Team Hitachi @ AutoMin 2021: Reference-free Automatic Minuting Pipeline with Argument Structure Construction over Topic-based Summarization

Atsuki Yamaguchi[*], *Gaku Morio*[*], *Hiroaki Ozaki*[*], *Ken-ichi Yokote and Kenji Nagamatsu*

Research and Development Group
Hitachi, Ltd.
Kokubunji, Tokyo, Japan

{atsuki.yamaguchi.xn, gaku.morio.vn, hiroaki.ozaki.yu, kenichi.yokote.fb, kenji.nagamatsu.dm}@hitachi.com

Abstract

This paper introduces the proposed automatic minuting system of the Hitachi team for the First Shared Task on Automatic Minuting (AutoMin-2021). We utilize a reference-free approach (i.e., without using training minutes) for automatic minuting (Task A), which first splits a transcript into blocks on the basis of topics and subsequently summarizes those blocks with a pre-trained BART model fine-tuned on a summarization corpus of chat dialogue. In addition, we apply a technique of argument mining to the generated minutes, reorganizing them in a well-structured and coherent way. We utilize multiple relevance scores to determine whether or not a minute is derived from the same meeting when either a transcript or another minute is given (Task B and C). On top of those scores, we train a conventional machine learning model to bind them and to make final decisions. Consequently, our approach for Task A achieve the best adequacy score among all submissions and close performance to the best system in terms of grammatical correctness and fluency. For Task B and C, the proposed model successfully outperformed a majority vote baseline.

Index Terms: automatic minuting, summarization, argument mining, topic segmentation

1. Introduction

This paper introduces the proposed automatic minuting system of the Hitachi team for the First Shared Task on Automatic Minuting (AutoMin-2021) [1]. The shared task consists of a main task (Task A) and two subtasks (Task B and C). Task A aims at generating minutes from meeting transcripts. Task B involves identifying whether or not a given minute belongs to a given transcript. Similar to Task B, Task C requires a model to identify whether or not two given minutes belong to the same or different meeting(s).

Our initial approach for Task A was to build a deep reinforcement learning-based end-to-end system [2] using the provided reference minutes. The system consists of a pick-up module and a summarization module. The former selects important utterances from transcripts. We also expected that this module could be useful for Task B and C to measure the relevance. The latter summarizes utterances that are selected by the pick-up module. Unfortunately, this approach did not perform well in our preliminary experiments. Through an error analysis, we found that (1) sentences in minutes did not always have their corresponding utterances in transcripts; (2) sentences in minutes highly depended on each annotator's aspect, and (3) reference resolution (resolving who "I" is or who"you" is, etc.) was necessary because minutes are often written from a third-person perspective, while the transcripts are often from a first-person perspective. These (1) and (2) observations also made solving Task B and C difficult. Consequently, we figured out that these "ground-truth" minutes did not contribute to summarizing transcripts.

Hence, we subsequently adopt a reference-free approach for Task A and did not utilize reference minutes to generate summaries given that it is challenging for a machine learning model to learn the relationship between a summary and a transcript due to (1) and (2). For Task B and C, we employ a conventional feature-based machine learning method to make classifications.

Our pipeline for Task A mainly consists of segmentation, summarization, and argument mining modules. The segmentation module is designed to extract utterances by *block*, the utterances of which should mention the same topic in a transcript. At the same time, this module can filter out irrelevant utterances by excluding them from a block. We build a Longformer-based segmentation module trained with our manually annotated data, which are sampled from English training data. Surprisingly, the topic boundaries are well matched between our two annotators with an average agreement ratio of 0.818. Given an utterance block, the summarization module generates its summary. To achieve this, we utilize a pre-trained BART model [3] fine-tuned on the SAMSum corpus [4]. Because the model can partially solve reference problem (3) through the segmentation and summarization modules, we can mostly have a comprehensive summarization of an input transcript. Finally, we structure the summarized blocks and formulate a resulting minute with an off-the-shelf argument mining parser [5]. This module is derived from our observation that most of the reference minutes are structured by itemization. Although a lot of variety exists in the formats, aspects, and perspectives of the structures mentioned in (2), we hypothesize that any coherent structures would improve readability and enable readers to recognize certain aspects and perspectives of the minutes. To compose such structures in a minute, we use the argument mining parser, which can predict argumentative labels and reasons for each sentence. In addition, our system can also handle transcripts in Czech. We simply add an mBART [6] translation module that converts an input transcript from Czech into English and also changes its output minute from English to Czech.

Our experimental results on the English test set show that our automatic minuting system achieves the best adequacy score among all submissions, indicating that it can generate a minute with major topics covered in each meeting. In addition, our system exhibits close scores to the best system in terms of

* Equal contribution

grammatical correctness and fluency.

For Task B and C, we define surface-oriented features rather than semantics-oriented ones. We first define relevance scores, such as tf-idf [7] cosine similarity, named entity overlap ratio, date consistency, and BERTScore [8]. On top of the relevance scores, we trained machine learning models, such as support vector machine (SVM) [9], logistic regression [10], and random forest [11]. We pick up the best performing model from them to make a final classification. The results demonstrate that our approach outperforms a majority vote baseline, showing its effectiveness in classifying minutes.

2. Task A: Automatic Minuting

2.1. Overview

We propose a module-based approach to generate minutes automatically from transcripts. Figure 1 shows an overview of our approach. It mainly consists of the following three modules: segmentation, summarization, and argument mining. We first split an input transcript into topic-specific blocks using a Longformer-based model [12]. Then we summarize texts in each block with BART [3] fine-tuned on the SAMSum [4], a summarization corpus of chat dialogue. Finally, we structure the summarized blocks and formulate a resulting minute with an argument mining parser. Note that our approach is applicable to transcripts written in both English and Czech by utilizing a pre-trained mBART model.

2.2. Segmentation Module

Given that state-of-the-art neural-network-based summarization models cannot process a long transcript at a time, it must be split into some segments so that the models can accommodate them. Our method segments such transcripts into blocks by topic, allowing us to generate a summary per topic in the later stage.

2.2.1. Approach

We utilize a sequence labeling task (BIO-tagging) to split a transcript into topic-specific blocks and employ a pre-trained $\text{Longformer}_{\text{LARGE}}$ model followed by a bidirectional Long Short-Term Memory (LSTM) [13] to solve the task. For each transcript, we first split it into sentences while inserting a special `[SEP]` token at the beginning of each sentence, followed by tokenization with a pre-trained Longformer tokenizer. Then we feed the tokenized transcript into the Longformer+LSTM model and predict a BIO tag for each `[SEP]` token. The motivation behind the use of Longformer and LSTM is that (i) Longformer can handle long sequences unlike other conventional Transformer-based models [14]; (ii) LSTM can force the model to learn the sequential nature of the task that each label is dependent on its adjacent labels.

More formally, given a transcript $\mathcal{T}$ composed of n sentences and its corresponding topic BIO labels, we first split $\mathcal{T}$ into sentences $S_{\mathcal{T}} = \{s_1, \dots, s_n\}$, while adding a `[SEP]` token at the beginning of each sentence. We then tokenize them using a pre-trained Longformer tokenizer and obtain their token-level representation $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_m)$. We feed $\mathbf{X}$ into the $\text{Longformer}_{\text{LARGE}}$ model and pick up its output representation of the `[SEP]` tokens $\mathbf{H}_{\mathcal{T}} = (\mathbf{h}_1, \dots, \mathbf{h}_n), \mathbf{h}_i \in \mathbb{R}^D$, where D corresponds to the dimensionality of each hidden layer in Longformer. $\mathbf{H}_{\mathcal{T}}$ is subsequently passed to a one-layer bidirectional LSTM, which yields $\mathbf{H}'_{\mathcal{T}} = (\mathbf{h}'_1, \dots, \mathbf{h}'_n), \mathbf{h}'_i \in \mathbb{R}^{2\times D}$. Finally, we put $\mathbf{H}'_{\mathcal{T}}$ into a final output linear layer and obtain $\mathbf{O} = (\mathbf{o}_1, \dots, \mathbf{o}_n)$, where $\mathbf{o}_i \in \mathbb{R}^3$ is the i-th output of $S_{\mathcal{T}}$. The task is trained with the token-level cross-entropy loss averaged over the `[SEP]` tokens:

$$\mathcal{L} = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{3} y_{ij}\log p_{ij}(s_i),$$

where $p_{ij}(s_i)$ represents the probability of the i-th input sentence s_i predicted as *beginning of a topic* ($j = 1$), *inside a topic* ($j = 2$), or *outside of a topic* ($j = 3$) by the Longformer+LSTM model. y_{ij} is the corresponding target label.

2.2.2. Topic Boundary Annotation

We manually annotated ten different transcripts in English training data, of which we utilized nine transcripts for training and one for validation. We assigned topic boundary labels per sentence because the Longformer+LSTM model predicts topic BIO labels by sentence. We defined the average agreement ratio between annotators as follows: Let $\mathcal{B}_1 : \{b_{1,1}, \dots\}$ be a set of all blocks given by annotator 1 and $\mathcal{B}_2 : \{b_{2,1}, \dots\}$ be a set of all blocks given by annotator 2, where b represents a block $[s_i, \dots, s_j]$. Then the agreement rate ($a_{1,2}$) is given by

$$a_{1,2} = \frac{1}{|\mathcal{B}_1|}\sum_{b_1\in\mathcal{B}_1}\max_{b_2\in\mathcal{B}_2}\frac{|b_1 \cap b_2|}{|b_1|}.$$

The average of $a_{1,2}$ and $a_{2,1}$ for the validation transcript[1] was 0.818.

2.2.3. Implementation Details

We implemented the segmentation model using PyTorch [15] and the `transformers` library [16]. We fine-tuned the model for 100 epochs with a batch size of 1 and utilized the best weights for inference that achieved a validation accuracy of 0.850 at 510 steps. We set the learning rate to 10^{-5} and utilized the Adam optimizer [17] with a linear warmup of the first 5% of steps and a gradient accumulation every five steps. As pre-processing, if the total number of tokens in $\mathcal{T}$ exceeded the maximum length of input tokens for Longformer $L_{max} = 4096$, we created chunks that can hold up to L_{max} tokens and put sentences in each chunk with a stride of 1024 tokens. We then fed each chunk into the segmentation model and obtained its predicted BIO labels. To concatenate predicted labels from multiple chunks, we prioritized the predicted label(s) of a prior chunk whenever prediction results were duplicated between the two chunks.

2.3. Summarization Module

The summarization module generates a summary for each block with a pre-trained language model. Specifically, we use a pre-trained BART model fine-tuned on CNN/DailyMail [18, 19] and the SAMSum corpus, which is a large corpus of chat dialogue consisting of over 16k samples with abstractive summaries. With the pre-trained model, we can generate a concise and fluent summary given a dialogue block. Because each summary in the SAMSum corpus extracts essential information in its dialogue, is written in the third person, and includes the names of speakers, we expect that a summarization model trained with the corpus should be able to resolve references.

[1] We used `en_train_009.txt` as our validation transcript.

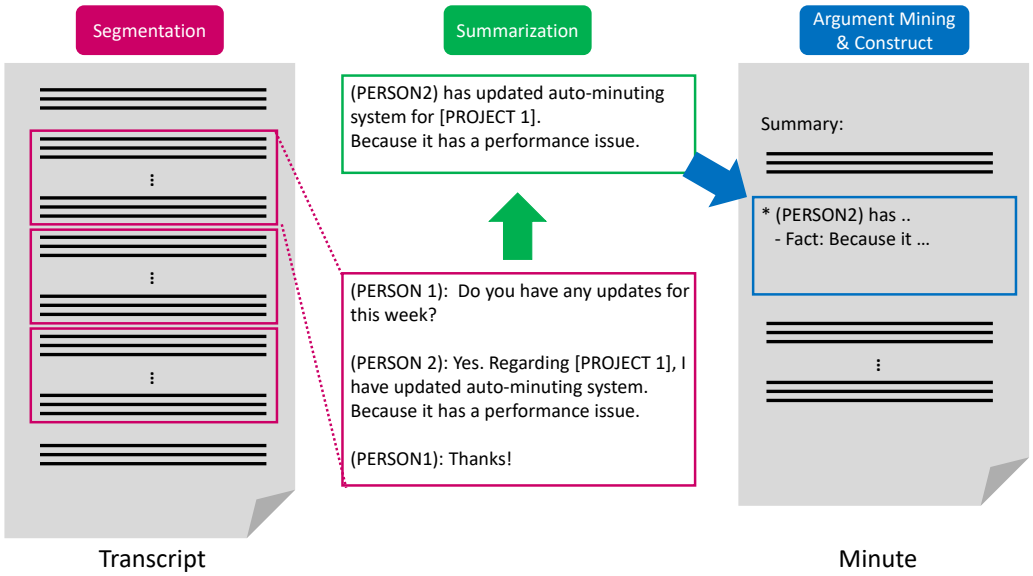


Figure 1: *Overview of our approach. We first split a transcripts into blocks and then summarize them using a pre-trained model. Finally, we compose all summaries into minutes with a technique of argument mining.*

2.3.1. Implementation Details

We employed a pre-trained $\text{BART}_{\text{LARGE}}$ model fine-tuned on the CNN/DailyMail and SAMSum datasets as our summarization model.[2] The weights were downloaded via the `transformers` library. For pre-processing, we added a speaker tag, such as "PERSON1:", to the beginning of each utterance as a prefix. If the number of tokens in a block exceeded the maximum length for BART $L_{max} = 1024$, we truncated some utterances in the block in advance to avoid generating incomplete sentences. During post-processing, we replaced some misspelled or unnatural tokens generated by the model with correct ones in accordance with pre-defined regular expression rules to make the generated summary more consistent.

2.4. Argument Mining Module

This module creates a structured minute from summarized texts. We use a concept of argument mining [20], an attractive research area that focuses on mining arguments such as reasons behind a claim. We provide an off-the-shelf argument mining parser [5] to construct structures. The parser predicts an argument graph based on a concept of the Cornell eRulemaking Corpus (CDCP) [21]. With the concept of the CDCP, we can predict an argumentative label for each proposition and relationship between propositions. In our system, we split the summarized text (i.e., a result from the summarization module in a block) into sentences by spaCy [22], recognizing each sentence as a proposition.

2.4.1. Argument Graph

We predict an argumentative label for each proposition using the argument mining parser. There are three types of proposition labels having the following description.

1. **Task** is a subjective sentence that contains a specific policy (e.g., what should be done). We expect that sentences with this label should include an important topic or discussion aspect. The original label name in CDCP is Policy.
2. **Fact** is an objective sentence. We map Testimony and Fact labels that are originally introduced in CDCP into this label.
3. **Disc** is a subjective sentence such as an opinion and claim. The original label name in CDCP is Value.

In addition to labeling each proposition, we predict argumentative relationships between propositions. The argument mining parser predicts a Reason or Evidence relationship, which is a support relationship between the propositions. We obtain an argument graph by combining both the proposition labels and relationships.

2.4.2. Structured Minute Construction

Our approach to construct the structured minutes (shown in Figure 2) from the predicted argument graph is as follows:

1. We assume that the first sentence in the summarized text and sentences that have a Task label are the *root* item of the structure. We expect these sentences represent a specific topic or aspect of the summarization. The root items are represented as "* ..." as shown on the right in Figure 2.
2. Subsequent sentences for the preceded root item become a child item. For instance, we represent a child item with a predicted proposition label of Fact as "- Fact: ...".
3. A sentence that has an outgoing relationship to a preceded sentence becomes a child item for the preceding sentence. We expect that we can make a supplemental or supporting sentence for its previous sentence clear for readers. For example, in Figure 2, the Fact sentence has an outgoing Reason relationship in the Disc sentence. We represent this by "- - Fact: ...".

[2] Namely `philschmid/bart-large-cnn-samsum` available at `https://huggingface.co/philschmid/bart-large-cnn-samsum`

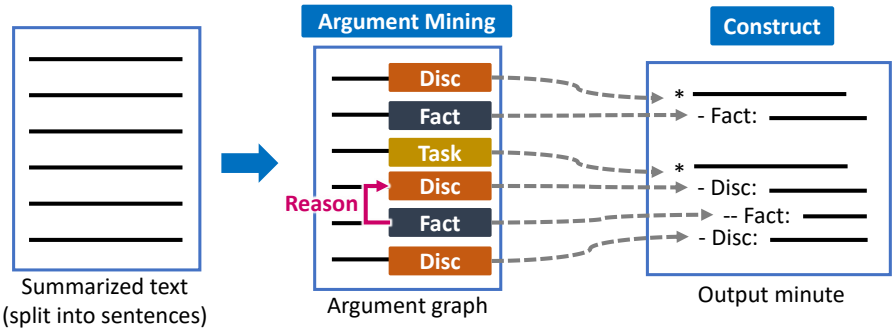


Figure 2: *Overview of the argument mining module. The summarized text in a block is split into sentences. The argument mining parser predicts an argument graph for the sentences. Finally, we structure a minute using itemization to make it coherent and to improve readability. Note that the first sentence in the argument graph and Task sentences should be the root for the itemization. We do not change the order of sentences to prevent corrupting the fluent and coherent summarized text.*

	Language	Adequacy	Grammatical correctness	Fluency	ROUGE 1	ROUGE 2	ROUGE L
Ours	en	**4.25**	4.34	3.93	0.217	0.0595	0.116
Average of all teams		2.81	3.25	2.92	0.203	0.0458	0.114
Best in all teams		4.25	**4.45**	**4.27**	**0.282**	**0.0655**	**0.159**
Ours	cs	2.69	1.25	2.06	0.159	0.0251	0.0694

Table 1: *Task A results on the test set. The adequacy, grammatical correctness and fluency are based on a manual evaluation by two annotators and assessed on a Likert Scale of 1 to 5, where 1 represents the worst and 5 refers to the best. All scores the are averaged across all test set samples and based on official results provided by the organizers.*

2.5. Results

Table 1 shows our submission results of the auto minuting task on the test set. First, we observe that our English system achieved the best adequacy score of 4.25 out of 5 among all submissions, suggesting that our approach can adequately capture major topics appearing in each meeting transcript. Second, our approach also exhibited similar human evaluation scores to the best system by 0.11 for grammatical correctness and 0.34 for fluency and far better scores than the average ones. These results demonstrate that our automatically generated minutes are generally grammatically consistent and have some fluency and coherency. Finally, our ROUGE scores were narrowly better than the average scores, and huge gaps existed between our results and the best ones in terms of the automatic evaluation metrics. These results show a different trend compared to the ones involving the human evaluation.

We further investigate the relationship between human and automatic evaluations. Table 2 shows the correlation coefficients between the human evaluation metrics and the automatic ones. We used average scores for the automatic metrics. Each metric has a weak correlation ranging from 0.42 to 0.544. Instead of averaging the automatic metrics over reference minutes, we took the maximum values among the reference minutes in Table 3. Here, we only used samples with more than two reference minutes. Compared to Table 2, Table 3 shows a strong correlation between adequacy and ROUGE 1. In contrast, grammatical correctness and fluency are less correlated with the automatic metrics, especially on ours. We discuss the relationship between the human evaluation metrics and the automatic ones in §2.6.

For the Czech results, our translation-based approach appeared to have failed to generate natural minutes because all of our Czech scores were far lower than those in English.

	ROUGE 1	ROUGE 2	ROUGE L
Adequacy	0.544	0.496	0.480
Correctness	0.430	0.433	0.451
Fluency	0.494	0.420	0.483

Table 2: *Correlation coefficients between human and automatic evaluations.*

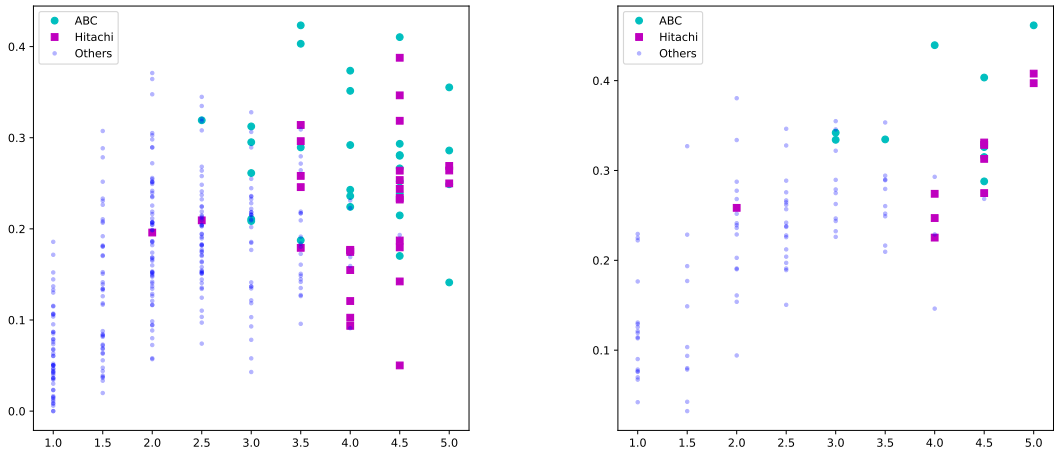
	Data	ROUGE 1	ROUGE 2	ROUGE L
Adequacy	All	0.715	0.580	0.549
	Ours	0.940	0.575	0.804
Correctness	All	0.408	0.333	0.365
	Ours	0.188	-0.218	-0.122
Fluency	All	0.547	0.397	0.486
	Ours	0.195	-0.0132	0.358

Table 3: *Correlation coefficients between human and automatic evaluations. Instead of averaging automatic metrics over reference minutes, we take maximum values among the reference minutes. We only use samples that have more than two reference minutes. "All" indicates that we include all of the participants' data, and "Ours" means only our results were used.*

2.6. Discussion

2.6.1. Automatic vs. Human Evaluations

The comparison between Table 2, 3 and Figure 3 indicates more than two types of adequate minutes in general, and those types are not similar to each other. This supports our hypothesis that the minutes are varied because of each annotator's diverse aspects as mentioned in §1. In addition, the results indicate that adequacy can be evaluated using word overlap between a target minute and the closest reference minute (Table 3). This is



(a) *Average* (b) *Max Pooling for more than two reference minutes*

Figure 3: *Adequacy vs. ROUGE 1 scores. The horizontal axis represents Adequacy of each generated minute for the test set. The vertical axis represents ROUGE 1 scores.*

consistent with the results of Task C, which we describe in the latter section.

Grammatical correctness and fluency were not related to any automatic metrics for ours shown in Table 3, even though our model achieved high performance in the human evaluations. This could be a nature of our reference-free approach. Here we consider the case in which a few variations did not occur in adequate minutes as we mentioned in §1. The generated minutes using the reference-free approach may have grammatically correct and adequate sentences. However, those sentences are very different from the reference ones. In this case, measuring the grammatical correctness and fluency using N-gram based metrics will be difficult.

The high human evaluation scores of grammatical correctness and fluency indicate a merit of our reference-free approach, which already captured grammatical correctness and fluency through pre-training on massive corpora. Compared with training an end-to-end model from scratch, which may require a lot of effort to capture grammatical correctness and fluency, we do not need any exhausting training.

2.6.2. Translation Strategy

In our preliminary experiments, we translated generated English minutes into Japanese (our mother tongue) and found they were acceptable. However, our Czech system did not perform well on the Czech test set (Table 1). According to Liu et al. [6], the translation accuracy of Japanese and Czech is almost the same. We hypothesize that low quality in transcript translations might lead to low scores on the Czech data, because converting transcripts in spoken language is considered to be more difficult than converting documents in written language.

3. Task B and C: Pair Classification

3.1. Overview

Figure 4 shows an overview of our approach. We first define relevancy scores such as tf-idf [7] cosine similarity, named entity overlap ratio, date consistency. and BERTScore. On top of the relevancy scores, we trained machine learning models such as SVM [9], logistic regression [10], random forest [11] and multi-layer perceptron (MLP). We picked up the best performing model from them.

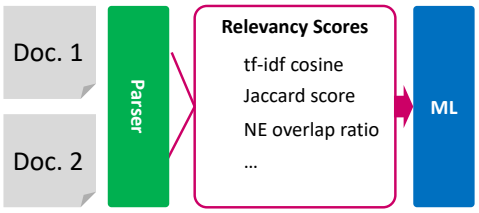


Figure 4: *Overview of our approach for Task B and C. We first compute relevancy scores when two documents are given, then train a conventional machine learning model on top of the scores.*

3.2. Pre-processing

To extract each relevancy score, we first applied a minute parser that can analyze a structure of lines in a minute. The minute parser can find mainly the title, date, attendees, topic(s), subtopics and item lines with their parents or children. We employed a transition-based parsing system [23, 24] as our minute parser. A transition-based parser usually consists of a stack and a buffer that holds unprocessed data. Then, a certain action will be taken in accordance with the contents in the stack and the buffer. We used random forest for the action predictor and trained it on manually annotated data sampled from the English training data. Table 4 lists actions that we defined. Here, b_0 is the top element of the buffer, s_0 is the top element of the stack, and s_1 is the second top element in the stack.

3.3. Features

3.3.1. tf-idf cosine similarity

Let $\mathcal{D}_1$ be an input document. For Task B, $\mathcal{D}_1$ is a transcript. Let $\mathcal{D}_2$ be another input document. For both Task B and C cases, $\mathcal{D}_2$ is a minute. We computed the cosine similarity of tf-idf document vectors [7] of $\mathcal{D}_1$ and $\mathcal{D}_2$.

Before calculating tf-idf, we eliminated stop words using

Action	Argument	Detail
LABEL	Label name (topic, title, date, etc.)	Set label to the b_0 and discard b_0.
ADD	-	Add a parent-child relationship between s_0 and b_0 and add b_0 to the stack.
REPLACE	-	Add a parent-child relationship between s_1 and b_0 and add b_0 to the stack.
ARC	-	Add a parent-child relationship between s_0 and b_0 and discard b_0.

Table 4: *Action set of our transition-based minute parser.*

NLTK[3] as a preprocessing. We also removed some words with fewer than three characters or more than fourteen characters and deleted punctuation and symbol characters. For calculating idf, we regard every sentence in $\mathcal{D}_1$ and $\mathcal{D}_2$ as a single document.

3.3.2. Jaccard Similarity Coefficient

We computed the Jaccard similarity coefficient [25] of vocabularies in $\mathcal{D}_1$ and $\mathcal{D}_2$. We excluded terms in the NLTK stopword list and terms fewer than three characters or more than fourteen characters.

3.3.3. NE Overlap

Here we define a named entity as an anonymized token such as "[PERSON X]," "[PROJECT X]," and "[ORGANIZATION X]." We extracted them with regular expressions. Let $\mathcal{E}$ be a function that extracts a set of all named entities in a given document. Then, the NE overlap score is calculated as

$$\text{NE Overlap} := \frac{|\mathcal{E}(\mathcal{D}_1) \cap \mathcal{E}(\mathcal{D}_2)|}{|\mathcal{E}(\mathcal{D}_1) \cup \mathcal{E}(\mathcal{D}_2)|}.$$

When either $\mathcal{E}(\mathcal{D}_1)$ or $\mathcal{E}(\mathcal{D}_2)$ is $\emptyset$, we define NE overlap as zero.

3.3.4. Date Consistency

We extracted dates with a Python `dateutil.parser`[4] from a "date line." We defined four dimensional features, and then those of dimensions representing whether or not each year, month, day, and hour was consistent between $\mathcal{D}_1$ and $\mathcal{D}_2$.

3.3.5. BERT Score

We use BERTScore [8] to compute the semantic similarity between documents. Because the transcript and minute documents are much longer for BERTScore, we need to split them. Here we split $\mathcal{D}_1$ and $\mathcal{D}_2$ into N chunks, i.e., $(\mathcal{D}_{1,1}, ...\mathcal{D}_{1,N})$ and $(\mathcal{D}_{2,1}, ...\mathcal{D}_{2,N})$. We compute the BERTScore for each chunk pair, i.e., $s(i) = \text{BERTSCORE}(\mathcal{D}_{1,i}, \mathcal{D}_{2,i})$. The averaged score $\frac{1}{N}\sum_{i=1}^{N} s(i)$ serves as the feature value. In this study, we selected $N = 4$.

3.4. Model Selection

We tuned the models with a hyperparameter optimization framework named Optuna [26][5]. Through the optimization, we applied 10-fold cross validation to avoid overfitting. To evaluate the validation score, we merged official dev and fold-out sets. The best hyperparameters for each task are shown in Table 5. Finally, we used an ensemble average of the 10 cross-validation models to predict outputs of the test set.

[3] `https://www.nltk.org/`

[4] `https://dateutil.readthedocs.io/en/stable/parser.html`

[5] `https://www.preferred.jp/en/projects/optuna/`

		Model	Hyperparameter values
Task B	en	SVM	kernel: linear, C: 8.05, γ: 0.00815
	cs	SVM	kernel: linear, C: 9.81, γ: 0.00172
Task C	en	MLP	hidden dim: 64, layer: 3
	cs	SVM	kernel: rbf, C: 2.87, γ: 2.11

Table 5: *Hyperparameters for the best models of our Task B and C submissions.*

		Model	Accuracy	Precision	Recall	F1
Task B	en	Ours	**0.977**	0.735	0.926	0.820
		Majority	0.944	-	-	-
	cs	Ours	**0.957**	0.905	0.633	0.745
		Majority	0.900	-	-	-
Task C	en	Ours	**0.939**	0.509	0.934	0.659
		Majority	0.936	-	-	-
	cs	Ours	**0.984**	0.846	0.971	0.904
		Majority	0.922	-	-	-

Table 6: *Overall results for our Task B and C submission.*

3.5. Results

Table 6 shows the evaluation results on the test dataset. We compare them with majority vote baselines, which always give FALSE for all data points. All our submitted models outperformed the baselines with high F1 scores ranging from 0.659 to 0.904. The task C English model only had a slightly better result than the baseline. This may have been due to the MLP model because it was prone to overfitting our hand-crafted features.

In our preliminary experiments, features based on term overlaps (i.e., tf-idf and NE) contributed to the models' performance well. This is in line with the results of the high correlation between ROUGE 1 and the adequacy scores.

4. Conclusion

In this paper, we have introduced the reference-free automatic minuting system and the document classification system using various surface-oriented features for the First Shared Task on Automatic Minuting (AutoMin-2021). Given that reference minutes for Task A reflect diverse annotators' aspects, we aimed at building a reference-free pipeline rather than mimicking reference minutes. Thus, we adopted a module-based approach, which first splits a transcript into topic-specific blocks and subsequently summarizes those blocks with a pre-trained BART model fine-tuned on the SAMSum corpus. Motivated by the assumption that any coherent structure would improve readability, we further applied a technique of argument mining to the generated minutes, tailoring them in a well-structured and coherent way. Our approach achieved the best adequacy score among all submissions and showed competitive performance with the best system in terms of two human evaluation metrics: grammatical correctness and fluency. In addition, our approach demonstrated

moderate ROUGE scores despite the reference-free approach.

To determine whether or not a minute is derived from the same meeting when either a transcript or another minute is given (Task B and C), we utilize multiple surface-based relevance scores. On top of those scores, we trained a conventional machine learning model, such as SVM, to bind them and made final decisions. Consequently, our approach demonstrated better results than the majority vote baselines. The results also suggest that some term overlapping-based features can be utilized as a useful evaluation metric to measure similarities between generated and reference minutes instead of ROUGE scores.

For future work, we intend to improve both grammatical correctness and fluency by paying more attention to the summarization module. In addition, we will update our multilingual pipeline by referring to gold minutes with the best ROUGE 1 or task C scores as proxy variables of the adequacy scores.

5. References

[1] T. Ghosal, O. Bojar, M. Singh, and A. Nedoluzhko, "Overview of the first shared task on automatic minuting (automin) at interspeech 2021," in *Proceedings of the First Shared Task on Automatic Minuting at Interspeech 2021*, 2021, pp. 1–25. [Online]. Available: http://dx.doi.org/10.21437/AutoMin.2021-1

[2] Y.-C. Chen and M. Bansal, "Fast abstractive summarization with reinforce-selected sentence rewriting," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 675–686. [Online]. Available: https://aclanthology.org/P18-1063

[3] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, Jul. 2020, pp. 7871–7880. [Online]. Available: https://aclanthology.org/2020.acl-main.703

[4] B. Gliwa, I. Mochol, M. Biesek, and A. Wawer, "SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization," in *Proceedings of the 2nd Workshop on New Frontiers in Summarization*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 70–79. [Online]. Available: https://www.aclweb.org/anthology/D19-5409

[5] G. Morio, H. Ozaki, T. Morishita, Y. Koreeda, and K. Yanai, "Towards better non-tree argument mining: Proposition-level biaffine parsing with task-specific parameterization," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, Jul. 2020, pp. 3259–3266. [Online]. Available: https://www.aclweb.org/anthology/2020.acl-main.298

[6] Y. Liu, J. Gu, N. Goyal, X. Li, S. Edunov, M. Ghazvininejad, M. Lewis, and L. Zettlemoyer, "Multilingual denoising pre-training for neural machine translation," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 726–742, 2020. [Online]. Available: https://aclanthology.org/2020.tacl-1.47

[7] C. Sammut and G. I. Webb, Eds., *TF–IDF*. Boston, MA: Springer US, 2010, pp. 986–987. [Online]. Available: https://doi.org/10.1007/978-0-387-30164-8_832

[8] T. Zhang*, V. Kishore*, F. Wu*, K. Q. Weinberger, and Y. Artzi, "Bertscore: Evaluating text generation with bert," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=SkeHuCVFDr

[9] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[10] C. Sammut and G. I. Webb, Eds., *Logistic Regression*. Boston, MA: Springer US, 2010, pp. 631–631. [Online]. Available: https://doi.org/10.1007/978-0-387-30164-8_493

[11] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[12] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The long-document transformer," *CoRR*, vol. abs/2004.05150, 2020. [Online]. Available: https://arxiv.org/abs/2004.05150

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, p. 1735–1780, Nov. 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[15] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035.

[16] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. [Online]. Available: https://www.aclweb.org/anthology/2020.emnlp-demos.6

[17] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR (Poster)*, 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[18] R. Nallapati, B. Zhou, C. dos Santos, Ç. Gulçehre, and B. Xiang, "Abstractive text summarization using sequence-to-sequence RNNs and beyond," in *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 280–290. [Online]. Available: https://aclanthology.org/K16-1028

[19] K. M. Hermann, T. Kočiský, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, p. 1693–1701.

[20] J. Lawrence and C. Reed, "Argument mining: A survey," *Computational Linguistics*, vol. 45, no. 4, pp. 765–818, Dec. 2019. [Online]. Available: https://aclanthology.org/J19-4006

[21] J. Park and C. Cardie, "A corpus of eRulemaking user comments for measuring evaluability of arguments," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. Miyazaki, Japan: European Language Resources Association (ELRA), May 2018. [Online]. Available: https://aclanthology.org/L18-1257

[22] M. Honnibal, I. Montani, S. Van Landeghem, and A. Boyd, "spaCy: Industrial-strength Natural Language Processing in Python," 2020. [Online]. Available: https://doi.org/10.5281/zenodo.1212303

[23] H. Yamada and Y. Matsumoto, "Statistical dependency analysis with support vector machines," in *Proceedings of the Eighth International Conference on Parsing Technologies*, Nancy, France, Apr. 2003, pp. 195–206. [Online]. Available: https://aclanthology.org/W03-3023

[24] J. Nivre, J. Hall, J. Nilsson, G. Eryiğit, and S. Marinov, "Labeled pseudo-projective dependency parsing with support vector machines," in *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*. New York City: Association for Computational Linguistics, Jun. 2006, pp. 221–225. [Online]. Available: https://aclanthology.org/W06-2933

[25] L. Hamers, Y. Hemeryck, G. Herweyers, M. Janssen, H. Keters, R. Rousseau, and A. Vanhoutte, "Similarity measures in scientometric research: The jaccard index versus salton's cosine formula," *Inf. Process. Manage.*, vol. 25, no. 3, p. 315–318, May 1989. [Online]. Available: https://doi.org/10.1016/0306-4573(89)90048-4

[26] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '19. New York, NY, USA: ACM, 2019, pp. 2623–2631. [Online]. Available: http://doi.acm.org/10.1145/3292500.3330701