Modern Talking in Key Point Analysis: Key Point Matching using Pretrained Encoders

Jan Heinrich Reimer and **Thi Kim Hanh Luu** and **Max Henze** and **Yamen Ajjour**
Martin Luther University Halle-Wittenberg, Germany
{jan.reimer,thi.luu,max.henze}@student.uni-halle.de
yamen.ajjour@informatik.uni-halle.de

Abstract

We contribute to the ArgMining 2021 shared task on Quantitative Summarization and Key Point Analysis with two approaches for argument key point matching. For key point matching the task is to decide if a short key point matches the content of an argument with the same topic and stance towards the topic. We approach this task in two ways: First, we develop a simple rule-based baseline matcher by computing token overlap after removing stop words, stemming, and adding synonyms/antonyms. Second, we fine-tune pretrained BERT and RoBERTa language models as a regression classifier for only a single epoch. We manually examine errors of our proposed matcher models and find that long arguments are harder to classify. Our fine-tuned RoBERTa-Base model achieves a mean average precision score of 0.913, the best score for strict labels of all participating teams.

1 Introduction

Arguments influence our decisions in many places of our daily life (Bar-Haim et al., 2020a). But with the increasingly larger amount of information found on the Web[1] and more effective argument mining, people often need to summarize arguments (Lawrence and Reed, 2019; Bar-Haim et al., 2020a). Bar-Haim et al. (2020a) see matching key points to arguments as an intermediate step towards automatically generating argumentative summaries (Section 2). The ArgMining 2021 shared task on Quantitative Summarization and Key Point Analysis (Friedman et al., 2021) is the first task on key point matching, which is an important step towards summarizing arguments. By matching arguments with a pre-defined set of key points, an argumentative text can be summarized using the prevalence of the key points in it. Different approaches of matching argument key point pairs, here called matchers, should be proposed and discussed. Given an argument and a key point, a matcher should return a real value between 0 and 1 which represents the extent to which the argument matches the key point.[2] For evaluating different argument key point matchers, the shared task organizers uses mean average precision evaluation as a metric (Friedman et al., 2021) to evaluate the approaches and publish the ArgKP-2021 benchmark dataset (Section 3) to compare matchers (Bar-Haim et al., 2020a).

Pretrained language models like BERT and RoBERTa are nowadays becoming standard approaches to tackle various Natural Language Processing tasks (Devlin et al., 2019; Liu et al., 2019). Because of their extensive pretraining, often fine-tuning these language models with even a small task-specific dataset can achieve state-of-the-art performance (Devlin et al., 2019). As the ArgKP-2021 dataset (Bar-Haim et al., 2020a) used in the ArgMining 2021 shared task on Quantitative Summarization is relatively small (24 083 labelled pairs), we decide to fine-tune BERT and RoBERTa language models rather than train a neural classifier from scratch (Section 4).

Contrasting this neural approach, we introduce a simple rule-based baseline matcher that compares preprocessed tokens of each argument to the tokens of each key point (Section 4). For the baseline, we compute token overlap after removing stop words, adding synonyms and antonyms, and stemming the tokens from both argument and key point using the NLTK toolkit (Bird and Loper, 2004).

Our fine-tuned RoBERTa-Base matcher achieves a mean average precision score of up to 0.967 and ranks second in the shared task's leaderboard (Section 5). In a manual error analysis, we find that the imbalanced ArgKP-2021 dataset causes neural models to predict non-matching argument key

[1] https://internetlivestats.com/

[2] https://2021.argmining.org/shared_task_ibm.html

Proceedings of The 8th Workshop on Argument Mining, pages 175–183
Punta Cana, Dominican Republic, November 10–11, 2021.

point pairs more precisely than matching pairs (Section 6). We further observe a tendency that large length differences between arguments and key points can cause errors. To encourage researchers to train more robust argument key point matchers, we release our source code under a free license.

2 Related Work

Similar tasks to key point analysis include clustering arguments (Reimers et al., 2019; Ajjour et al., 2019), detecting similar arguments in a pairwise fashion (Misra et al., 2016) and matching arguments to generic-arguments (Naderi and Hirst, 2017). Using points to summarize arguments were approached by Egan et al. (2016) on online discussion. Points were extracted by using the verbs and their syntactic arguments and are then clustered together to deliver a summary of the discussion.

Key point analysis is the task of matching a given argument with one or more pre-defined key points (Bar-Haim et al., 2020a). To develop models for the task, Bar-Haim et al. (2020a) introduced a dataset (ArgKP-2021) which contains 24 093 argument key point pairs on 28 topics. Each argument and key point is labeled manually as `match` or `no-match`. The authors experimented with several unsupervised and supervised approaches to perform the task in a cross-topic experimental setting. BERT (Devlin et al., 2019) performed the best in their experiments by reaching an F1 score of 0.68.

In a later work, Bar-Haim et al. (2020b) develop a summarization approach for online discussions that uses key point analysis. The summrization approach takes as input a set of comments on a given topic and extracts a set of representative key points from them. The output of the summarization approach is the set of extracts key points together with the count of matched comments for each key point. In its essence, the summarization approach uses a matching model that gives a score for a given comment and key point or a pair of key points. For matching models, Bar-Haim et al. (2020b) compare different variants of BERT (Devlin et al., 2019). Among the tested models, ALBERT (Lan et al., 2019) performed the best with an F1 score 0.809, but RoBERTa (Liu et al., 2019) were chosen for key point extraction at the end, which is 6 times faster than ALBERT and still achieves an F1 score of 0.773.

Our approaches for the key point analysis are based on BERT and RoBERTa. BERT stands for Bidirectional Encoder Representations from Transformers and is an open-source bidirectional language representation model published by Google (Devlin et al., 2019). BERT is pre-trained over unlabeled text to learn a language representation and can be fine-tuned on downstream tasks. During pre-training, BERT is trained on two unsupervised tasks: Masked Language Model and Next Structure Prediction. RoBERTa is an improved variant of BERT that is introduced by Facebook in 2019 (Liu et al., 2019). Liu et al. (2019) modified BERT by using a larger training data size of 160GB of uncompressed text, more compute power, larger batch-training size, and optimized hyperparameters. In comparison to BERT, pre-training tasks for RoBERTa were done with full-length sentences and include only Masked Language Model while applying different masks in each training epoch (dynamic masking). RoBERTa outperforms BERT on all 9 GLUE tasks in the single-task setting and 4 out of 9 tasks in the ensembles setting (Wang et al., 2018; Liu et al., 2019).

3 Data

The dataset used in the ArgMining 2021 shared task on Quantitative Summarization and Key Point Analysis is the ArgKP-2021 dataset (Bar-Haim et al., 2020a) which consists of 24 083 argument and key point pairs labeled as matching/non-matching. They all belong to one of 28 controversial topics, for example: "Assisted suicide should be a criminal offence". Every key point and argument pair is annotated with its stance towards the topic.

The training split of the ArgKP-2021 dataset has 5 583 arguments belonging to 207 key points within 24 topics. This leaves the validation split with 932 arguments and 36 key points for 4 topics. Friedman et al. (2021) complement the ArgKP-2021 dataset's training and validation split with a test split that is used to evaluate submissions to the shared task. The test split contains 723 arguments with 33 key points from 3 topics.

3.1 Characteristics

Here, we do qualitative and quantative analyses of the ArgKP-2021 dataset. Table 1 shows examples of argument key point pairs from the ArgKP-2021 dataset (Bar-Haim et al., 2020a). In pair A from Table 1, the argument matches the given key point.

Table 1: Examples of argument key point pairs from the ArgKP-2021 dataset (Bar-Haim et al., 2020a)

#	Argument	Key point
A	child actors can be overworked and they can miss out on their education.	Being a performer harms the child's education
B	as long as nuclear weapons exist, the entire world has to worry about nations deciding to fire them at another or terrorists getting hold of them and causing disaster	Nuclear weapons can fall into the wrong hands
C	'people reach their limit when it comes to their quality of life and should be able to end their suffering. this can be done with little or no suffering by assistance and the person is able to say good bye.	Assisted suicide reduces suffering

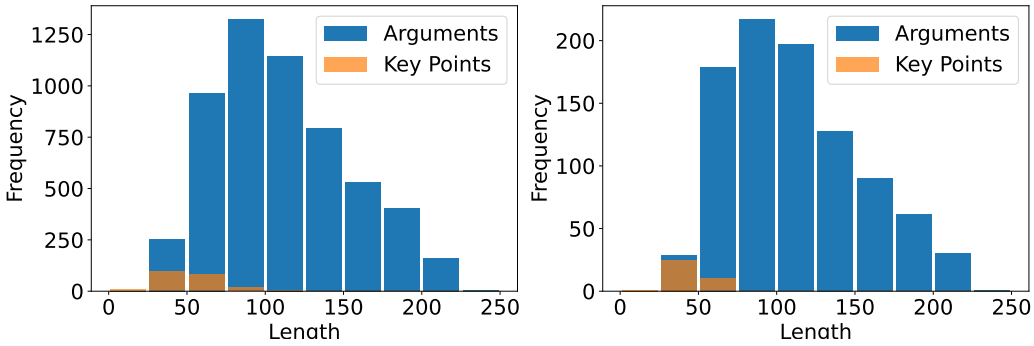


Figure 1: Lengths in characters for arguments and key points from the training and development set.

Both sentences discuss children actors and their education. The word "actors" is not explicitly used in the key point but is semantically similar to the word "performer". Such lexical variation can be opposed by using WordNet (Miller, 1995) to find synonyms and antonyms.

Pair B in Table 1 is a harder example, since the argument matches the key point but are expressed differently. The key point makes usage of "wrong hands" as figurative meaning for "nations" and "terrorists" from the argument. In comparison to pair A, the linguistic variation in pair B goes beyond finding synonyms and requires a deep understanding of the semantics of the argument and key point.

Figure 1 shows the average length of the arguments and key points in the training and developments splits. As shown, the arguments in the ArgKP-2021 dataset are substantially longer than key points. In the training set, the average length of arguments is 109 characters. Compared to that, key points are on average only half as long (52 characters). In the validation set, the key points have an average length of 41 characters and therefore key points are shorter than those in the training set. The average length of arguments remains almost the same at 108 characters. The proportion of arguments that are 67 characters longer than key points constitute 39 % of the training set and 44 % of the validation set. We can see that there are more short key points in the validation set. This length difference might be a challenge for the models in key point matching (Section 6). Pair C is an example of an argument and key point pair with a large length difference.

All in all, we identify the following major difficulties in matching key points to arguments: semantically similar words, meaning understanding, and the length difference between the arguments and key points. In the following section, we approach the first two problems while developing our baseline and approaches. In Section 6, we analyze the errors made by our approaches with regard to the length difference between the arguments and key points.

4 Approach

To match key points to arguments, we propose two different approaches. First, we discuss a simple yet effective baseline measuring token overlap be-

tween key points and arguments. Second, to improve upon this simple baseline, we introduce an approach based on BERT and RoBERTa (Devlin et al., 2019; Liu et al., 2019). We fine-tune both language models in standard configuration with only minor changes highlighted below.

4.1 Token Overlap Baseline

To be able to compare more sophisticated matchers, we first propose a very simple token overlap baseline using preprocessed tokens from each argument and key point, as parsed by the NLTK toolkit (Bird and Loper, 2004). In general, key points summarize ideas of their matched arguments. Our intuition, therefore, is that certain words or tokens from an argument are also likely to be present in its matched key points. Rather than using completely new words for summarization of arguments, a human would tend to reuse important words from the argument. For example, in the argument and key point pair C from Table 1 both, the argument and key point, contain the token "suffering".

We can further increase the token overlap between arguments and key points by preprocessing their tokens as following: First, we remove stop words for reducing noise within all arguments. Initially, this can seem counterproductive because with fewer words the highest possible overlap would also decrease and therefore could lead to worse performance. However, a lot of arguments and key points contain functional words like "the", "and" or "as". Removing these results in sentences that contain more specific information and thus leads to less confusion with the token overlap matcher. As a second preprocessing step, we reduce tokens to their corresponding stems by applying stemming using the Snowball stemmer (Porter, 1980). We expect the token overlap matcher to be able to generalize more when comparing tokens. For example, the words "assistance" and "assisted" from the above example (Table 1, C) are both stemmed to "assist" with the Snowball stemmer. Consequently, stemming creates an overlap between different forms of the same word and, for instance, increases the probability that an argument containing "assistance" is associated with a key point containing "assisted". Last, we increase generalization for token overlap even further by supplementing the set of tokens with synonyms and antonyms (Miller, 1995). This step should also increase the chance of overlapping tokens.

To compute the similarity between an argument and a key point, let tokens_a be the set of preprocessed tokens from an argument a and tokens_k the set of tokens from a key point k. We calculate the set of overlapping tokens like this:

$$\text{overlap}_{a,k} = \{t : t \in \text{tokens}_a \wedge t \in \text{tokens}_k\} \quad (1)$$

The token overlap matcher returns matching scores based on the overlap size weighted against the minimum size of either the argument or the key point:

$$\text{score}_{a,k} = \frac{|\text{overlap}_{a,k}|}{\min\{|\text{tokens}_a|, |\text{tokens}_k|\}} \quad (2)$$

That is, pairs with a higher proportion of tokens that appear both in the argument as well as in the key point are classified with a higher matching score.

4.2 Transformers Fine-tuning

To improve upon this simple token overlap baseline, we fine-tune BERT and RoBERTa Transformer models for classifying argument key point matches (Devlin et al., 2019; Liu et al., 2019). While BERT is pretrained on a very large document corpus (16GB of raw data), RoBERTa is pretrained on an even larger corpus (160GB). Thus RoBERTa models can be fine-tuned on higher end task performance (Liu et al., 2019). We tokenize both the arguments and the key points with BERT's default WordPiece tokenizer and the resulting sequences are trimmed to 512 tokens for both models. We then fine-tune the BERT-Base and RoBERTa-Base variants in the standard sentence-pair regression setting using the Simple Transformers library.[3] The input to the models is formatted as `[CLS] argument [SEP] key point [SEP]` for BERT and `<s> argument </s></s> key point </s>` for RoBERTa respectively. For classification, we interpret the regression output value as the probability of an argument matching a key point. That is, the training labels are always 0 or 1, depending on whether the corresponding pair in the training set matches or not. Both model variants contain 12 hidden layers with a hidden size of 768 and 12 attention heads. We train each of the two models for one single epoch at a learning rate of $\eta = 2 \cdot 10^{-5}$. We use an AdamW optimizer with $\beta = (0.9, 0.999)$ and zero weight decay (Loshchilov and Hutter, 2019). The optimizer is warmed up with a ratio of 6 % of the training data, and we fine-tune both models with

[3] https://simpletransformers.ai/

the binary cross-entropy loss. We explore three ways of handling argument key point pairs in the training set with missing ground-truth label. In the first way, we remove those pairs completely from the traning dataset. In the second and third ways, we assume all the arguments and key points with missing labels to be either a `match` or `no-match`. By comparing the effectiveness of the models, we find that the first way lead to the best effectiveness on the validation set. Similarly, we experiment with textual data augmentation[4] (swapping synonyms, randomly omitting words) to increase the amount of training data, leading to no improvement on validation scores either. Thus, for the submitted model, we consider only training pairs that have an associated ground-truth label and do not oversample. We don't restrict the models output to the interval $[0, 1]$—like we did for the baseline—, as the shared task did not mention constraints on the score value that should be returned by a matcher.

5 Results

Submissions to the ArgMining 2021 shared task on Quantitative Summarization and Key Point Analysis are evaluated with respect to mean average precision (Friedman et al., 2021). The organizers calculate the score by pairing each argument with the best matching key point according to the predicted matching probabilities. Within each topic-stance combination, only 50 % of the arguments with the highest predicted matching score are then considered for evaluation. The task organizers claim that this removal of 50 % of the pairs is necessary because some arguments do not match any of the key points, which would influence mean average precision negatively (Friedman et al., 2021). For the remaining argument key point pairs in each topic-stance combination, the average precision is calculated and the final score is computed as the mean of all average precision scores.

The task organizers consider two evaluation settings: strict and relaxed. Both settings are based on the ground-truth labels from the ArgKP-2021 dataset (Bar-Haim et al., 2020a). The two evaluation settings are created to account for argument key point pairs in the ArgKP-2021 with undecided labels (i.e. not enough agreement between annotators). In the strict setting, the shared task organizers consider those undecided pairs as `no-match`. In the relaxed setting, however, the shared task organizers consider the undecided pairs as `match` (Friedman et al., 2021). The mean average precision score is then calculated in the two settings based on the ground-truth labels and the derived labels for the undecided pairs. We stress that in this complex evaluation setup, the mean average precision score in the relaxed setting would favor assuming matches in case of model uncertainty. In comparison, in the strict setting mean average precision would favor assuming `no-match` between an argument and key point. However, we find that because only the most probable matching key point is being considered for evaluation, this effect is minor. The evaluation score in general favors matchers that can match a single key point for each argument with high precision. It is however not important if a matcher does predict non-matches with high certainty.

5.1 Discussion

In Table 2, we report mean average precision in the strict and relaxed settings of the training, validation, and test set in the ArgKP-2021 dataset. We complement the mean average precision scores by adding precision and recall scores of the `match` label, both in the strict and relaxed setting. To calculate precision and recall, we label an argument and key point pair as `match` if their score is higher than 0.5 and as `no-match` otherwise. To aggregate results of the strict and relaxed settings, we also report the average score of the two variants. The reported scores should allow for automated and unbiased evaluation of our models and easier comparison with competitive approaches. We report all 27 scores for the token overlap baseline model as well as for the fine-tuned BERT-Base and RoBERTa-Base models. To make our results more comparable, we add a second baseline, where matches between arguments and key points of same topic and stance are predicted with uniform random probability. That random baseline represents a worst-case matcher and any weak matcher should exceed its evaluation scores.

The token overlap baseline achieves a mean average precision of 0.483 in the strict setting and 0.575 in the relaxed setting on the test set. Thus, it is nearly twice as good as a random matcher with respect to mean average precision. Even though this baseline has reasonably good scores on all datasets, we are concerned about the large discrepancies between its scores on the validation set and the train-

[4] https://github.com/makcedward/nlpaug

Table 2: Performance of the random and token overlap baseline, BERT-Base, and RoBERTa-Base models with respect to mean average precision (mAP), precision (P), and recall (R) of the match label. Precision and recall are calculated by deriving boolean labels from the matching scores with a threshold of 0.5 for all approaches. We report scores for the training, validation, and test set in the strict and relaxed label settings, as well as the averages of the two settings. The best result per set is highlighted **bold**.

Approach	Strict			Relaxed			Average		
	mAP	P	R	mAP	P	R	mAP	P	R
Training set									
Random	0.260	0.173	0.500	0.409	0.330	0.501	0.335	0.252	0.501
Token Overlap	0.541	0.269	0.323	0.653	0.435	0.275	0.597	0.352	0.299
BERT-Base	0.889	**0.703**	**0.864**	**0.981**	**0.936**	**0.607**	0.935	**0.819**	**0.736**
RoBERTa-Base	**0.915**	0.702	0.820	0.979	0.927	0.572	**0.947**	0.814	0.696
Validation set									
Random	0.232	0.180	0.523	0.430	0.364	0.524	0.331	0.272	0.524
Token Overlap	0.643	0.219	0.390	0.802	0.416	0.366	0.722	0.317	0.378
BERT-Base	0.717	0.397	**0.802**	0.928	0.648	**0.649**	0.822	0.522	**0.725**
RoBERTa-Base	**0.879**	**0.567**	0.799	**0.984**	**0.816**	0.569	**0.932**	**0.692**	0.684
Test set									
Random	0.237	0.150	0.545	0.355	0.286	0.549	0.296	0.218	0.547
Token Overlap	0.483	0.232	0.225	0.575	0.350	0.178	0.529	0.291	0.201
BERT-Base	0.827	0.326	**0.848**	0.940	0.526	**0.721**	0.883	0.426	**0.784**
RoBERTa-Base	**0.913**	**0.490**	0.741	**0.967**	**0.716**	0.569	**0.940**	**0.603**	0.655

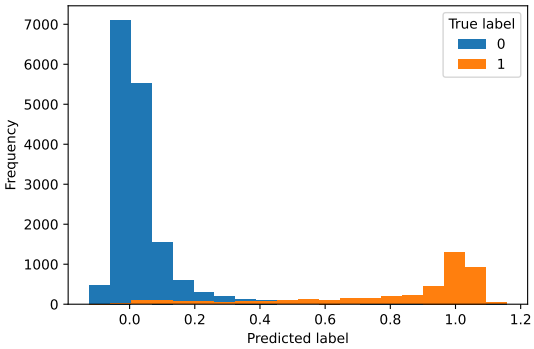
ing and test dataset. The rather simple baseline captures the similarity between an argument and a key point on the token level and might be sensative against more complicated parphrases.

Both fine-tuned matchers outperform the baselines by a large margin. While the BERT-Base matcher achieves higher relaxed mean average precision on the training set than the RoBERTa-Base matcher, the RoBERTa-Base matcher is overall better than BERT, especially with strict labels. The RoBERTa-Base matcher achieves a mean average precision of 0.913 in the strict setting on the test set, and 0.967 in the relaxed setting. As these scores on the test set are nearly as high as on the training set, we argue that RoBERTa is a more robust language model and generalizes better than BERT. Also the RoBERTa-Base matcher performs better in terms of precision, while the BERT-Base matcher is better with respect to recall for all dataset splits in both the strict and relaxed settings.

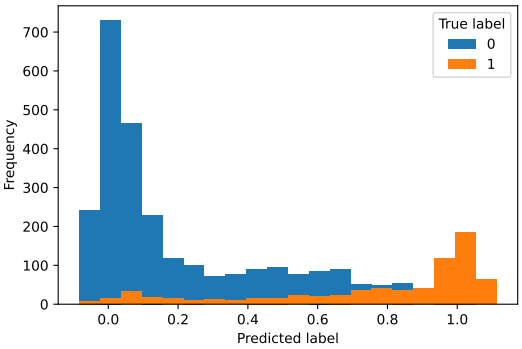
6 Error Analysis

To find errors of the two trained matchers, BERT-Base and RoBERTa-Base, in Figure 2 we show histograms of predicted match scores with respect to ground-truth labels. Both matchers classify most pairs correctly, which can be seen because the histogram spikes around 0 for the true `no-match` label and around 1 for the true `match` label. We also observe that predictions on the training set are closer to the true label than on the development set for both RoBERTa-Base and BERT-Base. Even though we expect any machine-learned matcher to perform better on training data than on validation data, we see this as a room for improvement with better generalization. We notice that in Figures 2a and 2c both approaches predict non-matching argument key point pairs better than matching key points. This effect is likely to occur because of the higher amount of non-matching pairs provided in the training dataset. Most arguments match with only a few or even just a single key point. But nonetheless each argument is compared to all other key points; hence, the underlying data to learn from is imbalanced (Barandela et al., 2004). Even though experiments with using textual data augmentation or simple oversampling to balance the dataset were unsuccessful (Dietterich, 1995), more advanced oversampling or undersampling approaches could possibly resolve this issue. We further identify that the predicted matching scores of BERT-Base are spread a bit more than scores from RoBERTa-Base.

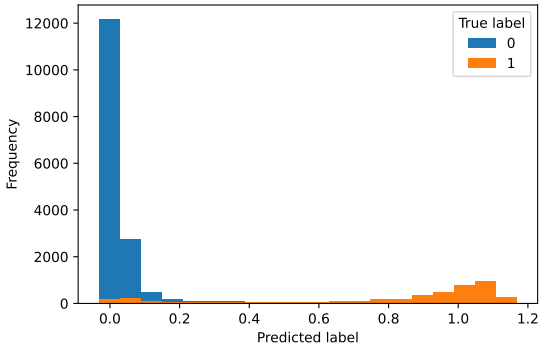
In Figure 2b, we observe that the BERT-Base matcher falsely predicts certain non-matching pairs



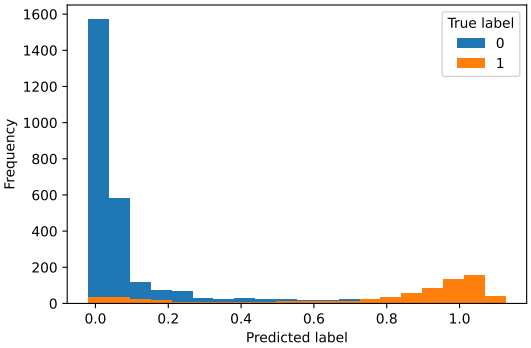
(a) Predictions with BERT-Base on the training set.



(b) Predictions with BERT-Base on the validation set.



(c) Predictions with RoBERTa-Base on the training set.



(d) Predictions with RoBERTa-Base on the validation set.

Figure 2: Histograms of predicted labels on the training and validation sets for argument key point pairs with the BERT-Base and RoBERTa-Base classifiers. For good classifiers, predicted labels should approximately equal the true label (0 or 1).

Table 3: Examples of argument key point pairs from the ArgKP-2021 dataset (Bar-Haim et al., 2020a) where the predicted score is off the ground truth label (True) with either the BERT-Base or RoBERTa-Base matcher.

#	Argument	Key point	True	BERT	RoBERTa
D	School uniforms can be less comfortable than students' regular clothes.	School uniforms are expensive	0	0.48	0.03
E	affirmative action discriminates the majority, preventing skilled workers from gaining employment over someone less qualified but considered to be a member of a protected minority group.	Affirmative action reduces quality	1	-0.05	0.03

with scores around 0.5. An example of such uncertain pair is the argument key point pair D from Table 3. This argument which is in the training dataset has no matching key points. For this argument, the BERT-Base matcher has not learned well how to classify matches for that type of argument, and therefore predicts a neutral score of 0.48. However, the RoBERTa-Base matcher does not make that error.

Both, the BERT-Base matcher and RoBERTa-Base falsely predict some argument key point pairs as `no-match` that are in fact labelled as a `match`. For example, it seems to be difficult to predict a `match` for the argument key point pair E from Table 3. The argument from the example is longer than most arguments and especially much longer than the key point (431 % more characters). It might be more challenging to reduce such longer arguments, that contain more complex information, to very compact key points. We confirm that observation by comparing the squared classification error with respect to the absolute difference between argument and key point lengths.

7 Conclusion and Future Work

We approach the practical problem of matching arguments with short key points with the goal of summarizing arguments. Although our token overlap baseline approach is very simple, it achieves a mean average precision of up to 0.575 on the test set, nearly double the score of a random matcher. The baseline approach is straightforward to implement but can not eliminate the problem of context understanding. RoBERTa-Base and BERT-Base have achieved good performance, because they can overcome the context understanding challenge. Our fine-tuned RoBERTa-Base model also performed better than BERT-Base in this task and scores a mean average precision of up to 0.967. With strict ground truth labels it achieves a mean average precision score of 0.913 on the test set, which is the best score of the participating teams in the shared task. This again shows the importance of architecture, training objectives, and hyperparameter selection.

7.1 Future Work

In Section 3, we observed that transformer models tend to misclassify argument key point pairs if the argument and key point largely differ in length. As an extension to our approach, we propose to combine transformer models with the overlap baseline in an ensemble. Another possible improvement are recent improvements in language models (Sun et al., 2021). If a language model is even more robust than, for example, RoBERTa, we expect a fine-tuned matcher to outperform the RoBERTa-Base matcher as well.

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 2915–2925.

Roy Bar-Haim, Lilach Eden, Roni Friedman, Yoav Kantor, Dan Lahav, and Noam Slonim. 2020a. From arguments to key points: Towards automatic argument summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 4029–4039. Association for Computational Linguistics.

Roy Bar-Haim, Yoav Kantor, Lilach Eden, Roni Friedman, Dan Lahav, and Noam Slonim. 2020b. Quantitative argument summarization and beyond: Cross-domain key point analysis. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 39–49. Association for Computational Linguistics.

Ricardo Barandela, Rosa Maria Valdovinos, José Salvador Sánchez, and Francesc J. Ferri. 2004. The imbalanced training sample problem: Under or over sampling? In *Structural, Syntactic, and Statistical Pattern Recognition, Joint IAPR International Workshops, SSPR 2004 and SPR 2004, Lisbon, Portugal, August 18-20, 2004 Proceedings*, volume 3138 of *Lecture Notes in Computer Science*, page 806. Springer.

Steven Bird and Edward Loper. 2004. NLTK: the natural language toolkit. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, Barcelona, Spain, July 21-26, 2004 - Poster and Demonstration*. ACL.

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.

Thomas G. Dietterich. 1995. Overfitting and undercomputing in machine learning. *ACM Comput. Surv.*, 27(3):326–327.

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.

Roni Friedman, Lena Dankin, Yoav Katz, Yufang Hou, and Noam Slonim. 2021. Overview of kpa-2021 shared task: Key point based quantitative summarization. In *Proceedings of the 8th Workshop on Argumentation Mining*. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

John Lawrence and Chris Reed. 2019. Argument mining: A survey. *Comput. Linguistics*, 45(4):765–818.

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.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. *CoRR*, abs/1711.05101.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

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 and Graeme Hirst. 2017. Classifying Frames at the Sentence Level in News Articles. In *Proceedings of Recent Advances in Natural Language Processing*, pages 536–542.

Martin F Porter. 1980. An algorithm for suffix stripping. *Program*.

Nils Reimers, Benjamin Schiller, Tilman Beck, Johannes Daxenberger, Christian Stab, and Iryna Gurevych. 2019. Classification and clustering of arguments with contextualized word embeddings. *arXiv preprint arXiv:1906.09821*.

Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, Weixin Liu, Zhihua Wu, Weibao Gong, Jianzhong Liang, Zhizhou Shang, Peng Sun, Wei Liu, Xuan Ouyang, Dianhai Yu, Hao Tian, Hua Wu, and Haifeng Wang. 2021. ERNIE 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. *CoRR*, abs/2107.02137.

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.