GSum: A General Framework for *G*uided Neural Abstractive *S*ummarization

Zi-Yi Dou, Pengfei Liu, Hiroaki Hayashi, Zhengbao Jiang, Graham Neubig
Language Technologies Institute, Carnegie Mellon University
{zdou, pliu3, hiroakih, zhengbaj, gneubig}@cs.cmu.edu

Abstract

Neural abstractive summarization models are flexible and can produce coherent summaries, but they are sometimes unfaithful and can be difficult to control. While previous studies attempt to provide different types of guidance to control the output and increase faithfulness, it is not clear how these strategies compare and contrast to each other. In this paper, we propose a general and extensible guided summarization framework (**GSum**) that can effectively take different kinds of external guidance as input, and we perform experiments across several different varieties. Experiments demonstrate that this model is effective, achieving state-of-the-art performance according to ROUGE on 4 popular summarization datasets when using highlighted sentences as guidance. In addition, we show that our guided model can generate more faithful summaries and demonstrate how different types of guidance generate qualitatively different summaries, lending a degree of controllability to the learned models.[1]

1 Introduction

Modern techniques for text summarization generally can be categorized as either *extractive* methods (Nallapati et al., 2017; Narayan et al., 2018b; Zhou et al., 2018), which identify the most suitable words or sentences from the input document and concatenate them to form a summary, or *abstractive* methods (Rush et al., 2015; Chopra et al., 2016; Nallapati et al., 2016; Paulus et al., 2018), which generate summaries freely and are able to produce novel words and sentences. Compared with extractive algorithms, abstractive algorithms are more flexible, making them more likely to produce fluent and coherent summaries. However, the unconstrained nature of abstractive summarization can also result in problems. First, it can result in *unfaithful* summaries (Kryściński et al., 2019), containing factual errors as well as hallucinated content. Second, it can be difficult to *control* the content of summaries; it is hard to pick in advance which aspects of the original content an abstractive system may touch upon. To address the issues, we propose methods for *guided* neural abstractive summarization: methods that provide various types of guidance signals that 1) constrain the summary so that the output content will deviate less from the source document; 2) allow for controllability through provision of user-specified inputs.

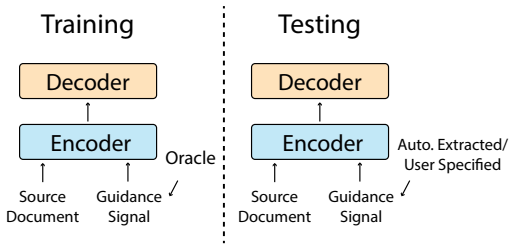

Figure 1: Our framework generates summaries using both the source document and separate guidance signals. We use an oracle to select guidance during training and use automatically extracted or user-specified guidance at test time.

There have been some previous methods for guiding neural abstractive summarization models. For example, Kikuchi et al. (2016) specify the length of abstractive summaries, Li et al. (2018) provide models with keywords to prevent the model from missing key information, and Cao et al. (2018) propose models that retrieve and reference relevant summaries from the training set. While these methods have demonstrated improvements in summarization quality and controllability, each focuses on one particular type of guidance – it remains unclear which is better and whether they are complementary to each other.

In this paper, we propose a *general and extensible guided summarization framework* that can take different kinds of external guidance as in-

[1]Code is available at https://github.com/neulab/guided_summarization.

Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 4830–4842
June 6–11, 2021.

Work	Guidance Form			
	Tokens	**Triples**	**Sentences**	**Summaries**
Kikuchi et al. (2016)	✓ (length tokens)	✗	✗	✗
Cao et al. (2018)	✗	✗	✗	✓ (retrieved sums.)
Li et al. (2018)	✓ (keywords)	✗	✗	✗
Liu et al. (2018a)	✗	✗	✓ (highlighted sents.)	✗
Liu et al. (2018b)	✓ (length tokens)	✗	✗	✗
Fan et al. (2018)	✓ (length, entity, style tokens)	✗	✗	✗
Zhu et al. (2020)	✗	✓ (relations)	✗	✗
Jin et al. (2020)	✗	✓ (relations)	✗	✗
Saito et al. (2020)	✓ (keywords)	✗	✓ (highlighted sents.)	✗
Ours	✓ (keywords)	✓ (relations)	✓ (highlighted sents.)	✓ (retrieved sums.)

Table 1: A comparison of different guided neural abstractive summarization models. Previous works have tried to provide guidance in different forms, including tokens, triples, sentences and summaries. Our proposed framework can incorporate them together and we have experimented with all four forms.

put. Like most recent summarization models, our model is based on neural encoder-decoders, instantiated with contextualized pretrained language models, including BERT (Devlin et al., 2019) and BART (Lewis et al., 2020). With this as a strong starting point, we make modifications allowing the model to attend to *both* the source documents and the guidance signals when generating outputs. As shown in Figure 1, we can provide automatically extracted or user-specified guidance to the model during test time to constrain the model output. At training time, to encourage the model to pay close attention to the guidance, we propose to use an *oracle* to select informative guidance signals – a simple modification that nonetheless proved essential in effective learning of our guided summarization models. Using this framework, we investigate four types of guidance signals: (1) highlighted sentences in the source document, (2) keywords, (3) salient relational triples in the form of (subject, relation, object), and (4) retrieved summaries.

We evaluate our methods on 6 popular summarization benchmarks. Our best model, using highlighted sentences as guidance, can achieve state-of-the-art performance on 4 out of the 6 datasets, including 1.28/0.79/1.13 ROUGE-1/2/L improvements over previous state-of-the-art model on the widely-used CNN/DM dataset. In addition, we perform in-depth analyses of different guidance signals and demonstrate that they are complementary to each other in that there is potential to aggregate their outputs together and obtain further improvements. An analysis of the results also reveals that our guided models can generate more faithful summaries and more novel words. Finally, we demonstrate that we can control the output by providing user-specified guidance signals, with different provided signals resulting in qualitatively different summaries.

2 Background and Related Work

Neural abstractive summarization typically takes a source document $\mathbf{x}$ consisting of multiple sentences $x_1, \cdots, x_{|\mathbf{x}|}$, runs them through an encoder to generate representations, and passes them to a decoder that outputs the summary $\mathbf{y}$ one target word at a time. Model parameters θ are trained to maximize the conditional likelihood of the outputs in a parallel training corpus $\langle \mathcal{X}, \mathcal{Y} \rangle$:

$$\arg\max_{\theta} \sum_{\langle \mathbf{x}^i, \mathbf{y}^i \rangle \in \langle \mathcal{X}, \mathcal{Y} \rangle} \log p(\mathbf{y}^i \mid \mathbf{x}^i; \theta).$$

Several techniques have been proposed to improve the model architecture. For example, models of copying (Gu et al., 2016; See et al., 2017; Gehrmann et al., 2018) allow words to be copied directly from the input to the output, and models of coverage discourage the model from generating repetitive words (See et al., 2017).

Guidance can be defined as some variety of signal $\mathbf{g}$ that is fed into the model in addition to the source document $\mathbf{x}$:

$$\arg\max_{\theta} \sum_{\langle \mathbf{x}^i, \mathbf{y}^i, \mathbf{g}^i \rangle \in \langle \mathcal{X}, \mathcal{Y}, \mathcal{G} \rangle} \log p(\mathbf{y}^i \mid \mathbf{x}^i, \mathbf{g}^i; \theta).$$

Within this overall framework, the types of information that go into $\mathbf{g}$ and the method for incorporating this information into the model may vary. While there are early attempts at non-neural guided models (Owczarzak and Dang, 2010; Genest and Lapalme, 2012), here we focus on neural approaches

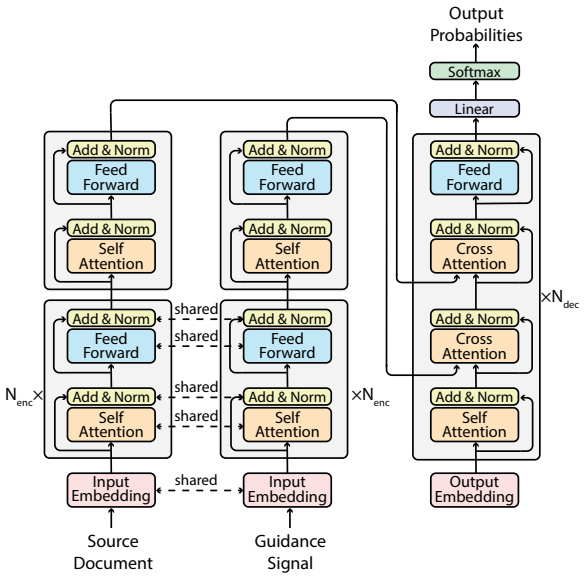

Figure 2: General framework of our model. The two encoders encode the source document and guidance signal, which are attended to by the decoder.

and summarize recent work in Table 1. For example, Li et al. (2018) first generate a set of *keywords*, which are then incorporated into the generation process by an attention mechanism. Cao et al. (2018) propose to search the training corpus and *retrieve* datapoint $\langle \mathbf{x}^j, \mathbf{y}^j \rangle$ whose input document $\mathbf{x}^j$ is most relevant to the current input $\mathbf{x}$, and treat $\mathbf{y}^j$ as a candidate template to guide the summarization process. Besides, Jin et al. (2020) and Zhu et al. (2020) extract *relational triples* in the form of (subject, relation, object) from source documents and represent them by graph neural networks. The decoders then attend to the extracted relations to generate faithful summaries. A concurrent work by Saito et al. (2020) propose to extract *keywords* or *highlighted sentences* using saliency models and feed them to summarization models.

There are also works on controlling the summary length (Kikuchi et al., 2016; Liu et al., 2018b) and styles (Fan et al., 2018) by explicitly feeding the desired features to the model. In addition, Liu et al. (2018a) and Chen and Bansal (2018) follow a two-stage paradigm, in which a subset of the source document $\{x_{i_1}, \cdots, x_{i_n}\}$ will first be selected by a pretrained extractor as *highlighted sentences* and then be fed into the model encoder in the second stage with the rest of the text discarded.

3 Methods

Figure 2 illustrates the general framework of our proposed method. We feed both the source documents and various types of guidance signals to the model. Specifically, we experiment with guidance signals including highlighted sentences, keywords, relations, and retrieved summaries, although the framework is general and could be expanded to other varieties of guidance as well.

3.1 Model Architecture

We adopt the Transformer model (Vaswani et al., 2017) as our backbone architecture, instantiated with BERT or BART, which can be separated into the encoder and decoder components.

3.1.1 Encoder

Our model has two encoders, encoding the input source document and guidance signals respectively.

Similar to the Transformer model, each of our encoders is composed of $N_{enc} + 1$ layers, with each encoding layer containing both a self-attention block and a feed-forward block:

$$\mathbf{x} = \text{LN}(\mathbf{x} + \textsc{SelfAttn}(\mathbf{x})),$$
$$\mathbf{x} = \text{LN}(\mathbf{x} + \textsc{FeedForward}(\mathbf{x})),$$

where LN denotes layer normalization. Note the source document and guidance signal do not interact with each other during encoding.

We share the parameters of the bottom N_{enc} layers and the word embedding layers between the two encoders, because 1) this can reduce the computation and memory requirements; 2) we conjecture that the differences between source documents and guidance signals should be high-level, which are captured at top layers of the encoders.

3.1.2 Decoder

Different from the standard Transformer, our decoder has to attend to both the source document and guidance signal instead of just one input.

Concretely, our decoder is composed of N_{dec} identical layers, with each layer containing four blocks. After the self-attention block, the decoder will first attend to the guidance signals and generate the corresponding representations, and hence the guidance signal will inform the decoder which part of the source documents should be focused on. Then, the decoder will attend to the whole source document based on the guidance-aware representations. Finally, the output representation will be fed into the feed-forward block:

$$\mathbf{y} = \text{LN}(\mathbf{y} + \textsc{SelfAttn}(\mathbf{y})),$$
$$\mathbf{y} = \text{LN}(\mathbf{y} + \textsc{CrossAttn}(\mathbf{y}, \mathbf{g})),$$
$$\mathbf{y} = \text{LN}(\mathbf{y} + \textsc{CrossAttn}(\mathbf{y}, \mathbf{x})),$$
$$\mathbf{y} = \text{LN}(\mathbf{y} + \textsc{FeedForward}(\mathbf{y})).$$

Ideally, the second cross-attention block allows the model to fill in the details of the input guidance signal, such as finding the name of an entity by searching through co-reference chains.

3.2 Choices of Guidance Signals

Before delving into the specifics of the types of guidance signal we used, we first note an important detail in training our model. At test time, there are two ways we can define the guidance signal: 1) **manual definition** where an interested user defines the guidance signal $\mathbf{g}$ by hand, and 2) **automatic prediction** where an automated system is used to infer the guidance signal $\mathbf{g}$ from input $\mathbf{x}$. We demonstrate results for both in experiments.

At training time, it is often prohibitively expensive to obtain manual guidance. Hence, we focus on two varieties of generating them: 1) **automatic prediction** using $\mathbf{x}$ as detailed above, and 2) **oracle extraction** where we use *both* $\mathbf{x}$ and $\mathbf{y}$ to deduce a value $\mathbf{g}$ that is most likely useful in generating $\mathbf{y}$.

Theoretically, automatic prediction has the advantage of matching the training and testing conditions of a system that will also receive automatic predictions at test time. However, as we will show in experiments, the use of oracle guidance has a large advantage of generating guidance signals that are highly informative, thus encouraging the model to pay more attention to them at test time.

With this in mind, we describe the four varieties of guidance signal we experiment with, along with their automatic and oracle extraction methods.

Highlighted Sentences. The success of extractive approaches have demonstrated that we can extract a subset of sentences $\{x_{i_1}, \cdots, x_{i_n}\}$ from the source document and concatenate them to form a summary. Inspired by this, we explicitly inform our model which subset of source sentences should be highlighted using extractive models.

We perform oracle extraction using a greedy search algorithm (Nallapati et al., 2017; Liu and Lapata, 2019) to find a set of sentences in the source document that have the highest ROUGE scores with the reference (detailed in Appendix) and treat these as our guidance $\mathbf{g}$. At test time, we use pretrained extractive summarization models (BertExt (Liu and Lapata, 2019) or MatchSum (Zhong et al., 2020) in our experiments) to perform automatic prediction.

Keywords. If we select full sentences, they may contain unnecessary information that does not occur in an actual summary, which could distract the model from focusing on the desired aspects of the input. Therefore, we also try to feed our model with a set of individual keywords $\{w_1, \ldots, w_n\}$ from the source document.

For oracle extraction, we first use the greedy search algorithm mentioned above to select a subset of input sentences, then use TextRank (Mihalcea and Tarau, 2004) to extract keywords from these sentences. We also filter the keywords that are not in the target summary. The remaining keywords are then fed to our models. For automatic prediction, we use another neural model (BertAbs (Liu and Lapata, 2019) in the experiments) to predict the keywords in the target summary.

Relations. Relations are typically represented in the form of relational triples, with each triple containing a subject, a relation, and an object. For example, *Barack Obama was born in Hawaii* will create a triple *(Barack Obama, was born in, Hawaii)*.

For oracle extraction, we first use Stanford OpenIE (Angeli et al., 2015) to extract relational triples from the source document. Similar to how we select highlighted sentences, we then greedily select a set of relations that have the highest ROUGE score with the reference, which are then flattened and treated as guidance. For automatic prediction, we use another neural model (similarly, BertAbs) to predict the relation triples on the target side.

Retrieved Summaries. Intuitively, gold summaries of similar documents with the input can provide a reference point to guide the summarization. Therefore, we also try to retrieve relevant summaries from the training data $\langle \mathcal{X}, \mathcal{Y} \rangle$.

For oracle extraction, we directly retrieve five datapoints $\{\langle \mathbf{x}_1, \mathbf{y}_1 \rangle, \ldots, \langle \mathbf{x}_5, \mathbf{y}_5 \rangle\}$ from training data whose summaries $\mathbf{y}_i$ are most similar to the target summary $\mathbf{y}$ using Elastic Search.[2] For automatic prediction at test time, we retrieve five datapoints whose source documents $\mathbf{x}_i$ are most similar to each input source document $\mathbf{x}$ instead.

4 Experiments

4.1 Datasets

We experiment on 6 datasets (statistics in Table 2):

Reddit (Kim et al., 2019) is a highly abstractive dataset and we use its *TIFU-long* version.

[2] https://github.com/elastic/elasticsearch

Dataset	Source	#Pairs			#Tokens		#Ext
		Train	Valid	Test	Doc.	Sum.	
Reddit	Social Media	41,675	645	645	482.2	28.0	2
XSum	News	203,028	11,273	11,332	430.2	23.3	2
CNN/DM	News	287,084	13,367	11,489	766.1	58.2	3
WikiHow	Knowledge Base	168,126	6,000	6,000	580.8	62.6	4
NYT	News	44,382	5,523	6,495	1183.2	110.8	4
PubMed	Scientific Paper	83,233	4,946	5,025	444.0	209.5	6

Table 2: Statistics of the datasets. **#Ext** denotes the number of sentences we extract for extractive summarization.

XSum (Narayan et al., 2018a) is an abstractive dataset that contains one-sentence summaries of online articles from BBC.

CNN/DM (Hermann et al., 2015; Nallapati et al., 2016) is a widely-used summarization dataset consisting of news articles and associated highlights as summaries. We use its non-anonymized version.

WikiHow (Koupaee and Wang, 2018) is extracted from an online knowledge base and requires high level of abstraction.

New York Times (NYT) (Sandhaus, 2008) is a dataset that consists of news articles and their associated summaries.[3] We follow Kedzie et al. (2018) to preprocess and split the dataset.

PubMed (Cohan et al., 2018) is relatively extractive and is collected from scientific papers.

4.2 Baselines

Our baselines include the following models:

BertExt (Liu and Lapata, 2019) is an extractive model whose parameters are initialized with BERT (Devlin et al., 2019).

BertAbs (Liu and Lapata, 2019) is an abstractive model with encoder initialized with BERT and trained with a different optimizer than its decoder.

MatchSum (Zhong et al., 2020) is an extractive model that reranks the candidate summaries produced by BertExt and achieves state-of-the-art extractive results on various summarization datasets.

BART (Lewis et al., 2020) is an state-of-the-art abstractive summarization model pretrained with a denoising autoencoding objective.

4.3 Implementation Details

We build our models based on both BertAbs and BART, and follow their hyperparameter settings to train our summarizers. For our model built on BertAbs, there are 13 encoding layers, with the top layer randomly initialized and separately trained between the two encoders. For our model built on BART, there are 24 encoding layers, with the top layer initialized with pretrained parameters yet separately trained between the two encoders. The first cross-attention block of the decoder is randomly initialized whereas the second cross-attention block is initialized with pretrained parameters. BertAbs is used to predict guidance signals of relations and keywords during test time. Unless otherwise stated, we use oracle extractions at training time.

[3] https://catalog.ldc.upenn.edu/LDC2008T19

Model	Guide	R-1	R-2	R-L
BertExt* (Base)	-	43.25	20.24	39.63
BertAbs*	-	41.72	19.39	38.76
BertAbs (Ours)	-	41.58	18.99	38.56
Ours				
BertAbs + Sentence	Auto.	43.78	20.66	40.66
	Oracle	55.18	32.54	52.06
BertAbs + Keyword	Auto.	42.21	19.36	39.23
	Oracle	45.08	22.22	42.07
BertAbs + Relation	Auto.	41.40	18.66	38.40
	Oracle	45.96	23.09	42.92
BertAbs + Retrieve	Auto.	40.88	18.24	37.99
	Oracle	43.69	20.53	40.71

Table 3: Results (ROUGE; Lin (2004)) on CNN/DM. "Auto" and "oracle" denote using automatically predicted and oracle-extracted guidance at test time respectively. Results with * are from Liu and Lapata (2019).

4.4 Main Results

We first compare different kinds of guidance signals on the CNN/DM dataset using BertAbs, then evaluate the best guidance on the other five datasets using both BertAbs and BART.

Performance of Different Guidance Signals. As shown in Table 3, if we feed the model with automatically constructed signals, feeding either highlighted sentences or keywords can outperform the abstractive summarization baseline by a large margin. Especially, feeding highlighted sentences can outperform the best baseline by more than 1

Model	R-1	R-2	R-L
Oracle	55.76	33.22	51.83
Extractive			
BertExt (Base)*	43.25	20.24	39.63
BertExt (Large)*	43.85	20.34	39.90
MatchSum†	44.41	20.86	40.55
Abstractive			
BertAbs*	41.72	19.39	38.76
BertAbs (Ours)	41.58	18.99	38.56
BertExtAbs*	42.13	19.60	39.18
BART ‡	44.16	21.28	40.90
BART (Ours)	44.66	21.53	41.35
Ours			
BertAbs + BertExt	43.78	20.66	40.66
BART + MatchSum	**45.94**	**22.32**	**42.48**

Table 4: Comparisons with state-of-the-art models on CNN/DM. The highest numbers are in **bold**. Marked results are from Liu and Lapata (2019)*, Zhong et al. (2020)†, Lewis et al. (2020)‡.

ROUGE-L point. Using relations or retrieved summaries as guidance will not improve the baseline performance, likely because it is hard to predict these signals during test time.

If we use an oracle to select the guidance signals, all varieties of guidance can improve the baseline performance significantly, with the best-performing model achieving a ROUGE-1 score of 55.18. The results indicate that 1) the model performance has the potential to be further improved given a better guidance prediction model; 2) the model does learn to depend on the guidance signals.

Comparisons with State of the Art. We then try to build our model on the state-of-the-art model, using highlighted sentences as guidance as it achieves the best performance on CNN/DM. First, we build our model on BART and train it with oracle-extracted highlighted sentences as guidance. Then, we use MatchSum to predict the guidance at test time. From Table 4, we can see that our model can achieve over 1 ROUGE-1/L point improvements compared with the state-of-the-art models, indicating the effectiveness of the proposed methods.

Performance on Other Datasets. We report the performance of the highlighted sentence model on all the other five datasets in Table 5. Generally, the model works better when the dataset is more extractive. For abstractive datasets such as Reddit and XSum, our model cannot achieve performance increases when the abstractive summarization baseline is already rather strong. For extractive datasets such as PubMed and NYT, on the other hand, our model can achieve some improvements over the baselines even though the abstractive baseline outperforms the extractive oracle model in some cases.

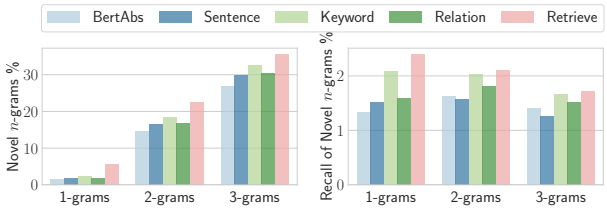

Figure 3: Our model can generate more novel words and achieve higher recall of novel words in the gold reference compared with baseline.

4.5 Analysis

We perform extensive analyses on CNN/DM to gain insights into our (BERT-based) models. Unless otherwise stated, we use oracle extractions at training time and automatic prediction at test time.

Novel n-grams. While we sometimes provide information extracted from the source document as guidance signals, it is unclear whether the model will over-fit to and regurgitate this guidance, or still generate novel expressions. To measure this, we count the number of novel n-grams in the output summaries, namely n-grams that do not appear in the source document. As shown in Figure 3, all of our guided models in fact generate *more* novel n-grams than the baseline, likely because at training time the model is trained to compress and paraphrase the extracted information from the source document into the gold summary. In addition, our models cover more novel n-grams that are in the gold reference than baseline. The results indicate that our guided models can indeed generate novel expressions, and are not referencing the input guidance too strongly.

Complementarity of Different Guidance Signals. While some guidance signals achieve worse performance than others, it is still possible to aggregate their outputs and obtain better performance if their outputs are diverse and they complement each-other. To verify this hypothesis, we try to select the best output of the four guidance signals for each test datapoint and investigate if we can aggregate their best outputs and achieve better performance.

Concretely, for each test input, we perform an oracle experiment where we compute the ROUGE

Model	Reddit			XSum			WikiHow			PubMed			NYT		
	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
Oracle	36.21	13.74	28.93	29.79	8.81	22.66	35.59	12.98	32.68	45.12	20.33	40.19	58.44	38.39	50.00
Extractive															
BertExt (Base)	23.86	5.85	19.11	22.86	4.48	17.16	30.40	8.67	28.32	40.29	14.37	35.88	45.98	25.29	42.46
MatchSum	25.09	6.17	20.13	24.86	4.66	18.41	31.85	8.98	29.58	41.21	14.91	36.75	46.98	26.67	43.62
Bert-Based															
BertAbs	**26.92**	6.35	19.81	38.76	**16.33**	**31.15**	38.16	15.06	34.71	36.04	12.16	29.02	49.94	31.44	46.67
Ours (BertAbs + MatchSum)	26.89	**6.75**	**20.35**	**38.77**	16.14	30.96	**38.29**	**15.10**	**34.80**	**37.82**	**12.32**	**30.53**	**50.50**	**31.57**	**47.24**
BART-Based															
BART	**35.00**	**12.89**	**27.96**	**45.51**	**21.94**	**36.75**	41.46	**17.80**	39.89	44.72	16.48	41.00	54.13	35.15	47.00
Ours (BART + MatchSum)	34.52	12.71	27.58	45.40	21.89	36.67	**41.74**	17.73	**40.09**	**45.09**	**16.72**	**41.32**	**54.27**	**35.37**	**47.63**

Table 5: Results of our model guided with highlighted sentences on five datasets. Highest numbers in each section are in **bold**. We use MatchSum to predict the guidance at test time. Extractive results are from Zhong et al. (2020).

Win [%]				Combined
Sentence	Keyword	Relation	Retrieve	R-1/R-2/R-L
39.28	19.55	21.12	20.05	48.30/25.25/45.15

Table 6: No guidance signals can outperform all the other ones for all the test data, and aggregating the best outputs of the four guided models achieves significant improvements over the best single guided model (43.78/20.66/40.66 R-1/R-2/R-L scores).

	Sentence	Keyword	Relation	Retrieve
Sentence	<u>43.78</u>	46.11	46.20	46.27
Keyword	-	<u>42.21</u>	44.39	44.35
Relation	-	-	<u>41.40</u>	44.60
Retrieve	-	-	-	<u>40.88</u>

Table 7: Combining the best outputs of each pair of guidance signals leads to improvements (in terms of ROUGE-1), indicating every pair of guidance complements each other. The <u>underlined</u> results are the model performance without combinations.

score of each output of the four guidance signals and pick the best one. As shown in Table 6, despite the fact that the highlighted sentence signal achieves the best overall performance, it still underperforms one of the other three varieties of guidance more than 60% of the time. In addition, by aggregating their best outputs together, we can achieve a ROUGE-1/L point of 48.30/45.15, which significantly outperforms any single guided model. Further, we try to aggregate these guidance signals in a pairwise manner, and Table 7 demonstrates that each guidance signal is complementary to each other to some extent. Thus, we can safely conclude that each type of guidance signal has its own merits and one promising direction is to utilize a system combination method such as Hong et al. (2015) to aggregate the results together.

Controllability. It is also of interest what effect this guidance has on the model outputs qualitatively. We sample several generated outputs (Table 8) and find that different provided signals can result in different outputs. Especially, for our sentence-guided model, providing the model with *by running tissue paper over his son seth makes him sleep* enables the model to generate the exact same sentence, and when the model is fed with *one grateful viewer of the video commented...*, it will generate *one viewer commented....* The examples demonstrate that our model can generate summaries mostly faithful to the guidance signals while also performing abstraction.

Faithfulness of Generated Summaries. We also evaluate whether our generated summaries are faithful to the source document. We randomly sample 100 datapoints from the test set and ask 3 people from Amazon Mechanical Turk to evaluate their factual correctness. Each person gives a score between 1 and 3, with 3 being perfectly faithful to the source document. Table 9 shows that our guided model can generate more faithful summaries compared with the baseline.

Necessity of Using Oracles During Training. As mentioned previously, we use an oracle to select guidance signals during training. In this part, we investigate if we can provide automatically constructed guidance to the model during training as well. Table 10 shows that this methodology will lead to significantly worse performance. We con-

Model	Guidance	Output
Ref.	-	nathan dailo has found a way to get his son to sleep in 42 seconds. in a youtube video he demonstrates how stroking his 3-month-old son's face with a white piece of tissue paper sends him to sleep. the video has received almost 26,000 views in just two weeks.
Sentence	**nathan dailo, from sydney, uploaded a video to his youtube channel demonstrating how he gets his three-month-old son seth to** drift off **in just 42 seconds**. **the clip that has now received almost 26,000 views** sees the father gliding a piece of white tissue paper over his son's face repeatedly until he nods off. in his youtube video, nathan dailo shows that **by running tissue paper over his son seth makes him sleep.**	**nathan dailo, from sydney,** ***uploaded a video to his youtube channel demonstrating how he gets his three-month-old son seth to*** *sleep* **in just 42 seconds**. **the clip has now received almost 26,000 views**. ***by running tissue paper over his son seth makes him sleep***.
	the clip that has now received almost 26,000 views sees the father gliding a piece of white tissue paper over his son's face repeatedly until he nods off. in his youtube video, **nathan dailo shows that by running tissue paper over his son seth makes him sleep**. **one** grateful **viewer** of the video **commented:'this worked for my baby 5 times already so thanks!'**	**nathan dailo**, from sydney, ***shows that by running tissue paper over his son seth makes him sleep*** in 42 seconds. **the clip has now received almost 26,000 views**. ***one viewer commented:'this worked for my baby 5 times already so thanks!'***
Keyword	**video sleep dailo youtube** way **seconds demonstrates weeks** piece **received**	nathan **dailo**, from sydney, uploaded a **video** to his **youtube** channel. he **demonstrates** how he gets his three-month-old son seth to **sleep** in just 42 **seconds**. *the clip has now* ***received*** *almost 26,000 views since being uploaded two* ***weeks*** *ago.*
	son paper sends found **demonstrates** way views **face dailo piece**	nathan **dailo**, from sydney, posted a video to his youtube channel. he **demonstrates** how he gets his three-month-old **son** seth to sleep in just 42 seconds. *the father glides a piece of white* ***tissue paper*** *over his son's* ***face*** *repeatedly until he nods off.*
Relation	(**father**, gliding, piece of **tissue paper**), (**clip**, sees **father over**, **his son**'s face)	*a* ***father*** *from sydney has worked out a way of getting his three-month-old son to sleep.* in his youtube video, nathan dailo shows that by running **tissue paper** over **his son** seth makes him sleep in just 42 seconds. the **clip** has now received almost 26,000 views since being uploaded two weeks ago.
	(**nathan dailo**, **uploaded**, **video to his youtube channel**), (has drifted, **has received**, **26,000 views**)	***nathan dailo****, from sydney,* ***uploaded a video to his youtube channel***. the clip **has received 26,000 views** since being uploaded two weeks ago. it shows the father gliding a piece of white tissue paper over his son's face repeatedly until he nodded off.
Retrieve	phil smith **posted** the **video** of him. **his son** noah's tooth **to his youtube channel** which went viral with over 100,000 **views**. smith tied noah's tooth to dental attached to a golf ball. smith hits the ball and the tooth is gone **in just seconds**, made by the expression in noah's face.	*nathan dailo, from sydney,* ***posted*** *a* ***video to his youtube channel***. he demonstrates how he gets **his** three-month-old **son** to sleep **in just** 42 **seconds**. the clip has now received 26,000 **views** since being uploaded two weeks ago.
	karen wiltshire, 46, sends the babies to sleep by their and playing them the sound of a. it can take anything from a few minutes to half an hour for the tiny newborns to off to sleep. **mother-of-two** from poole, dorset, **has won an award in studio children photography for her work**.	***mother-of-two*** *from sydney,* ***has won an award in studio children photography for her work***. in a youtube video, nathan dailo shows how he gets his three-month-old son seth to sleep in just 42 seconds. the clip has now received 26,000 views since being uploaded two weeks ago.

Table 8: Different guidance signals lead to qualitatively distinct outputs. Tokens that appear in both the guidance and output are **bolded**. Sentences generated differently because of the effect of guidance content are *italicized*.

BertAbs	Ours			
	Sentence	Keyword	Relation	Retrieve
2.117	2.393*	2.347*	2.303*	2.310*

Table 9: Human evaluation of the faithfulness of different model outputs. * indicates significant improvements ($p < 0.001$) over baseline with using bootstrap.

Train	Test	R-1	R-2	R-L
Oracle	Auto	43.78	20.66	40.66
	Oracle	55.18	32.54	52.06
Auto	Auto	41.61	19.04	38.65
	Oracle	43.07	20.79	40.13

Table 10: Using automatically constructed guidance during training degrades the performance significantly.

jecture that this is because when the relevancy between guidance and reference is weakened, the model will not learn to depend on the guidance signals and thus the model will be reduced to the original abstractive summarization baseline.

5 Conclusion

We propose a general framework for guided neural summarization, using which we investigate four types of guidance signals and achieve state-of-the-art performance on various popular datasets. We demonstrate the complementarity of the four guidance signals, and find that our models can generate more novel words and more faithful summaries. We also show that we can control the output by providing user-specified guidance signals.

Given the generality of our framework, this opens the possibility for several future research directions including 1) developing strategies to ensemble models under different guidance signals; 2) incorporating sophisticated techniques such as copy or coverage over the source document, the guidance signal, or both; and 3) experimenting with other kinds of guidance signals such as salient

elementary discourse units.

Acknowledgements

We thank Shruti Rijhwani, Yiran Chen, Jiacheng Xu and anonymous reviewers for valuable feedback and helpful suggestions. This work was supported in part by a grant under the Northrop Grumman SOTERIA project and the Air Force Research Laboratory under agreement number FA8750-19-2-0200. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory or the U.S. Government.

References

Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D Manning. 2015. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*.

Ziqiang Cao, Wenjie Li, Sujian Li, and Furu Wei. 2018. Retrieve, rerank and rewrite: Soft template based neural summarization. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Sumit Chopra, Michael Auli, and Alexander M Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

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 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization. In *Proceedings of the Workshop on Neural Machine Translation and Generation (WNGT)*.

Sebastian Gehrmann, Yuntian Deng, and Alexander M Rush. 2018. Bottom-up abstractive summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Pierre-Etienne Genest and Guy Lapalme. 2012. Fully abstractive approach to guided summarization. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*.

Kai Hong, Mitchell Marcus, and Ani Nenkova. 2015. System combination for multi-document summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Hanqi Jin, Tianming Wang, and Xiaojun Wan. 2020. Semsum: Semantic dependency guided neural abstractive summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Chris Kedzie, Kathleen McKeown, and Hal Daumé III. 2018. Content selection in deep learning models of summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. 2016. Controlling output length in neural encoder-decoders. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Byeongchang Kim, Hyunwoo Kim, and Gunhee Kim. 2019. Abstractive summarization of reddit posts with multi-level memory networks. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset. *arXiv preprint*.

Wojciech Kryściński, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Evaluating the factual consistency of abstractive text summarization. *arXiv preprint*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Chenliang Li, Weiran Xu, Si Li, and Sheng Gao. 2018. Guiding generation for abstractive text summarization based on key information guide network. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*.

Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018a. Generating wikipedia by summarizing long sequences. In *Proceddings of the Intertional Conference on Learning Representatinos (ICLR)*.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the Coference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yizhu Liu, Zhiyi Luo, and Kenny Zhu. 2018b. Controlling length in abstractive summarization using a convolutional neural network. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of the SIGNLL Conference on Computational Natural Language Learning (CoNLL)*.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018a. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018b. Ranking sentences for extractive summarization with reinforcement learning. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Karolina Owczarzak and Hoa Trang Dang. 2010. Overview of the TAC 2010 summarization track. In *Proceedings of the Text Analysis Conference (TAC)*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *Proceddings of the Intertional Conference on Learning Representatinos (ICLR)*.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Itsumi Saito, Kyosuke Nishida, Kosuke Nishida, and Junji Tomita. 2020. Abstractive summarization with combination of pre-trained sequence-to-sequence and saliency models. *arXiv preprint*.

Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium*.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive summarization as text matching. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Qingyu Zhou, Nan Yang, Furu Wei, Shaohan Huang, Ming Zhou, and Tiejun Zhao. 2018. Neural document summarization by jointly learning to score and select sentences. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Chenguang Zhu, William Hinthorn, Ruochen Xu, Qingkai Zeng, Michael Zeng, Xuedong Huang, and Meng Jiang. 2020. Boosting factual correctness of abstractive summarization with knowledge graph. *arXiv preprint*.

A Greedy Selection Algorithm

Algorithm 1 demonstrates how we use an oracle to select a subset of source sentences that have

Algorithm 1 Greedy Selection Algorithm

Input: A source document $\mathbf{x}$ consisting of multiple sentences $\{x_1, \cdots, x_{|\mathbf{x}|}\}$, its reference summary $\mathbf{y}$, and a pre-defined integer N
Output: Oracle-selected highlighted sentences $\mathbf{o}$

```
o = {}
for i = 1, ··· , N do
    max_rouge= 0
    for s in x/o do
        rouge_1, rouge_2 = cal_rouge(o ∪ {s})
        cur_rouge = rouge_1 + rouge_2
        if cur_rouge > max_rouge then
            max_rouge = cur_rouge
            max_sent = s
        end if
    end for
    if max_rouge == 0 then
        break
    end if
    o = o ∪ { max_sent }
end for
return o
```

the highest ROUGE scores with the reference summary. We use a similar algorithm to select the relation triples as well. Concretely, we flatten each relational triple (s, r, o) by concatenating its elements together and treat each concatenated text as a source sentence, then use Algorithm 1 to select the relation triples greedily.

B Analysis

We perform more analysis on CNN/DM in this section. Unless otherwise stated, we use oracle extractions at training time and BertAbs as our base model.

B.1 Controllability

In addition to the qualitative results in the main paper, we also perform a quantitative analysis to demonstrate the controllability of our models.

The quantitative results in Table 3 of the main text already demonstrate to some extent that we can control the model with guidance signals, as guidance signals of better quality can lead to better summaries. To further demonstrate this, we randomly sample guidance signals multiple times and plot the correlation between guidance quality and output quality in Figure 4. We can clearly see that there is a strong correlation between these two variables, indicating the controllability of our model.

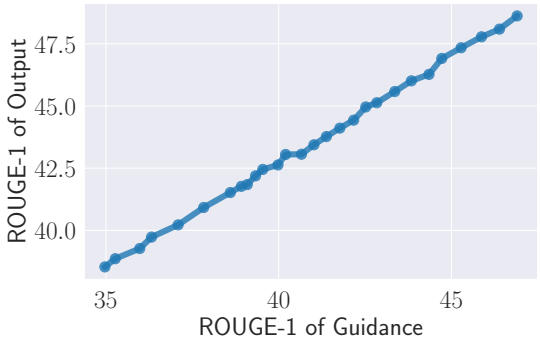

Figure 4: There is a strong correlation between the guidance quality and output quality, demonstrating the controllability of our guided model.

Model	Ref.	Guidance	R-1	R-2	R-L
Sentence	1st	1st	49.49	29.39	46.25
		2nd	28.66	10.09	26.05
	2nd	1st	20.63	5.29	18.25
		2nd	40.33	23.16	37.36
Keyword	1st	1st	40.52	21.06	37.54
		2nd	33.35	14.67	30.60
	2nd	1st	22.49	7.26	20.17
		2nd	28.75	12.65	26.19
Relation	1st	1st	40.45	21.05	37.52
		2nd	33.56	14.65	30.79
	2nd	1st	22.85	7.47	20.47
		2nd	28.48	12.42	25.89
Retrieve	1st	1st	39.32	19.74	36.32
		2nd	33.89	15.29	31.14
	2nd	1st	22.61	7.55	20.34
		2nd	28.31	12.33	25.72

Table 11: We divide each summary reference into two halves and deduce the oracle guidance from them separately. Feeding incompatible guidance signals can lead to degraded performance.

In addition, we try to divide each test reference summary into two halves, then use oracle extraction to obtain guidance signals for both of these two halves and feed them to the model. Table 11 shows that feeding incompatible guidance signals can lead to degraded performance, which further demonstrates that we can control the summary through provision of user-specified inputs.

B.2 Semantic Similarity

To evaluate the semantic similarities between our model outputs and the reference, we also compute the METEOR scores (Banerjee and Lavie, 2005). As shown in Table 12, all of our guided models can outperform BertAbs in temrs of both of METEOR. However, it is surprising that BertExt achieves the

Model	METEOR		#Words (k)
	exact match	+ stem/syn/para	
BertExt	22.24	20.69	828.62
BertAbs	19.43	18.01	669.16
Ours			
Sentence	20.21	18.88	626.73
Keyword	20.16	18.70	700.48
Relation	20.12	18.60	749.30
Retrieve	19.59	18.07	717.22

Table 12: Semantic similarity evaluation. We report results both in exact match mode (rewarding *exact matches* between words) and full mode (rewarding *matching stems*, *synonyms* and *paraphrases* as well).

Model	Train	Test	R-1	R-2	R-L
Sentence	Oracle	Auto	43.78	20.66	40.66
		Oracle	55.18	32.54	52.06
	Auto	Auto	41.61	19.04	38.65
		Oracle	43.07	20.79	40.13
Keyword	Oracle	Auto	42.21	19.36	39.23
		Oracle	45.08	22.22	42.07
	Auto	Auto	41.72	19.15	38.78
		Oracle	41.76	19.25	38.83
Relation	Oracle	Auto	41.40	18.66	38.40
		Oracle	45.96	23.09	42.92
	Auto	Auto	40.29	18.30	37.33
		Oracle	40.67	18.41	37.70
Retrieve	Oracle	Auto	40.88	18.24	37.99
		Oracle	43.69	20.53	40.71
	Auto	Auto	40.86	18.5	37.95
		Oracle	41.45	18.86	38.46

Table 13: Using automatically constructed guidance during training degrades the performance significantly.

best performance, possibly because METEOR has a tendency to favor long summaries.

B.3 Automatic Factual Correctness Evaluation

Besides human evaluation, we have also tried to use factCC (Kryściński et al., 2019)[4] to evaluate the factual correctness of our model outputs automatically. However, as shown in Figure 5, the factCC tool will give the gold reference an accuracy of about 10%. Considering our model is optimized towards the gold reference, the factCC score might not be a good indicator of whether there are factual errors in a generated summary.

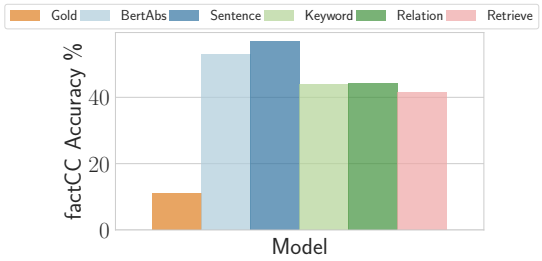

Figure 5: The factCC model will give the gold reference an accuracy of about 10%.

B.4 Necessity of Using Oracles During Training

We have demonstrated in the main paper that it is necessary to use an oracle to select guidance signals during training for highlighted sentence models. In this part, we investigate if this is true for all the three guidance signals as well. Table 13 shows that this methodology will lead to significantly worse performance for other guidance signals as well, which further verifies our hypothesis that when the relevancy between guidance and reference is weakened, the model will not learn to depend on the guidance signals and thus the model will be reduced to the original abstractive summarization baseline.

B.5 Domain Adaptation.

We also evaluate the performance of our highlighted sentence-guided models under domain adaptation settings, namely train a summarization model on one dataset and test it on some other datasets. As shown in Table 14, generally, extractive models can outperform abstractive ones under domain adaptations settings and our model can achieve better performance than abstractive baselines. However, while our model is given the extracted sentences by the extractive model, we still cannot outperform extractive baselines. These negative results indicate that our model may still fail to fully depend on guidance signals when doing adaptation. Possible future directions include dropping out the input documents occasionally during training so that the model can learn to better condition on the guidance.

[4] https://github.com/salesforce/factCC

Method	CNNDM						XSUM						NYT					
	XSUM			NYT			CNNDM			NYT			CNNDM			XSUM		
BertExt	20.55	2.84	15.55	44.80	24.35	41.37	35.98	13.38	32.56	37.35	16.67	33.84	40.18	17.21	36.40	19.93	2.75	14.94
BertAbs	20.39	2.85	15.89	40.99	20.41	37.91	26.31	5.54	21.80	20.60	3.75	16.53	35.77	14.24	32.67	16.11	2.24	12.85
Ours	20.55	2.89	16.00	43.55	21.83	40.51	26.72	5.62	22.08	23.74	3.61	18.37	36.23	14.37	33.15	16.14	2.14	12.92

Table 14: Performance of sentence-guided model under domain adaptation settings. The first row and second row represent source and target domains respectively.