An Empirical Analysis of Text Summarization Approaches for *Automatic Minuting*

Muskaan Singh, Tirthankar Ghosal, and Ondřej Bojar
Institute of Formal and Applied Linguistics,
Faculty of Mathematics and Physics,
Charles University, Czech Republic
(last-name)@ufal.mff.cuni.cz

Abstract

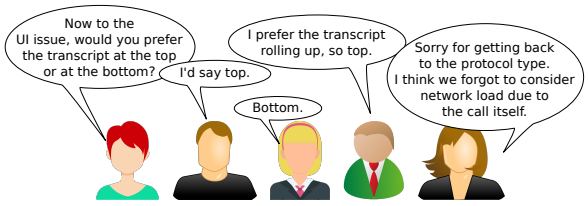
A significant portion of the working population has their mainstream interaction virtually these days. Meetings are being organized and recorded daily in volumes likely exceeding what can be ever comprehended. With the deluge of meetings, it is important to identify and jot down the essential items discussed in the meeting, usually referred to as the *minutes*. The task of minuting is diverse and depends on the goals, style, procedure, and category of the meeting. *Automatic Minuting* is close to summarization; however, not exactly the same. In this work, we evaluate the current state-of-the-art summarization models for automatically generating meeting minutes. We provide empirical baselines to motivate the community to work on this very timely, relevant yet challenging problem. We conclude that off-the-shelf text summarization models are not the best candidates for generating minutes which calls for further research on meeting-specific summarization or minuting models. We found that Transformer-based models perform comparatively better than other categories of summarization algorithms; however, they are still far from generating a good multi-party meeting summary/minutes. We release our experimental code at https://github.com/ELITR/Minuting_Baseline_Experiments.

1 Introduction

With the world adapting to the *new normal* in the pandemic and virtual interactions going mainstream, meeting are held and recorded daily in volumes likely exceeding what can be ever perceived. With the deluge of meetings, it is essential to record the key points of the discussions during the meeting to take stock and identify action items for the future, usually referred to as the *minutes* (see Figure 1). However, not all meetings have the same goal. Some are general meetings, some are topic-focused, while some are informal. According to a certain study,[1] there are six major categories of working meetings: status update, information sharing, decision making, problem-solving, innovation, and team-building meetings. Each meeting has a different set of agenda items and objectives expected to appear in its minutes.

To deal with the flooded information from multiple meetings, which sometimes results in severe cognitive overload, it is essential to provide minutes of the meeting to the participants. Without a meaningful note-taking scribe, it is challenging to correctly remember the contents of a meeting, even for the participants. Not only to the participants, but minutes also help the non-participants (e.g., absentees) to quickly understand what was being discussed, decisions-made, or action items proposed. However, the task is not straightforward, it is sometimes difficult even for meeting participant to take notes on the fly. With the great progress of NLP in almost all areas of speech and text processing, an automatic minuting assistant would be a valuable addition to the meeting workflow. However, the task of *Automatic Minuting* is challenging due to a variety of other reasons, which include: comprehending the goal of the meeting, identifying the crux of the discussion while

[1] http://meetingsift.com/the-six-types-of-meetings/



Original agenda as prepared by the organizer beforehand:
- Protocol type: push or pull?
- Layout of the user interface:
  - Transcript grows at the top or bottom of the document?
  - Or in a side pane?

Shared document, everyone allowed to edit.
Starts with the agenda and gets populated by Automatic Minuting AM
- Protocol type: push or pull?
  AM > Pull easier to implement.
  AM > Updates can get lost with push *in case the user*
  AM > Consider network load.
- Layout of the user interface:
  - Transcript grows at the top or bottom of the document?
    AM > Top AM > Bottom AM > Top, transcript rolling up.
  - Or in a side pane?

Transcript, optionally editable to correct ASR errors:
11:03 Sorry for getting back to the protocol type. I think we forgot ...
11:02 I prefer the transcript rolling up, so top.
11:02 Bottom
...

Figure 1: A Proposal for Automatic Minuting

eliminating small talk and redundancies, identifying topics and drifts, etc.

1.1 Relation of Text Summarization and Automatic Minuting

Automatic minuting is close to text summarization, but the goals of these two tasks are somewhat different. *Text summarization* intends to sum up the central concepts of the text, preserving fluency and coherence of the output summary, while *minuting* is a kind of a slot-filling task; it is motivated more towards topical coverage and churning out action points. Thus, the resulting minutes are expected to contain a bulleted list where fluency or coherence may be less critical. An example highlighting the subtle difference between a meeting summary and a minute is in Figure 2. Comprehending multi-party dialogues is itself challenging, so is automatically producing a text summary. Hence, the problem grows more intense when these two problems come together.

1.2 Contribution

Our work is an attempt towards this complex task of automatic minuting while exploring the performance of existing state-of-art text summarization techniques. Our contributions in this work are:

- We implement 13 different summarization methods (extractive, abstractive) an test them on

(A) Meeting Transcript segment:
ME: . . . I've done some research. We have we have been doing research in a usability lab where we observed users operating remote controls. we let them fill out a questionnaire. Remotes are being considered ugly.and an additional eighty percent indicated that they would spend more money on a fancy-looking remote control. Fifty percent of the people indicated they only loo used about ten percent of the buttons on a remote control . . .
ID: I've got a presentation about the working design. first about how it works. It's really simple. Everybody knows how a remote works. The user presses a button. The remote determines what button it is, uses the infrared to send a signal to the TV ... they only use about ten percent of the buttons, we should make very few buttons . . .
UI: But Got many functions in one remote control, you can see, this is quite simple remote control. few buttons but This re remote control got a lot of buttons. people don't like it, so what I was thinking about was keep the general functions like they are.
PM: Extra button info. that should be possible as. let's see what did we say. More. Should be fancy to, fancy design,easy to learn. Few buttons, we talked about that. Docking station, LCD. general functions And default materials. . . And we have to be very attent in putting the corporate image in our product. So it has to be visible in our design, in the way our device works. . .
PM: . . . I will put the minutes in the project document folder. . . And we have a lunch-break now.

(B) Meeting minutes segment:
- Discussion about the research performed on usability of remote controls and talked about the docking station, LCD, and general functions.
- Eighty percent indicated that they would spend more money on a fancy-looking remote control while ten percent use very few buttons.
- Working of a remote was explained and decided to make few buttons.
- It should have a fancy design which is easy to learn with few buttons on the right places.
- A lot of functions of the remote control should be put in a simple manner.
- Pricing needs to be decided and should be a great deal to people. Survey indicated that an LCD screen in the remote control would be preferred.

(C) Meeting summarization segment:
The Project Manager stated the agenda and the marketing expert discussed what functions are most relevant on a remote,what the target demographic is, and what his vision for the appearance of the remote is.The Marketing Expert also brought up the idea to include a docking station to prevent the remote from getting lost and the idea to include an LCD screen.The User Interface Designer pushed for a user interface with large buttons, a display function, a touchscreen, and the capability of controlling different devices.The team then discussed teletext, the target demographic, the buttons the remote should have, the idea of marketing a remote designed for the elderly, an audio signal which can sound if the remote is lost, LCD screens, and language options · · · whether to include teletext in the design despite the new requirement which indicates that the team is not to work with teletext.The buttons are generally used, but the main feature is ugly and ugly.The remote will only have a few buttons.The remote will feature a small LCD screen.The remote will have a docking station.

Figure 2: A meeting of AMI dataset with (a) transcript, (b) minutes, and (c) summarization. Notations: PM -project manager, ME -marketing expert, ID -industrial designer, UI -user interface designer are roles of the speakers.

three different meeting datasets: AMI (Mccowan et al., 2005) and ICSI (Zechner, 2001) and AutoMin[2].

- We evaluate the output minutes using five automatic evaluation metrics along with expert, crowd-sourced human evaluations on criteria like adequacy, coverage, fluency, and grammaticality.

2 Related Work

Text and speech summarization are widely popular NLP tasks, and there is a lot of literature describing their methods and results. However, in this work, we focus on summarizing multi-party dialogues in a meeting setup, and for this task, the amount of prior work is not so extensive.

The majority of the existing meeting summarization experiments are conducted on the AMI (Carletta, 2007) or ICSI corpus (Janin et al., 2003a). In our work, we do not provide a novel method for the task; instead, we evaluate the performance of text summarization methods to attempt the novel task of *Automatic Minuting*. Most of the prior work in summarization are on newspaper texts (Rush et al., 2015; Chopra et al., 2016; Nallapati et al., 2016; See et al., 2017; Celikyilmaz et al., 2018; Chen and Bansal, 2018; Zhong et al., 2019; Xu and Durrett, 2019; Liu and Lapata, 2019; Lebanoff et al., 2019; Cho et al., 2019; Wang et al., 2020; Xu et al., 2019; Jia et al., 2020) using the standard CNN-daily mail (Hermann et al., 2015) or Newsroom (Grusky et al., 2018) corpora.

Although comparatively lesser, meeting summarization is explored in the works of Chen et al. (Chen and Metze, 2012), Wang et al. (Wang and Cardie, 2013). Some investigations to generate meeting summaries explore with leveraging entailment graphs and ranking strategy by (Mehdad et al., 2013),decisions, action items and progress by (Wang and Cardie, 2013), template generation by (Oya et al., 2014),multi-sentence compression by (Shang et al., 2018), incorporation of multi-modal information by (Li et al., 2019). Recently, a very promising model was proposed by (Zhu et al., 2020) to generate meeting summarization utilizing the word and turn level hierarchical structure.

[2]https://elitr.github.io/automatic-minuting/index.html

Symbol	Representation
$\tau = (\tau_1, \tau_2, \ldots, \tau_\nu)$	Transcript (Meeting recordings)
$\mu_i = (s_1, \ldots, s_{M_i})$	Minutes
$\rho_j \in \mathrm{P}$	Speakers
α	Agenda of the meeting
s_k	Minute Item
N_i	Total utterances
δ_j	Individual utterances
η	Neural network parameters
$P(\mu\|\tau;\eta)$	Conditional probability (minute/transcript)

Table 1: Problem Description Notations

3 Problem Description

Each meeting consists of multiple participants where every person participates with some utterance or conversation represented by δ. Formally, $\tau_i = ((\rho_1, \delta_1), (\rho_2, \delta_2), \ldots, (\rho_{N_i}, \delta_{N_i}))$ where $\rho_j \in \mathrm{P}$ are the speakers, N_i is the number of utterances in the transcript τ_i and δ_j are the individual utterances (sequences of words; $1 \leq j \leq N_i$).

The minutes formed by human annotators for meeting τ_i is denoted by μ_i, which is a sequence of segments (think items in bulleted list). Formally, $\mu_i = (s_1, \ldots, s_{M_i})$, where s_k is the given minutes item, i.e. a sequence of words and punctuation.

The goal is to automatically generate the minutes $\mu_i = (\mu_1, \ldots, \mu_n)$ given the transcript $\tau = ((\rho_1, \delta_1), (\rho_2, \delta_2), \ldots, (\rho_{N_i}, \delta_{N_i}))$ for a specific agenda α of meeting (see Table 1).

4 Methods

Here we cover details of the end-to-end summarization models with the goal to maximize the conditional probability $P(\mu|\tau;\eta)$ of minute μ given a meeting transcript τ and neural network parameters η.

4.1 Extractive Methods

Given a transcript, extractive methods are supposed to select a subset of the words or sentences which best represent the discussion of the meeting. In this section, we study these extractive methods to generate minutes for a meeting automatically.

- **TF-IDF** (Christian et al., 2016) receives the input transcript for pre-processing and removes all the stopwords, stemming, and word tagging. Further, calculates their TF-IDF value and cumulate across each sentence, highest-scoring top-n selected as minutes.

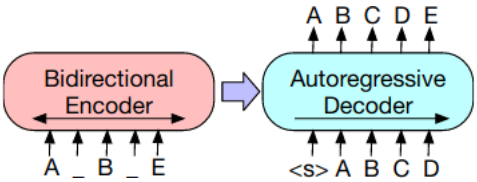


Figure 3: A systematic diagram from BART (Lewis et al., 2019)

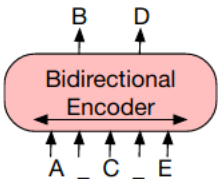


Figure 4: A schematic comparison of BART in Figure 3 with BERT from (Devlin et al., 2018)

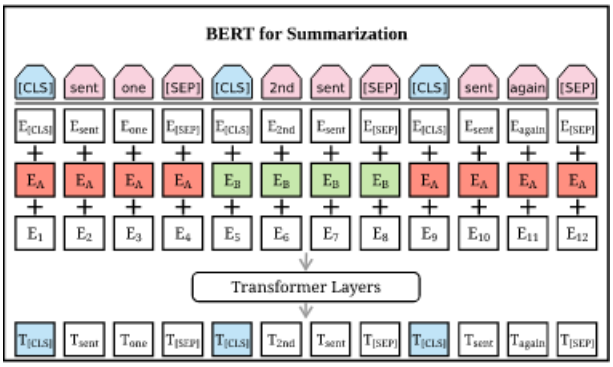


Figure 5: An illustration of BERTSUM from (Liu and Lapata, 2019)

- **Unsupervised**, is a heuristic approach, where we use different hand-crafted features (such as word frequency, cue words, numeric data, sentence length, and proper nouns) to rank the sentences. Sentences above a given threshold are selected into the minutes.
- **TextRank** (Mihalcea and Tarau, 2004) is a text summarization technique based on a graph algorithm. The input transcript has individual sentences, each represented by a vector embeddings. The similarity (refer to PageRank algorithm (Xing and Ghorbani, 2004)) between each sentence vector is stored in a matrix and converted into a graph. The graph represents sentences as vertices and similarity score as edges. The top-ranked sentences formulate the minutes for a particular transcript.
- **LexRank** (Erkan and Radev, 2004) is another text summarization technique based on a graph algorithm. It is similar to TextRank, but the edges between the vertices have a score obtained from the cosine similarity of sentences represented as TF-IDF vectors. A threshold takes only one representative of each similarity group (sentences similar enough to each other) and derives the resulting minute for the given transcript.
- **Luhn Algorithm** (Luhn, 1958) is one of the oldest algorithms proposed for summarization based on the frequency of words. It is a naive approach based on TF-IDF and focussing on the "window size" of non-important words between words of high importance. It also assigns higher weights to sentences occurring near the beginning of a document.
- **LSA: Latent Semantic Analysis (LSA)** (Gong and Liu, 2001) algorithm derives the statistical relationship of words in a sentence. It combines the term frequency in a matrix with singular value decomposition.

4.2 Abstractive Methods

Given a transcript, the task is to generate a concise minute that captures the salient notions of the meeting. The generated abstractive minute potentially contains new phrases and sentences that have not appeared in the meeting transcript.

- **BART** (Lewis et al., 2019), uses the basic seq2seq architecture with bidirectional encoder as in BERT (refer to Figure 4) with additional left-to-right denoising autoencoder (refer to Figure 3). The pretraining of seq2seq tasks involves a random shuffling of the original transcript and a novel in-filling scheme, where text spans are replaced with the mask token value. It exhibits a significant performance gains when fine-tuned for text generation and comprehension tasks.
- **BERTSUM** (Liu and Lapata, 2019) is an extension to BERT (Devlin et al., 2018) with novel document-level encoder which has multiple [CLS] symbols injected to input document

sequence for memorizing sentence representations. Additionally, it applies interval segmentation embedding (illustrated in Figure 5 with red and green color) to distinguish multiple sentences. These embeddings are summed and given as input to several bidirectional transformer layers, generating contextual vectors and further decoding. Additionally, there is new fine-tuning schedule which adopts different optimizers for the encoder and decoder for alleviating the mismatch(as the encoder is pre-trained while decoder is not).

- **BERT2BERT** (Rothe et al., 2020) uses BERT checkpoints to initialize encoder-decoder to provide a better understanding of input, mapping of input to context, and generation from context while the attention variable initialize randomly. While in this paper, we tokenize our data using WordPiece[3] to match the pre-training vocabulary for BERT as well as for noise consistency training and maintaining copy to protect gradient propagation through it.

- **Longformer Encoder-Decoder (LED)** (Beltagy et al., 2020) is another variant for longformer which supports long document generative seq-2-seq task. This encoder-decoder model has its attention mechanism, combining local window attention with task-motivated global attention that supports larger models (with thousands of tokens).

- **Pegasus** (Zhang et al., 2020) uses transformer-based encoder-decoder model for sequence-to-sequence learning. In PEGASUS, important sentences are removed/masked from an input document and are generated together as one output sequence from the remaining sentences, similar to an extractive summary.

- **Roberta2Roberta** (Liu et al., 2019) is an encoder-decoder model, meaning that both the encoder and the decoder are RoBERTa models. In this work, we initialize the Roberta-large model with checkpoints. It involves pre-training with the Masked Language Modeling (MLM) objective, where the model randomly masks 15% of the words in an input sentence and predicts them back based on other words in that sentence.

- **T5** (Raffel et al., 2019) is also an encoder-decoder transformer model. It can be easily pre-trained on a multi-task mixture of unsupervised and supervised, with each task converted in text-to-text format. In this work, we pre-train T5 by fill-in-the-blank-style with denoising objectives while using similar hyperparameters and loss functions.

5 Experiments

In this section, we describe the experimental details for off-the-shelf text summarization models for automatic minuting. We describe the hyperparameter setting for different models in Table 2.

5.1 Dataset

We base our experiments on two popular and one new dataset.

AMI For our experiments, we use the popular AMI dataset (Mccowan et al., 2005), which contains 100 hours of meeting discussions with their abstractive and extractive summaries. The audio recordings of all the meetings are provided with manually corrected transcripts. The AMI corpus contains a wide range of annotations such as dialogue acts and topic segmentation, named entities, and manually written meeting minutes. The AMI corpus consists of 138 meeting instances with their corresponding summaries.

ICSI corpus (Janin et al., 2003b) are mostly from regular meetings of computer science working teams. The corpus contains 70 hours of recordings in English (for 75 meetings collected in Berkeley during the years 2000-2002). The speech files range in length from 17 to 103 minutes and involve from 3 to 10 participants. Interestingly, the corpus contains a significant portion of non-native English speakers, varying in fluency from nearly-native to challenging-to-transcribe. All audio files are manually transcribed. ICSI consists of 75 meeting instances.

AutoMin [4] dataset is from the first shared task on

[3] https://github.com/google-research/bert/blob/master/tokenization.py

[4] https://elitr.github.io/automatic-minuting/cfp.html

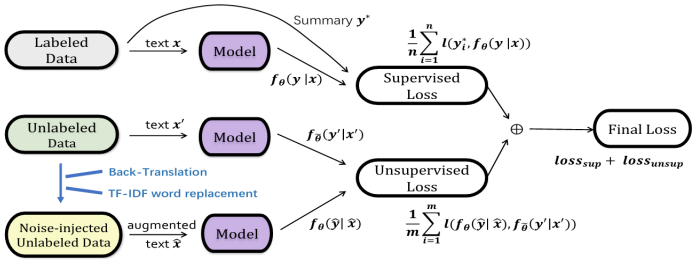


Figure 6: Illustration of BERT2BERT model for noised consistency training from (Liu et al., 2021)

Table 2: Hyperparameter settings and parameters of execution for the examined models

	Models						
Hyperparameter	**BART**	**BertSum**	**BERT2BERT**	**LED**	**Pegasus**	**Roberta2Roberta**	**T5**
learning rate	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5
weight decay	0.001	0.1	0.001	0.1	0.01	0.001	0.01
max. grad. norm	1.0	1.0	1.0	1.0	1.0	1.0	1.0
warmup steps	1300	500	300	500	1200	1400	500
batch size	24	32	32	24	48	32	32
max epochs	4	10	4	4	4	10	4
Runtime Parameters							
Python	3.7.3	3.7.3	3.7.3	3.7.3	3.7.3	3.7.3	3.7.3
GPU: GeForce RTX	2080 Ti	2080 Ti	2080 Ti	2080 Ti	3090	2080 Ti	2080 Ti
GPU count	1	1	1	1	1	1	1
GPU RAM (GB)	11	11	11	11	25	11	11
Machine RAM (GB)	248.8	248.8	248.8	248.8	183.0	248.8	248.8

automatic minuting at Interspeech 2021. It consists of manually created minutes from multiparty meeting transcripts. This dataset contains real project meetings in two different settings: technical project meetings (both in English and Czech) and parliamentary proceedings (English). We only use English data for our experiments which consists of 123 meetings with multiple minutes. For evaluation on AutoMin, we average out our scores.

5.2 Quantitative Evaluation

In this section, we evaluate all the generated outputs from different models described in Section 4 and show them in Table 3. We use the popular automatic summarization metrics like ROUGE (1, 2, L, WE) (Lin, 2004), BERTScore(Zhang et al., 2019) and BLEU (Papineni et al., 2002) which are lexical to evaluate the quality of the summary. The scores are averaged across the datasets. We see that in the abstractive methods, T5 performs best in terms of the metrics we took. It is based on Transfer learning, where a model is first pre-trained on "Colossal Clean Crawled Corpus" a data-rich task before being fine-tuned on a downstream task. It has been shown to achieve state-of-the-art results on many benchmarks covering summarization. The extractive summarization algorithms LSA performs best in the extractive methods as we analyzed. LSA algorithm exhibits the statistical relationship of words in a sentence, combining the term frequency in a matrix with singular value decomposition and therefore performs state-of-art results for AutoMin. However, these quantitative metrics indicate the quality of the generated summary by these various models across the different datasets. Along with the quantitative evaluation, we vouch for the qualitative assessment of the generated minutes.

5.3 Qualitative Evaluation

To assess the quality of the automatically generated minutes, we conduct a qualitative evaluation of those by human assessors. We evaluate the qualitative performance of both the extractive and abstractive methods that we employ for the meeting summarization task (see Section 4). We

Table 3: Quantitative Analysis of Baseline Abstractive and Extractive Summarization Methods. The highest score have been highlighted for a particular model across AMI, ICSI and AutoMin

Abstractive Approaches							
	Dataset	ROUGE_1	ROUGE_2	ROUGE_L	ROUGE_WE	BERTScore	BLUE
BART (Lewis et al., 2019)	AMI	18.29	3.42	9.95	3.33	29.47	20.63
	ICSI	6.68	00.28	3.58	0.00	43.91	20.26
	Automin	24.88	6.36	14.09	6.22	32.08	15.24
BERTSUM (Liu and Lapata, 2019)	AMI	13.25	1.73	7.42	2.19	29.59	25.37
	ICSI	5.01	0.17	2.87	0.062	4.87	20.89
	Automin	20.73	3.67	11.28	4.95	28.94	22.80
BERT2BERT (Rothe et al., 2020)	AMI	12.95	2.04	6.75	2.50	14.56	21.90
	ICSI	5.97	0.15	2.93	0.22	24.59	21.22
	Automin	23.51	5.19	12.03	6.22	19.42	15.54
LED (Beltagy et al., 2020)	AMI	5.51	0.52	4.09	0.46	43.57	34.66
	ICSI	1.45	0.03	1.12	0.02	22.34	**35.31**
	Automin	9.24	1.28	6.96	0.51	35.80	26.21
Pegasus (Zhang et al., 2020)	AMI	14.56	2.51	8.10	2.75	24.85	21.43
	ICSI	5.76	0.18	3.12	0.06	42.82	19.67
	Automin	22.72	4.55	11.97	4.66	29.12	16.68
Roberta2Roberta (Liu et al., 2019)	AMI	13.50	2.16	7.82	2.11	26.29	21.30
	ICSI	6.27	0.18	3.22	0.07	42.66	17.45
	Automin	16.67	3.12	9.48	3.13	28.09	28.90
T5 (Raffel et al., 2019)	AMI	16.14	2.70	9.00	2.92	34.34	22.44
	ICSI	5.99	0.21	3.32	0.02	**49.64**	20.77
	Automin	**27.01**	**6.71**	**14.63**	**7.59**	33.30	16.79
Extractive Approaches							
TF-IDF (Christian et al., 2016)	AMI	11.36	1.59	5.72	2.62	16.07	25.29
	ICSI	3.65	0.06	2.18	0.02	**48.71**	21.14
	Automin	19.06	3.29	8.45	3.63	25.30	22.43
Unsupervised	AMI	11.98	1.76	7.13	1.81	36.87	24.60
	ICSI	5.91	0.17	3.08	0.06	32.29	21.58
	Automin	23.45	5.04	12.96	2.68	29.93	22.60
TextRank (Mihalcea and Tarau, 2004)	AMI	10.12	1.56	5.33	2.22	8.62	24.74
	ICSI	5.94	0.12	2.85	0.05	19.28	21.64
	Automin	22.96	**5.45**	11.94	7.19	17.92	18.32
LexRank (Erkan and Radev, 2004)	AMI	10.81	1.52	5.97	2.45	12.56	25.39
	ICSI	5.03	0.11	2.82	00.03	31.62	19.72
	Automin	22.55	4.14	12.21	5.13	24.94	16.09
Luhn Algorithm (Luhn, 1958)	AMI	10.11	1.57	5.35	2.24	7.92	**26.16**
	ICSI	6.14	0.13	2.95	0.07	17.99	20.75
	Automin	22.55	4.14	12.21	5.13	24.94	19.05
LSA (Gong and Liu, 2001)	AMI	10.34	1.78	5.44	2.25	7.35	23.46
	ICSI	6.48	0.15	3.16	0.05	26.33	20.90
	Automin	**23.52**	**7.73**	**13.29**	**8.90**	14.61	22.43

ask our annotators to evaluate each automatically generated minute/meeting summary in terms of their *adequacy*, *fluency*, *grammaticality*, and *coverage* using the 5-star Likert rating scale (Likert, 1932). The annotators assign an integer from 1 (worst) to 5 (best) against each criterion to assess the *goodness* of the minutes. We had three annotators for the task evaluating a sample of randomly selected minutes from each of our three datasets generated by the different text summarization methods. We show our human evaluation of the automatically generated summaries in Table 4 by both abstractive and extractive methods. For each method, we average out the evaluations by our annotators on the sample instances. Kindly find the output samples in `https://anonymous.4open.science/r/minuting-baselines-AB22/README.md` We provide our annotators with the transcripts of the meetings and the corresponding minutes. Our annotators have at least a Master's degree and education in English. For *adequacy*, we ask our annotators to judge if the minute adequately sums up the main contents of the meeting. *Fluency* would refer to how fluent, coherent, and readable is the output minute text. *Grammaticality* would mean the grammatical correctness of the minute. Finally, by

Table 4: Qualitative Analysis of Baseline Abstractive and Extractive methods. The highest score have been highlighted for a particular model across AMI, ICSI and AutoMin

Abstractive Methods					
	Dataset	Adequacy	Fluency	Grammaticality	Coverage
BART (Lewis et al., 2019)	AMI	2.66	**3.33**	**4**	**3.33**
	ICSI	2.66	3	3.66	2.33
	Automin	**3**	3	3.33	3.33
BERTSUM (Liu and Lapata, 2019)	AMI	2.33	3.33	**4**	2.66
	ICSI	2	3	3	3
	Automin	**2.66**	**3.33**	3.66	**3**
BERT2BERT (Rothe et al., 2020)	AMI	**3.33**	3	**4**	3
	ICSI	3	**3.33**	3.33	3
	Automin	2.33	2.66	3.66	**3**
LED (Beltagy et al., 2020)	AMI	1	1	1	1
	ICSI	1	1	1.33	1
	Automin	**1.33**	**1.66**	**1.66**	**1.33**
Pegasus (Zhang et al., 2020)	AMI	2.66	**3.66**	**5**	3
	ICSI	3	2.66	3.33	**3.33**
	Automin	**3**	3	3.66	2.66
Roberta2Roberta (Liu et al., 2019)	AMI	2	2.66	3	2.33
	ICSI	**2**	**3**	**3.33**	1.66
	Automin	2	2.66	2.66	**2.33**
T5 (Raffel et al., 2019)	AMI	1.66	2	3.66	1.33
	ICSI	2	**3.33**	3.66	2.33
	Automin	**2.66**	3	**3.66**	**3**
Extractive Methods					
TF-IDF (Christian et al., 2016)	AMI	**1.66**	**2.33**	**2.66**	**2.33**
	ICSI	1.33	2	2.33	2
	Automin	1.66	2	2.66	2
Unsupervised	AMI	2	**3**	**3**	**2.33**
	ICSI	1.66	3	3	2
	Automin	**2.33**	2.66	3.33	2.33
TextRank (Mihalcea and Tarau, 2004)	AMI	**2.66**	2.66	**3**	**3.66**
	ICSI	1.33	**2.66**	2.85	2
	Automin	2	2.66	2.33	2.66
LexRank (Erkan and Radev, 2004)	AMI	**2.66**	**2.33**	**2.66**	**2.33**
	ICSI	1.66	2.33	2.33	2.33
	Automin	1.33	2.33	2.66	2.33
Luhn Algorithm (Luhn, 1958)	AMI	1	**2.66**	**2.66**	**3**
	ICSI	2	2.33	2.33	2.33
	Automin	**2.66**	2.66	3	3
LSA (Gong and Liu, 2001)	AMI	**2.33**	**3**	**3**	**3.66**
	ICSI	2.33	3	2.66	3
	Automin	1.66	2	2	2.66

coverage we ask the annotators to rate if the minutes cover the major topics in the meeting transcript.

We can see from Table 4 that the BERT-based models yield output that our annotators found better in terms of *Adequacy*, *Fluency*, and *Coverage*. BART, Pegasus, T5 score better in *Grammaticality*. Overall the scores are low for the *AutoMin* dataset as it is the only dataset that has minutes in the form of bulleted points; semantic coherence of texts is not a major priority there. However, AutoMin simulates the human minuting behavior on the fly during actual meetings. Output from the extractive methods scores comparatively less w.r.t. that of abstractive methods in human evaluation. The reason being that these extractive methods extract texts from the transcripts without regard to coherence, readability, or grammar; hence are not well ranked by our evaluators. However, we see that *TextRank* and LSA provide comparable coverage w.r.t. the deep neural-based abstractive algorithms. Each algorithm is motivated towards achieving a different objective in the generated summary, and hence there is no *one shoe fits all* algorithm for the minuting task. Hence it definitely calls for more fine-tuned algorithms towards this specific task.

6 Conclusion and Future Work

In this work, we perform an empirical analysis of several *off-the-shelf* text summarization models when applied in the task of automatic minuting. We see that automatic minuting is challenging and could not be well-addressed with the existing summarization models. Both our quantitative and qualitative evaluation reveals that the extractive models perform better than the abstractive ones. However, they are still far from being acceptable. To sum up, we intend to provide baseline evaluations to the community for this challenging task with this paper. As future work, we would want to explore a template-based extractive method to generate the meeting summary from the transcripts. Our investigation indicates that leveraging on BERTSum could be a plausible direction to probe next. In future we would try, if possible, speaker segmentation embedding (i.e. EA, EB, EC, ED ...) for BERTSUM model to reflect different speakers in multi-party dialogue, instead of interval segmentation embedding (i.e. EA, EB, EA, EB ...).

Acknowledgements

This work has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No 825460 (ELITR) and the grant 19-26934X (NEUREM3) of the Czech Science Foundation.

References

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Jean Carletta. 2007. Unleashing the killer corpus: Experiences in creating the multi-everything ami meeting corpus. *Language Resources and Evaluation*, 41(2):181–190.

Asli Celikyilmaz, Antoine Bosselut, Xiaodong He, and Yejin Choi. 2018. Deep communicating agents for abstractive summarization. *arXiv preprint arXiv:1803.10357*.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. *arXiv preprint arXiv:1805.11080*.

Yun-Nung Chen and Florian Metze. 2012. Integrating intra-speaker topic modeling and temporal-based inter-speaker topic modeling in random walk for improved multi-party meeting summarization. In *Thirteenth Annual Conference of the International Speech Communication Association*.

Sangwoo Cho, Logan Lebanoff, Hassan Foroosh, and Fei Liu. 2019. Improving the similarity measure of determinantal point processes for extractive multi-document summarization. *arXiv preprint arXiv:1906.00072*.

Sumit Chopra, Michael Auli, and Alexander M Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 93–98.

Hans Christian, Mikhael Pramodana Agus, and Derwin Suhartono. 2016. Single document automatic text summarization using term frequency-inverse document frequency (tf-idf). *ComTech: Computer, Mathematics and Engineering Applications*, 7(4):285–294.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479.

Yihong Gong and Xin Liu. 2001. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 19–25.

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. *arXiv preprint arXiv:1804.11283*.

Karl Moritz Hermann, Tomáš Kočiskỳ, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *arXiv preprint arXiv:1506.03340*.

A. Janin, D. Baron, J. Edwards, D. Ellis, D. Gelbart, N. Morgan, B. Peskin, T. Pfau, E. Shriberg, A. Stolcke, and C. Wooters. 2003a. The icsi meeting corpus. In *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03).*, volume 1, pages I–I.

A. Janin, D. Baron, J. Edwards, D. Ellis, D. Gelbart, N. Morgan, B. Peskin, T. Pfau, E. Shriberg, A. Stolcke, and C. Wooters. 2003b. The icsi meeting corpus. pages 364–367.

Ruipeng Jia, Yanan Cao, Hengzhu Tang, Fang Fang, Cong Cao, and Shi Wang. 2020. Neural extractive summarization with hierarchical attentive heterogeneous graph network. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3622–3631.

Logan Lebanoff, Kaiqiang Song, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019. Scoring sentence singletons and pairs for abstractive summarization. *arXiv preprint arXiv:1906.00077*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Manling Li, Lingyu Zhang, Heng Ji, and Richard J Radke. 2019. Keep meeting summaries on topic: Abstractive multi-modal meeting summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2190–2196.

Rensis Likert. 1932. A technique for the measurement of attitudes. *Archives of psychology*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. *arXiv preprint arXiv:1908.08345*.

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. *arXiv preprint arXiv:1907.11692*.

Junnan Liu, Qianren Mao, Bang Liu, Hao Peng, Hongdong Zhu, and Jianxin Li. 2021. Noised consistency training for text summarization. *arXiv preprint arXiv:2105.13635*.

Hans Peter Luhn. 1958. The automatic creation of literature abstracts. *IBM Journal of research and development*, 2(2):159–165.

I. Mccowan, G. Lathoud, M. Lincoln, A. Lisowska, W. Post, D. Reidsma, and P. Wellner. 2005. The ami meeting corpus. In *In: Proceedings Measuring Behavior 2005, 5th International Conference on Methods and Techniques in Behavioral Research. L.P.J.J. Noldus, F. Grieco, L.W.S. Loijens and P.H. Zimmerman (Eds.), Wageningen: Noldus Information Technology*.

Yashar Mehdad, Giuseppe Carenini, Frank Tompa, and Raymond Ng. 2013. Abstractive meeting summarization with entailment and fusion. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 136–146.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Tatsuro Oya, Yashar Mehdad, Giuseppe Carenini, and Raymond Ng. 2014. A template-based abstractive meeting summarization: Leveraging summary and source text relationships. In *Proceedings of the 8th International Natural Language Generation Conference (INLG)*, pages 45–53.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging pre-trained checkpoints for sequence generation tasks. *Transactions of the Association for Computational Linguistics*, 8:264–280.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Guokan Shang, Wensi Ding, Zekun Zhang, Antoine Jean-Pierre Tixier, Polykarpos Meladianos, Michalis Vazirgiannis, and Jean-Pierre Lorré. 2018. Unsupervised abstractive meeting summarization with multi-sentence compression and budgeted submodular maximization. *arXiv preprint arXiv:1805.05271*.

Lu Wang and Claire Cardie. 2013. Domain-independent abstract generation for focused meeting summarization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1395–1405.

Danqing Wang, Pengfei Liu, Yining Zheng, Xipeng Qiu, and Xuanjing Huang. 2020. Heterogeneous graph neural networks for extractive document summarization. *arXiv preprint arXiv:2004.12393*.

Wenpu Xing and Ali Ghorbani. 2004. Weighted pagerank algorithm. In *Proceedings. Second Annual Conference on Communication Networks and Services Research, 2004.*, pages 305–314. IEEE.

Jiacheng Xu and Greg Durrett. 2019. Neural extractive text summarization with syntactic compression. *arXiv preprint arXiv:1902.00863*.

Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. Discourse-aware neural extractive text summarization. *arXiv preprint arXiv:1910.14142*.

Klaus Zechner. 2001. *Automatic Summarization of Spoken Dialogues in Unrestricted Domains*. Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA, USA.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Ming Zhong, Pengfei Liu, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2019. Searching for effective neural extractive summarization: What works and what's next. *arXiv preprint arXiv:1907.03491*.

Chenguang Zhu, Ruochen Xu, Michael Zeng, and Xuedong Huang. 2020. A hierarchical network for abstractive meeting summarization with cross-domain pretraining. *arXiv preprint arXiv:2004.02016*.