an important point nonetheless.

For the Reviews dataset, we selected two key points per topic, since this was the length of summaries in the experiments conducted by Ganesan et al. on this data. We compared our results to a configuration where the key point candidates are the union of the sentences in the human-generated gold summaries that were released as part of this dataset.

We obtained precision of 0.731 for coverage of 60%, and 0.568 for 100% coverage, better than the results for the key points that were based on the gold summaries (0.642 and 0.454, respectively). The precision differences in coverage levels of 0.6 and above are statistically significant[13]. Table 6 shows both the automatically extracted key points and the key points selected from human summaries, along with their coverage, for several selected topics.

Error Analysis. The dominant types of matching errors differed amongst the datasets. The most common type in Reviews data was the comment and KP having opposite polarity. This was expected, since in the ArgKP training data, the argument and the key point always have the same polarity. The highest proportion of comments that were not related to the key point was in the Survey dataset. This is likely an outcome of analyzing all the comments in this dataset under a single topic. The dominant issue in the Arguments dataset was key points that were related to the comment but had slight changes that altered their meaning. For example: *"if people have been committing crimes anyway, they deserve to be caught through whatever means necessary, including the use of entrapment."* was matched to *"sometimes the only way police can catch a criminal is by entrapment."*, where the phrase *"the only way"* was crucial for capturing the right meaning of the key point.

5 Related Work

The task of *Multi-document summarization (MDS)* is defined as creating a summary that is both concise and comprehensive from multiple texts with a shared theme, e.g., news articles covering the same event (McKeown et al., 2002). A major chal-

[13] Using Z test for two population proportions, with $p = 0.05$ for coverage of 0.6, and $p = 0.01$ for coverage of 0.8 and 1.0.

Key Point	%	P	Top Comments
Consider a monorail system to help traffic congestion	9%	0.74	Need much, much better traffic flow, (example, 183 or 620, Palmer).
			Traffic flow is terrible!
Austin needs better public transportation	8%	0.90	For a progressive city, Austin is lacking in public transportation.
			Make improvements to public transportation in north Austin.
Affordability of housing and living in Austin	5%	0.85	Address rapidly increasing cost of living
			The cost of living here is insane.
Rising property values and taxes.	5%	0.77	Reduce property taxes and housing costs so that retiring and still living here is a real possibility.
			*This city is not affordable due to horrendous tax and service fees including all city service bills - electric, water, etc.
Please consider increasing the number of parks,walking and biking trails.	4%	0.84	Consider better developed bike lanes throughout the city.
			Developing of greenery areas and more parks.
Austin utility services need an overhaul-especially water/wastewater.	4%	0.78	City needs to fix serious drainage issues, and let citizens protect their homes while they await a cure.
			Water/wastewater rates are ridiculous.

Table 5: Top key points for the City of Austin Community Survey. Match threshold was set so that the extracted 20 key points cover 60% of the sampled comments. For each key point we show the percentage of matching comments (out of the sampled comments), the precision of matched comments and the top two matching comments. All comments shown in the tables were judged as correct matches, except for the one marked with '*'.

Topic	KPs Extracted from Gold Summaries	%	KPs Extracted from Reviews	%
Accuracy of Garmin nuvi 255W GPS	The garmin seems to be generally very accurate.	73%	Most of the times, this info was very accurate.	72%
	Set-up and usage are considered to be very easy.	13%	Easy to use, excellent accuracy, nice and intuitive interface.	16%
Battery-life of iPod Nano 8GB	The battery life of the ipod nano is very short.	79%	The only bad thing is it's battery life.	90%
	It seems to continue using battery even when the ipod is not in use, otherwise, it's a great product.	8%	Long battery life and easy directions make this a snap to use.	7%
Rooms of Bestwestern Hotel SFO	Good, clean and tidy rooms and bathroom.	39%	The hotel had nice, well, decorated, fairly, modern rooms.	30%
	The rooms were small.	25%	The rooms are a bit small, but not unusual for San Francisco.	18%

Table 6: Top two key points extracted from gold summaries and from original reviews, on selected topics from the Reviews dataset. For each key point we show the percentage of matching comments, with match threshold 0.999.

lenge for applying supervised learning to MDS has been the limited amount of available training data. Most of the approaches applied to the task were extractive, operating over graph-based representations of sentences or passages (Erkan and Radev, 2004; Christensen et al., 2013). Recently, Liu et al. (2018) proposed a method for creating a large-scale dataset from Wikipedia (WikiSum), which allowed training an abstractive neural model for this task. Key point analysis adds a quantitative dimension that is not addressed by MDS, by measuring the prevalence of each point in the summary.

Many of the works on Opinion Summarization take an alternative, sentiment-based approach. These works aim to identify the main aspects discussed in user reviews, and quantify the sentiment towards each of these aspects (Hu and Liu, 2004; Snyder and Barzilay, 2007; Titov and McDonald, 2008). However, as noted by Ganesan et al. (2010), it is still hard for a user to understand why an aspect received a particular rating. As demonstrated in Table 6, key points can address this limitation by providing a more informative summary of user reviews. However, the detection of the stance (or sentiment) of each key point with respect to the topic was left out of the scope of the current work, and we plan to address it in future work.

In computational argumentation, several works have focused on pairwise argument similarity and clustering (Ajjour et al., 2019; Reimers et al., 2019; Misra et al., 2016). These works, however, did not attempt to create textual summaries from the resulting clusters. Egan et al. (2016) summarized argumentative discussions through the extraction of salient "points", where each point is a verb and its syntactic arguments. The current work also extracts points from argumentative data, but our goal is to go beyond textual summaries, by matching each key point to its corresponding sentences in the input data. Similar to Egan et al., we also experimented with extracting syntactic subtrees as key points, but found that this often results in incomplete sentences or omission of important information. Selecting short, high quality sentences as key points was found to perform better in our experiments.

The line of research that is most relevant to the current work deals with matching argumentative texts to predefined, short lists of manually-composed arguments or points (Hasan and Ng, 2014; Boltužić and Šnajder, 2014; Naderi, 2016). Bar-Haim et al. (2020) matched crowd-contributed arguments, taken from the dataset of Gretz et al. (2020), to key points composed by a debate expert. We used the labeled dataset developed by Bar-Haim et al. to train our comment matching model.

As previously discussed, the main contributions we make to this line of work are (i) Fully-automatic key point analysis, enabled by automatic key point extraction, and (ii) Demonstrating the applicability of key point analysis to additional domains besides argumentation, including surveys and user reviews. Furthermore, we were able to achieve promising results on these domains using models that were only trained on argumentation data.

6 Conclusion

Key Point Analysis is a novel framework for summarizing arguments, opinions and views. It provides both textual and quantitative view of the main points in the summarized data, and allows the user to interactively drill down from points to the actual sentences they cover. Previous work only applied key point analysis in the context of argumentation data, and required a domain expert for writing the key points.

The current work addresses both of the above limitations. First, we present an automatic method for key point extraction, which is shown to perform on par with a human expert. Second, our work demonstrates the potential of key point analysis in multiple domains besides argumentation. Furthermore, we show that the necessary knowledge for key point analysis, once acquired by supervised learning from argumentation data, can be successfully applied cross-domain, making it unnecessary to collect domain specific labeled data for each target domain.

In future work, we would like to improve comment matching, e.g., by making it stance-aware. We also plan to experiment with sequence-to-sequence neural models for generating key point candidates from comments.

References

Yamen Ajjour, Milad Alshomary, Henning Wachsmuth, and Benno Stein. 2019. Modeling frames in argumentation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2922–2932, Hong Kong, China. Association for Computational Linguistics.

Roy Bar-Haim, Lilach Eden, Roni Friedman, Yoav Kantor, Dan Lahav, and Noam Slonim. 2020. From arguments to key points: Towards automatic argument summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4029–4039, Online. Association for Computational Linguistics.

Filip Boltužić and Jan Šnajder. 2014. Back up your stance: Recognizing arguments in online discussions. In *Proceedings of the First Workshop on Argumentation Mining*, pages 49–58, Baltimore, Maryland. Association for Computational Linguistics.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2013. Towards coherent multi-document summarization. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1163–1173, Atlanta, Georgia. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Charlie Egan, Advaith Siddharthan, and Adam Wyner. 2016. Summarising the points made in online political debates. In *Proceedings of the Third Workshop on Argument Mining (ArgMining2016)*, pages 134–143, Berlin, Germany. Association for Computational Linguistics.

Günes Erkan and Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Intell. Res.*, 22:457–479.

Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. 2010. Opinosis: A graph based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 340–348, Beijing, China. Coling 2010 Organizing Committee.

Shai Gretz, Roni Friedman, Edo Cohen-Karlik, Assaf Toledo, Dan Lahav, Ranit Aharonov, and Noam Slonim. 2020. A large-scale dataset for argument quality ranking: Construction and analysis. In *AAAI*.

Kazi Saidul Hasan and Vincent Ng. 2014. Why are you taking this stance? identifying and classifying reasons in ideological debates. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 751–762, Doha, Qatar. Association for Computational Linguistics.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 168–177, New York, NY, USA. ACM.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Peter J. Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

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

Kathleen R. McKeown, Regina Barzilay, David Evans, Vasileios Hatzivassiloglou, Judith L. Klavans, Ani Nenkova, Carl Sable, Barry Schiffman, and Sergey Sigelman. 2002. Tracking and summarizing news on a daily basis with columbia's newsblaster. In *Proceedings of the Second International Conference on Human Language Technology Research*, HLT '02, page 280–285, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Amita Misra, Brian Ecker, and Marilyn Walker. 2016. Measuring the similarity of sentential arguments in dialogue. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 276–287, Los Angeles. Association for Computational Linguistics.

Nona Naderi. 2016. Argumentation mining in parliamentary discourse. In *Proceedings of the 11th International Conference of the Ontario Society for the Study of Argumentation*, pages 1–9.

Nils Reimers, Benjamin Schiller, Tilman Beck, Johannes Daxenberger, Christian Stab, and Iryna Gurevych. 2019. Classification and clustering of arguments with contextualized word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 567–578, Florence, Italy. Association for Computational Linguistics.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Ng. 2008. Cheap and fast – but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 254–263, Honolulu, Hawaii. Association for Computational Linguistics.

Benjamin Snyder and Regina Barzilay. 2007. Multiple aspect ranking using the good grief algorithm. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 300–307, Rochester, New York. Association for Computational Linguistics.

Ivan Titov and Ryan McDonald. 2008. A joint model of text and aspect ratings for sentiment summarization. In *Proceedings of ACL-08: HLT*, pages 308–316, Columbus, Ohio. Association for Computational Linguistics.

Assaf Toledo, Shai Gretz, Edo Cohen-Karlik, Roni Friedman, Elad Venezian, Dan Lahav, Michal Jacovi, Ranit Aharonov, and Noam Slonim. 2019. Automatic argument quality assessment - new datasets and methods. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5624–5634, Hong Kong, China. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the Workshop: Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2018, Brussels, Belgium, November 1, 2018*, pages 353–355. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 5754–5764.

Appendices

A Matching Models Run Times

Table 7 lists run-time measurements for one of the splits of the ArgKP dataset: training over 15,235 argument-kp pairs in the train-set and inference over 3,776 pairs in the dev-set and 6,839 pairs in the test-set, using an NVIDIA Tesla V100 GPU.

	Train	Dev	Test
BERT	00:18:59	00:00:17	00:00:32
XLNet	01:09:38	00:00:23	00:00:42
RoBERTa	00:59:43	00:00:17	00:00:31
ALBERT	01:06:50	00:01:39	00:03:03

Table 7: Run time (hours:minutes:seconds)

B Key Point Selection Algorithm

The pseudo-code of the key point selection algorithm (Section 3.2) is listed in Algorithm 1. Given a set of comments, a set of key point candidates and a threshold t, the algorithm outputs a sorted list of selected key points.

Algorithm 1 Key Point Selection
Input: Comments C, KP Candidates K, Threshold t
Output: A ranked subset of K

procedure SELECT_KEY_POINTS(C, K, t)
  $k\_to\_c \leftarrow Get\_Matches(C, K, t)$
  $K \leftarrow$ sort_descending(keys of k_to_c) by #matches
  $R \leftarrow []$
  for $k1$ in K **do**
    for $k2$ in K up to and excluding k1 **do**
      $s \leftarrow Avg(Score(k1, k2), Score(k2, k1))$
      if $s > t$ **then**
        add $k1 \cup k\_to\_c[k1]$ to R
        remove $k1$ from k_to_c
        break
      end if
    end for
  end for
  $kps \leftarrow$ keys of k_to_c
  $kp\_to\_c \leftarrow k\_to\_c \cup Get\_Matches(R, kps, t)$
  return sort_descending(keys of kp_to_c) by #matches
end procedure

procedure GET_MATCHES(C, K, t)
  $k\_to\_c \leftarrow \{\}$
  for c in C **do**
    $match\_c \leftarrow \text{argmax}_{k \in K}\, Score(c, k)$
    if $Score(c, match\_c) > t$ **then**
      add c to $k\_to\_c[match\_c]$
    end if
  end for
  return k_to_c
end procedure