IMPROVING FAITHFULNESS IN ABSTRACTIVE TEXT SUMMARIZATION WITH EDUs USING BART

NARJES DELPISHEH
Bachelor of Science, Iran University of Science and Technology (IUST), 2009

A thesis submitted
in partial fulfilment of the requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

Department of Mathematics and Computer Science
University of Lethbridge
LETHBRIDGE, ALBERTA, CANADA

IMPROVING FAITHFULNESS IN ABSTRACTIVE TEXT SUMMARIZATION WITH EDUs USING BART

NARJES DELPISHEH

Date of Defence: April 25, 2023

Dr. Yllias Chali Thesis Supervisor	Professor	Ph.D.
Dr. John Anvik Thesis Examination Committee Member	Associate Professor	Ph.D.
Dr. Tyler R. Bonnell Thesis Examination Committee Member	Assistant Professor	Ph.D.
Dr. John Sheriff Chair, Thesis Examination Committee	Assistant Professor	Ph.D.

Dedication

To my family, whose support has been invaluable in making this journey memorable.

Abstract

Abstractive summarization aims to reproduce the essential information of a source document in a summary by using the summarizer's own words. Although this approach is more similar to how humans summarize, it is more challenging to automate as it requires a complete understanding of natural language. However, the development of deep learning approaches, such as the sequence-to-sequence model with an attention-based mechanism, and the availability of pre-trained language models have led to improved performance in summarization systems. Nonetheless, abstractive summarization still suffers from issues such as hallucination and unfaithfulness. To address these issues, we propose an approach that utilizes a guidance signal using important Elementary Discourse Units (EDUs). We compare our work with previous guided summarization and two other summarization models that enhanced the faithfulness of the summary. Our approach was tested on CNN/Daily Mail dataset, and results showed an improvement in both truthfulness and good quantity coverage of the source document.

Acknowledgments

I would like to take this opportunity to express my deep gratitude and appreciation to the following individuals who have played a significant role in the completion of this thesis.

First and foremost, I would like to extend my sincerest thanks to my supervisor Dr. Yllias Chali, whose support, guidance, and insightful feedback have been instrumental in shaping my research and academic journey. Your dedication to mentoring and fostering my growth as a researcher have been truly inspiring, and I am honored to have had the privilege of working under your supervision.

I would also like to extend my gratitude to my two committee members Dr. John Anvik and Dr. Tyler R. Bonnell, whose constructive criticism and feedback have been constructive in refining my research. Your evaluation and suggestions have challenged me to improve the quality of my work.

I would like to thank my thesis defense meeting chair Dr. John Sheriff, for taking the time to evaluate my work and provide valuable feedback during my thesis defense. Your insights and comments have helped me better understand the strengths and limitations of my research and presentation.

Additionally, I would like to appreciate the Alberta Innovative Scholarship program for awarding me the scholarship that enabled me to pursue my research. The financial support provided by the program has been crucial in helping me focus on my research and academic goals.

I would also like to express my gratitude to Digital Research Alliance of Canada for providing me with access to their high-performance computing resources. These resources were integral in conducting the computational work required for my research. Moreover, I

would like to thank the technical support team for their complete responses in addressing any issues or questions that I had during my work.

Finally, I would like to express my heartfelt gratitude to my family for their support, encouragement, and understanding throughout this challenging journey. Your love and support have been my source of inspiration and motivation, and I am grateful for all that you have done for me.

Once again, I extend my sincere appreciation to all those who have helped me throughout this process, and I am grateful for their contributions to my academic journey.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Written text has been the most popular form of information since the advent of computers. However, with the growth of textual information on the Internet, processing and summarizing large amounts of information has become overwhelming. Automatic summarization can extract important information from a text and create an informative summary.

There are two types of summarization based on the summary production approach: extractive and abstractive summarization. In extractive summarization, the focus is on selecting important sentences from the document and presenting them as they are, which can sometimes result in a less coherent summary. On the other hand, in abstractive summarization, the aim is to extract important information from the document and rephrase it into a summary. This approach improves the coherence of the summary, but it is a more challenging task as it requires a comprehensive understanding of natural language.

With the advent of the deep learning approach, new summarization systems (Rush et al., 2015; Paulus et al., 2018; Lewis et al., 2020; Zhang et al., 2020a) have emerged, resulting in a significant improvement in both automatic and manual evaluation scores. The sequence-to-sequence model (Sutskever et al., 2014) with an attention-based mechanism (Bahdanau et al., 2014; Luong et al., 2015) is a popular deep learning architecture that has contributed to this performance boost.

The seq2seq model consists of two neural networks: the encoder and the decoder. The encoder takes in a sequence of source tokens and encodes them into a single vector representation. The decoder then takes this vector and decodes it into a new sequence of target

tokens. This approach allows the neural model to effectively capture relevant information from the text and generate a new sequence of words that contains only salient information, resulting in a shorter summary.

In addition to the advancements made in deep learning approaches, pre-trained language models such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019) trained on massive amounts of unlabeled data have also played a significant role in improving the performance of summarization systems. By leveraging pre-trained language models, neural models can acquire general knowledge across many topics and improve the fluency of language generation. However, despite these improvements, abstractive summarization still faces challenges such as hallucination and unfaithfulness. Hallucination occurs when the model generates tokens that are irrelevant to the document. Unfaithfulness refers to text that is inaccurate or misleading compared to the original source text.

To tackle the issues of hallucination and Unfaithfulness in language generation, we implemented a mechanism that employs a guidance signal to regulate the important EDUs (Elementary Discourse Units). To evaluate the effectiveness of our approaches, we conducted two separate kinds of experiments. In the first experiment we guide the summarization model using EDUs. Our results indicate that this approach outperforms other previous models with other guidance. In the second experiment, we compared our guided EDUs summarization with other previous models that aimed to improve faithfulness in summarization. Our model produces more informative summarization and is more successful in this regard. The results were evaluated using the effective faithfulness criteria.

1.1 Contributions

The contribution of this thesis to the research on improving faithfulness in text summarization is presented as follows:

- We demonstrate that incorporating important EDUs, in addition to the source text, into an abstractive text summarization model can be an effective approach for gener-

ating faithful summaries.

- We employ EDUs to guide BART text summarization and fine-tune the model on Compute Canada's Alliance server.
- We show that using EDUs or a combination of EDUs and sentences as a guidance signal can outperform other guidance signals, such as sentences, in generating more accurate summaries.
- We compare our model with previous work on faithful summary generation and demonstrate a significant improvement in multiple evaluation metrics, resulting in increased faithfulness and informativeness of summaries.

1.2 Overview

The objective of this thesis is to extensively investigate the topic of faithful abstractive summarization by utilizing guided summarization. Chapter 2 presents general concepts such as a comprehensive review of deep learning techniques, automatic summarization, and evaluation methods for summarization. In Chapter 3, we review the previous studies related to faithfulness and EDUs. In Chapter 4, we introduce our proposed guided summarization model to enhance faithfulness of the summarization. Chapter 5 discusses our work and implementation on CNN/Daily Mail dataset using BART summarization. Finally, we synthesize the findings and conclusions drawn from the research.

Chapter 2

Background

This section offers information and clarification that is pertinent to this study.

2.1 Natural Language Processing

Natural Language Processing (NLP) is a field of computer science that focuses on the interaction between computers and human language. It involves the use of computational methods to analyze, generate, and understand human language in all its forms, including written text, spoken language, and sign language. NLP has a wide range of applications, including text summarization, machine translation, speech recognition, sentiment analysis, information extraction, and text classification, to name just a few. With the growing volume of textual data on the internet, NLP has become increasingly important for extracting valuable insights from unstructured data.

2.1.1 Text Preprocessing

Text preprocessing is a crucial step in NLP that involves cleaning and transforming raw text data into a format that can be easily processed by algorithms. This includes tasks such as tokenization, stemming, and stopword removal. Studies have shown that proper text preprocessing can greatly improve the performance of NLP models (Zhang and Wallace, 2015).

2.1.2 Named Entity Recognition (NER)

Named Entity Recognition is a subfield of NLP that deals with the identification and extraction of entities in text such as names of people, organizations, locations, and other important information. This task is usually performed using algorithms that learn to recognize patterns in the text that correspond to different types of entities (Nadeau and Sekine, 2007).

2.1.3 Sentiment Analysis

Sentiment analysis is the task of identifying the sentiment or emotion expressed in text, such as positive, negative, or neutral. Sentiment analysis has many applications, including customer feedback analysis and social media monitoring. Many sentiment analysis models use machine learning algorithms, such as support vector machines (SVMs) and deep neural networks (Tang et al., 2015).

2.2 Subword Models

Tokenization is a crucial step in natural language processing, where a text is broken down into smaller chunks such as words or punctuation. This process helps in creating a vocabulary that comprises unique words in the corpus. However, when dealing with large corpora, the vocabulary size can become too large, leading to issues with memory and computation time. To avoid this, the size of the vocabulary needs to be limited, and infrequent tokens represented with an out-of-vocabulary token. Instead of tokenizing text into words, it can be done into characters, but this approach can be too fine-grained for models to capture the meaning of sentences. Therefore, a hybrid approach that divides unique words into subwords is used. This approach balances the number of vocabulary entries and subword tokenization. Algorithms for subword tokenization capture prefixes and suffixes, and there is no chance of out-of-vocabulary words with this approach.

For example, the word ”unbelievable” can be broken down into subwords such as ”un-”,

”be-”, ”liev-”, and ”-able” using algorithms like byte-pair encoding or WordPiece. These subwords help the model to understand the context better, as it can recognize them as meaningful parts of the word.

2.2.1 Byte-Pair Encoding (BPE)

Byte-Pair Encoding (BPE) (Sennrich et al., 2016) is a subword tokenization algorithm used to break down words into smaller parts for natural language processing tasks. The algorithm starts by setting a vocabulary with entries that have only one character observed in the corpus. It then tokenizes the corpus into words and computes the frequencies of each unique word. The algorithm repeatedly learns merge rules by counting the frequencies of all possible combinations of two characters consecutively seen in the corpus. The most frequent character pair is selected, and the resulting subword is added to the vocabulary as a new entry. This process continues until the vocabulary size reaches the desired number. With the merge rules learned, the algorithm can tokenize new text into subwords.

2.2.2 WordPiece

WordPiece (Schuster and Nakajima, 2012) is another subword tokenization algorithm that is widely used for natural language processing tasks such as the BERT model. This algorithm is similar to BPE, but it chooses character pairs that maximize the likelihood of the training data if the pair is added to the vocabulary. This is done in order to optimize the algorithm’s performance on the specific language data it is being trained on. The algorithm selects the character pair with the highest probability of two characters divided by the probability of the first character followed by the second character. This process iteratively continues until the vocabulary size reaches the desired value.

2.2.3 Unigram Language Model

The Unigram language model (Kudo, 2018) is another subword tokenization algorithm that works differently from the previous ones. Instead of merging smaller parts, it trims

larger chunks into smaller ones to create a smaller vocabulary. At each training step, the algorithm calculates the loss values for the entire training data given the current vocabulary and a unigram language model. These loss values represent the measure of the difference between the predicted and actual target output of the model. The algorithm then tries to find out how much the overall loss would increase if a symbol was removed from the vocabulary for each character in the vocabulary. The algorithm then removes the 10-20% of characters with the lowest loss values, which have the least effect on the overall loss, and repeats this process until the desired vocabulary size is reached. This algorithm does not have any merge rules, and there can be more than one way to tokenize a text after training. The probabilities of each possible tokenization are saved along with the vocabulary, which allows for the computation of the probabilities of each token in the training corpus.

2.3 Word Embedding

Word embedding is a technique used in natural language processing that transforms words into numerical representations, making them easier to process for machine learning algorithms. There are various methods for creating word embeddings, including one-hot encoding, embedding matrix, Word2vec, and GloVe.

2.3.1 One-hot Encoding

One-hot encoding is a simple way of creating a word embedding, where each word is represented as a vector with only one ”hot” or ”on” bit, and all the other bits are set to zero. The position of the ”hot” bit indicates the index of the word in the vocabulary. For instance, the word ”cat” can be represented as a one-hot encoded vector of length 10,000, where the bit at index 3 would be ”on” and all the other bits are ”off.” The example for one-hot encoding is: OH(w) = [0, 0, ..., 1, ..., 0] where w is the word, and the vector contains all zeros except for a one at the index of the word in the vocabulary.

2.3.2 Embedding matrix

An embedding matrix is a dense matrix where each row represents a word embedding. The matrix is learned during training, and the resulting embeddings are more informative than one-hot encodings as they capture the relationships between words. E = [e_1, e_2, ..., e_n] is embedding matrix where e_1, e_2, ..., e_n are the word embeddings. For instance, in a model trained on a large text corpus, the embeddings for ”cat” and ”dog” are likely to be located close together in the embedding space, while the embeddings for ”cat” and ”computer” are likely to be far apart. This is because ”cat” and ”dog” have a similar meaning and are often used in similar contexts, whereas ”cat” and ”computer” are not as closely related.

2.3.3 Word2vec

Word2vec (Mikolov et al., 2013a) is a neural network-based technique for generating word embeddings. It creates embeddings by predicting the context of a word given a window of neighboring words. The resulting embeddings capture the semantic and syntactic relationships between words. For example, ”king” and ”queen” will have similar embeddings because they are related words. The formula for Word2vec is:

$$P(w_c|w_i) = \frac{\exp({v_{w_c}}^T v_{w_i})}{\sum_{w_j \in V} \exp({v_{w_j}}^T v_{w_i})} \tag{2.1}$$

where w_c is the context word, w_i is the input word, V is the vocabulary, and v_w is the embedding for word w.

2.3.4 GloVe

GloVe (Global Vectors) (Pennington et al., 2014) is another technique for generating word embeddings that combines the global co-occurrence statistics of words with the local context of the words. It creates embeddings by optimizing a global objective function that captures the relationships between words based on their co-occurrence probabilities. The resulting embeddings are also capable of capturing the semantic and syntactic relationships

between words.

$$J = \sum_{i=1}^{V} \sum_{j=1}^{V} f(P_{ij})(\log P_{ij} - \log \hat{P}_{ij})^2 \tag{2.2}$$

where V is the vocabulary, f is a weighting function, P_{ij} is the co-occurrence probability of words i and j, and $\hat{P}_{ij}$ is the probability of the word j appearing in the local context (within a certain distance) of word i.

2.4 Deep Neural Networks

Deep neural networks (DNNs) use complex mathematical models to process data. Deep learning has shown promising results in various natural language processing tasks such as text summarization, machine translation, sentiment analysis, and text classification.

2.4.1 Feed-Forward Neural Networks (FFNNs)

Feed-Forward Neural Network (FFNN) is a type of artificial neural network that treats input features as unique and independent of one another. In other words, the output from each neuron in a layer is based solely on the input values, without any feedback from subsequent layers. FFNN is commonly used in tasks such as classification, regression, and prediction, where the output layer produces probability estimates for the input data (Hagan et al., 1996). The multi-layer perceptron (MLP) is a specific type of FFNN that consists of one or more hidden layers. Training an MLP involves adjusting the weights and biases of the connections between the neurons to minimize the error between the predicted output and the true output.

2.4.2 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a type of neural network architecture designed to analyze data with a grid-like structure, such as images or time series data (Goodfellow et al., 2016). The key idea behind CNNs is to apply filters, also known as kernels,

to extract features from the input data. The filters slide over the input data, performing convolution operations, to generate a set of feature maps. The feature maps capture the presence of different features at different spatial locations of the input data. The feature maps are then passed through activation functions and pooling layers, which reduce the dimensionality of the feature maps while preserving the essential information (LeCun et al., 2015). The output of these layers is then fed into fully connected layers, which generate the final predictions. Various modifications to CNNs have been proposed to improve their performance. For example, the use of residual connections in ResNet has shown significant improvements in training deep CNNs (He et al., 2016).

2.4.3 Recurrent Neural Networks (RNNs)

Recurrent Neural Networks (RNNs) are a type of neural network architecture that allows information to be retained and processed over time. Unlike feed-forward neural networks, RNNs have loops that enable the network to use the output from the previous step as input to the current step. This architecture makes RNNs well-suited for sequential data, such as time-series data, speech, and natural language, where the input depends on the previous input values. However, RNNs suffer from the vanishing gradient problem, where the gradients become very small or vanish over time, making it difficult to learn long-term dependencies (Hochreiter and Schmidhuber, 1997). As a result, various types of RNNs, such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), have been proposed to address this issue and have shown significant improvements in performance in many applications.

2.4.4 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a type of Recurrent Neural Network (RNN) architecture designed to address the vanishing gradient problem that arises in traditional RNNs (Hochreiter and Schmidhuber, 1997). LSTM uses a gating mechanism to selectively forget or remember information from previous timesteps, which helps the network learn long-

term dependencies in sequential data (Graves, 2013). The gating mechanism consists of three gates: the input gate, the forget gate, and the output gate. The input gate controls how much new information is added to the memory cell, the forget gate controls how much information is discarded from the memory cell, and the output gate controls how much information is output from the memory cell to the next timestep.

2.4.5 Gated Recurrent Unit (GRU)

Gated Recurrent Unit (GRU) is a type of Recurrent Neural Network (RNN) architecture that was introduced as a simpler alternative to Long Short-Term Memory (LSTM) (Chung et al., 2014). Like LSTM, GRU is designed to address the vanishing gradient problem in traditional RNNs by using a gating mechanism to selectively update the hidden state based on the input and previous hidden state. However, unlike LSTM, GRU has only two gates: the reset gate and the update gate. The reset gate controls how much of the previous hidden state should be forgotten, while the update gate controls how much of the new hidden state should be retained (Cho et al., 2014). This simpler architecture results in fewer parameters, making GRU computationally less expensive than LSTM while achieving comparable performance. GRU has shown particular promise in tasks where training data is limited, as it is less prone to overfitting than other RNN architectures (Greff et al., 2017).

2.4.6 Bidirectional Recurrent Neural Networks (BRNNs)

Bidirectional Recurrent Neural Networks (BRNNs) are a type of Recurrent Neural Network (RNN) architecture that processes sequential data in both forward and backward directions (Schuster and Paliwal, 1997). This allows the network to capture both past and future contexts of each input, which is useful in tasks such as speech recognition and natural language processing. BRNNs consist of two RNNs, one processing the input sequence from the beginning to end, and the other processing it from the end to beginning. The hidden states of the two RNNs are concatenated at each timestep to generate the final output (Graves and Schmidhuber, 2005).

2.5 Encoder-Decoder Model

The Encoder-Decoder Model consists of two main parts: an encoder and a decoder. The encoder processes the input sequence and produces a fixed-length representation of the input, while the decoder generates the output sequence based on the encoded representation. One of the advantages of the Encoder-Decoder Model is that it can handle variable-length input and output sequences.

The Encoder-Decoder Model has been shown to be effective in many natural language processing tasks. For example, in machine translation, the model can be trained to translate a sentence from one language to another by encoding the source sentence into a fixed-length representation and then decoding it to generate the target sentence (Sutskever et al., 2014). In text summarization, the model can be trained to generate a summary of a longer text by encoding the input text and then decoding a summary from the encoded representation (See et al., 2017).

2.6 Sequence Model Decoders

In a sequence-to-sequence task, the goal of the model is to generate the optimal output sequence $\hat{Y}^*$ given a input sequence X, where Y is the desired output. This can be expressed mathematically as:

$$\hat{Y}^* = argmaxProb(Y|X) \tag{2.3}$$

However, the search space for finding the optimal output sequence can be very large. To address this, there are several techniques available to reduce the number of candidate sequences and generate the final output sequence.

2.6.1 Greedy Decoding

Greedy decoding is an algorithm that selects the word with the highest probability from the model's vocabulary to generate the output sequence. However, this approach does not

guarantee that the output sequence will be the most probable one overall. It only considers a limited number of possible words, which can lead to errors. If the algorithm selects an incorrect word at any point in the sequence, it can have a significant impact on the rest of the generated words.

2.6.2 Beam Search

Beam search is an algorithm used to find the most probable sequence during decoding in Neural Machine Translation. At each step of the decoding process, the algorithm selects the top β most likely candidates from the model's vocabulary, where β is a hyperparameter of the model. Increasing β can improve the precision of the algorithm, but also makes it more computationally expensive, as more candidates need to be estimated at each word location. Nonetheless, this method is commonly used in Neural Machine Translation as it can lead to better results than other algorithms.

2.7 Attention Mechanism

Attention mechanism allows the model to focus on the most relevant parts of the input sequence when generating the output sequence. In other words, the attention mechanism can learn to assign different weights to different parts of the input sequence based on their importance for generating the output sequence (Bahdanau et al., 2014). The attention mechanism has been shown to be very effective in many natural language processing tasks. For example, in machine translation, the mechanism can be used to align the source and target sentences and focus on the relevant parts of the source sentence when generating the target sentence (Luong et al., 2015). In text summarization, the mechanism can be used to select the most important sentences or phrases from the input text and use them to generate the summary (Rush et al., 2015).

2.8 Transformer

The Transformer is a neural network architecture that was introduced by Vaswani et al. (2017), as an alternative to recurrent neural networks for sequence-to-sequence tasks. The Transformer architecture is based on the self-attention mechanism, which allows the model to attend to different parts of the input sequence to make predictions.

2.8.1 Positional Embeddings

Positional Embeddings are added to the input sequence to indicate the position of each token in the sequence. The positional embeddings are added to the learned embeddings of each token before feeding them into the model. The positional embeddings are learned during training and allow the model to encode the position of each token in the sequence.

2.8.2 Multi-head Self-Attention

Multi-head self-attention is the key component of the Transformer architecture. It allows the model to attend to different parts of the input sequence to make predictions. Self-attention mechanism computes a weighted sum of the values, where the weights are computed as the softmax of the dot products between each query vector and all key vectors. The weights are then used to compute the self-attended representation of the input sequence. The formula for self-attention is:

$$\text{Self-Attended} = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.4}$$

Here, $Q \in \mathbb{R}^{n\times d_k}$, $K \in \mathbb{R}^{n\times d_k}$, and $V \in \mathbb{R}^{n\times d_v}$ are the query, key, and value matrices, respectively. The dot product between Q and K^T is divided by the square root of the dimension of the key vectors d_k to reduce the variance of the dot products.

Multi-head self-attention is obtained by performing self-attention multiple times in parallel on different subspaces of the input sequence. Each of these parallel self-attention operations is called a head. The output of the multi-head self-attention operation is the

concatenation of the outputs of all heads.

$$\text{MultiHead}(Q,K,V) = \text{Concat}(\text{head}_1, \text{head}_2, \ldots, \text{head}_h)W^O \tag{2.5}$$

where $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$ is the i-th attention head, and $W_i^Q \in \mathbb{R}^{dmodel \times d_k}$, $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$, and $W^O \in \mathbb{R}^{hd_v \times d_{model}}$ are learned weight matrices.

The *Concat* function concatenates the outputs of all attention heads along the last dimension. The output of $MultiHead(Q,K,V)$ has the same shape as the input Q

2.8.3 Architecture of transformer

The Transformer architecture (Figure 2.1) consists of an encoder and a decoder. The encoder processes the input sequence and produces a sequence of hidden states, while the decoder generates the output sequence based on the encoded representation. Both the encoder and the decoder are composed of a stack of self-attention layers followed by a feed-forward neural network layer. The self-attention layers allow the model to attend to different parts of the input sequence, while the feed-forward neural network layers allow the model to make non-linear transformations of the hidden states. The architecture also includes residual connections and layer normalization, which help to stabilize the training process and improve the performance of the model (Vaswani et al., 2017).

The multi-head self-attention mechanism is obtained by performing this self-attention operation in parallel multiple times, with different learned Q, K, and V matrices for each ”head”. The outputs of all heads are concatenated and passed through a linear layer to obtain the final self-attended representation.

2.9 Text Summarization

Text summarization involves two main approaches: extractive summarization and abstractive summarization. Extractive summarization involves selecting a subset of sentences from one or more source documents and combining them into a summary without altering

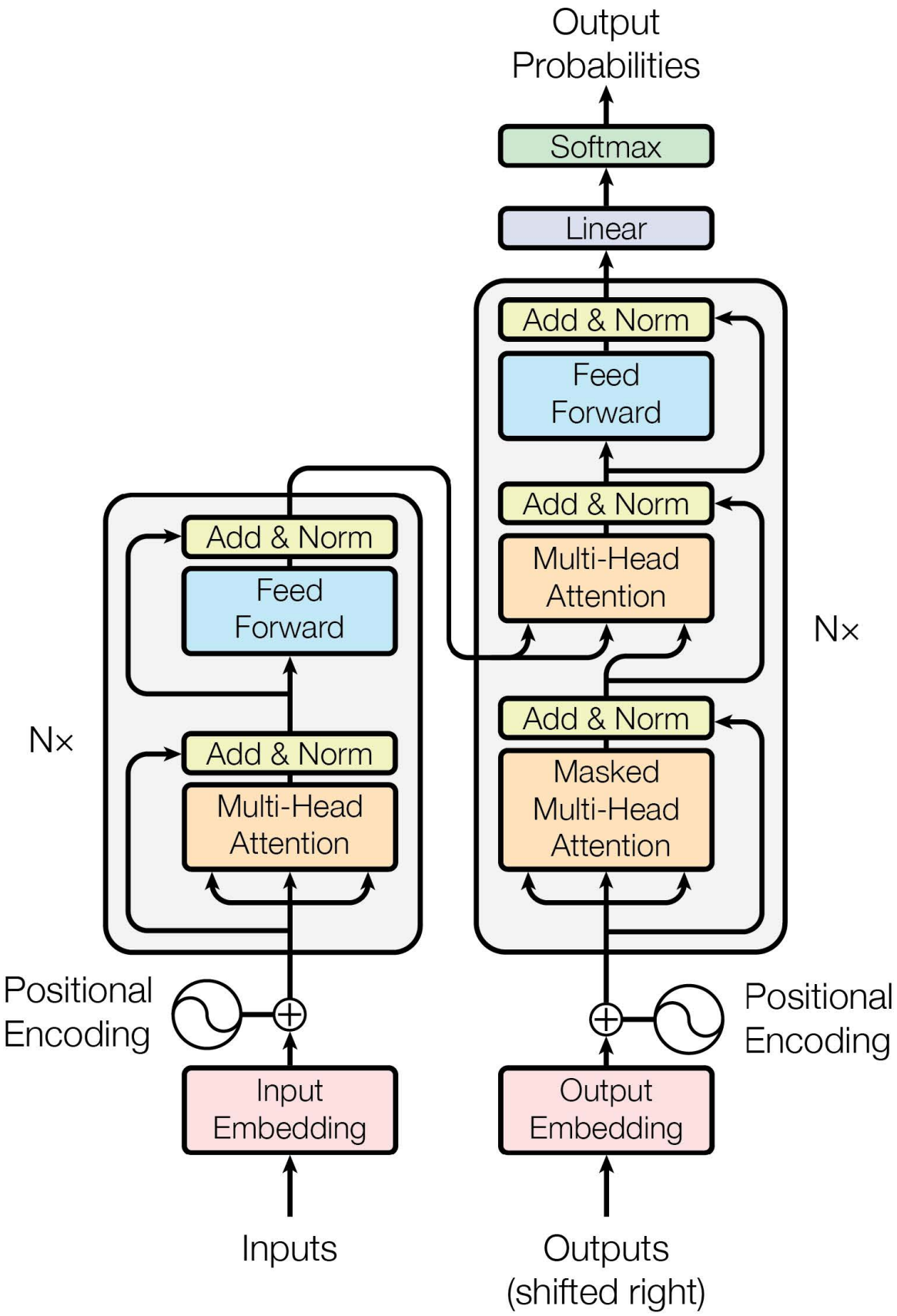

Figure 2.1: Transformer architecture (Vaswani et al., 2017)

the wording. On the other hand, abstractive summarization involves condensing the key information from the source document into a summary by rephrasing it and possibly introducing new words. Abstractive summarization is more challenging than extractive summarization because it requires not only identifying important textual units but also capturing the underlying semantic information. Currently, most abstractive summarization systems use the seq2seq approach. The next subsections will discuss various techniques used for extractive and abstractive summarization.

2.9.1 Extractive Summarization

In extractive summarization, the goal is to select a subset of sentences from the source document that contains the most relevant information while still fitting within a given length constraint. This task can be seen as a knapsack problem, which is known to be NP-hard. Additionally, it is challenging to determine a correct valuation for each sentence. Evaluating extractive approaches is also difficult, as each method has its own evaluation approaches. However, some previous works have addressed the issues with extractive summarization, such as developing new methods for sentence valuation and creating evaluation metrics. Despite these efforts, extractive summarization is less popular than abstractive summarization, as it is challenging to produce a fluent and coherent summary using this approach. Furthermore, most available summarization datasets contain abstractive gold summaries, which makes them more suitable for evaluating abstractive summarization approaches.

Sentence selection: Sentence selection will find a subset of sentences that have desirable features such as feature scores or low redundancy coverage. Various studies such as studies were done by Carbonell and Goldstein (1998) and Ren et al. (2016), have explored this approach. However, selecting sentences based on these properties alone may not guarantee the coherence of the summary.

Sentence Compression: In sentence compression, certain tokens are removed from a sentence to make it shorter while retaining the important information. Almeida and Martins

(2013) and Berg-Kirkpatrick et al. (2011) are some of the researchers who have worked on this approach. These models have two objectives: selecting sentences and deleting words. These models are trained together to achieve an optimal balance between the two objectives.

2.9.2 Abstractive Summarization

Abstractive summarization is a technique used to create a summary that is generated by methods such as paraphrasing, generalization, and deductive reasoning. It is more complicated than the extractive approach and requires a deeper understanding of natural language. Recent advancements in the field of abstractive summarization, such as the seq2seq approach and pre-trained language models, have brought significant improvements.

Sentence Fusion: Sentence fusion is a method used to produce a summary by combining and rephrasing related information. This approach typically involves using intermediate representations such as dependency trees by Filippova and Strube (2008) or abstract meaning representation by Liu et al. (2015) to facilitate the merging process. However, there are some limitations to this approach, including the possibility of noise introduction from the intermediate representation parser, and difficulty in generating fluent sentences from intermediate representations.

Neural model: The development of the seq2seq model with attention mechanism (Luong et al., 2015) has resulted in significant advancements in automatic summarization approaches. This approach encodes the source document sequence into a hidden representation and then decodes the target summary sequence. The availability of large summarization datasets such as CNN/DM (Hermann et al., 2015), Gigaword (Graff et al., 2003), and XSUM (Narayan et al., 2018), among others, has contributed to the progress of neural summarization. Furthermore, the introduction of pre-trained language models and seq2seq pre-trained language models has further enhanced the performance of state-of-the-art systems. However, despite its progress, the neural model approach still has some limitations. Generated summaries may have grammatical errors, hallucinations, and other linguistic in-

accuracies (Wiseman et al., 2017). Additionally, this approach requires large datasets to perform well.

2.10 Pre-trained Language Models

Two main strategies for using pre-trained models are feature-based and fine-tuning. The feature-based approach involves pre-training a model on a large, unlabeled dataset and using it as a supplementary source of features for the downstream task. In contrast, the fine-tuning approach uses the pre-trained model as a starting point and performs further parameter fine-tuning on the downstream task. While the feature-based approach has been used in models such as word2vec and ELMo (Mikolov et al., 2013b; Peters et al., 2018), recent research has shown that the fine-tuning approach is more effective (Devlin et al., 2019) and is the most commonly used approach in current research. Therefore, this discussion will focus on pre-trained models that utilize the fine-tuning strategy.

2.10.1 Bidirectional Encoder Representations from Transformers (BERT)

The BERT model consists of two main steps: pre-training and fine-tuning. In the pre-training step, the model is trained on a large dataset without any labeled data. In the fine-tuning step, the pre-trained model is further trained on a labeled dataset specific to the downstream task.

BERT uses WordPiece embeddings (Wu et al., 2016) for input representation and has been trained on two unsupervised tasks: Masked Language Model (MLM) and Next Sentence Prediction (NSP). MLM involves randomly masking some tokens and predicting them, while NSP predicts whether the next sentence is related to the previous sentence. However, the BERT model is not optimally trained, and RoBERTa has been introduced to address this issue. RoBERTa improves upon BERT in four ways: longer training time, larger batch sizes, removal of the next sentence prediction objective, and dynamically changing mask patterns during training. RoBERTa (Liu et al., 2019) outperforms BERT

on both GLUE and SQuAD, benchmark datasets, indicating its superior performance.

2.10.2 BART

BERT and RoBERTa are pre-trained encoder models that are not suitable for summarization tasks, unlike BART (Lewis et al., 2020), which is an encoder-decoder pre-trained model specifically designed for summarization. This difference in architecture enables BART to generate output sequences, while BERT can only classify input sequences. BART uses the Transformer architecture like BERT, but the activation layer is replaced with the GeLU (Hendrycks and Gimpel, 2016) instead of ReLU. GeLU (Gaussian Error Linear Unit) is a non-linear activation function that is designed to be smoother and more continuous than ReLU (Rectified Linear Unit). This difference in activation function is a significant factor in enabling BART to generate output sequences, whereas BERT and RoBERTa can only classify input sequences. Additionally, BART uses a combination of denoising auto-encoders and sequence-to-sequence objectives, while BERT uses a masked language modeling objective.

Model Architecture and Data Representation: BART is based on a sequence-to-sequence transformer architecture that uses both an encoder and decoder to generate output sequences. The model's input is represented using byte pair encodings (BPE) and positional embeddings. BART also uses a special symbol called "mask" to represent tokens that are masked during training.

Pre-training BART: BART is pre-trained using a combination of denoising autoencoder and sequence-to-sequence objectives. During pre-training, the input sequences are randomly corrupted using various techniques such as Token Masking (where random tokens are replaced with masked elements), Token Deletion (where some tokens are randomly deleted from the input), Text Infilling (where a random span of text is masked, with the length of the span sampled from a Poisson distribution), Sentence Permutation (where sentences are randomly shuffled), and Document Rotation (where random tokens are se-

lected and rotated to the beginning of the document). The corrupted sequences are then fed into the model, and the model is trained to reconstruct the original sequence. This pre-training objective helps BART to learn rich representations of the input sequences, which can be fine-tuned for a wide range of tasks.

Fine-tuning BART: BART can be fine-tuned for various natural language processing tasks such as Sequence Classification Tasks, Token Classification Tasks, and Sequence Generation Tasks. In Sequence Classification Tasks, BART is fine-tuned to classify input sequences into predefined categories. In Token Classification Tasks, BART is fine-tuned to predict the class of each token in the input sequence. In Sequence Generation Tasks, BART is fine-tuned to generate output sequences given input sequences.

2.10.3 Text-to-Text Transfer Transformer (T5)

T5 is a pre-trained model designed to unify various pre-trained models framework approaches. T5's approach (Raffel et al., 2019) to pre-training is similar to BART, where both input and output are in the form of a sequence of tokens. T5 uses Transformers as its component, but it differs from the standard architecture in a few ways. For instance, T5 removes the Layer Normalization bias and places the layer normalization outside the residual path. Moreover, T5 uses a position embedding scheme where a scalar is added to the corresponding logit used for computing the attention weights.

2.10.4 Pegasus

Pegasus (Zhang et al., 2020a) is a pre-trained language model designed for text summarization. It is built upon the same text-to-text architecture as BART and T5, where the input and output are in the form of sequences of tokens. However, Pegasus introduces a novel pre-training objective called Gap Sentences Generation (GSG), which is specific to summarization. GSG involves masking entire sentences from a document and concatenating the gap sentences into a pseudo-summary. Pegasus uses three different approaches to select the masked sentences: random, lead, and principal. The random approach uniformly

selects sentences at random, the lead approach selects only the top-m sentences, and the principal approach selects sentences that are highly aligned with the rest of the document based on ROUGE scores.

2.11 Evaluation metrics in NLP

In the earliest attempt to evaluate automatic summarization by Edmundson (1969), human annotators were asked to judge the quality of the system-produced summary in terms of its similarity to the gold summary and the information it contained. However, there was no consensus on how to evaluate summarization at that time. Summarization evaluation became the focus of researchers during DUC 2001 and continued to be refined in subsequent years.

2.11.1 Criteria for Summarization

In order to establish criteria for summarization, it is important to first determine the purpose of evaluating a summary. There are two main purposes for evaluating natural processing systems (Sparck Jones and Galliers, 1995): evaluating the objective of the system and evaluating the function of the system in relation to an external task. The first purpose is called intrinsic evaluation, which evaluates the summary's inherent quality, such as its fluency, coherence, and informativeness. The second purpose is called extrinsic evaluation, which measures how well the summarization system performs in a specific task. For example, we can use a Q&A approach (Gorinski and Lapata, 2015) to measure the summary's effectiveness in answering questions derived from the source document. However, extrinsic evaluation adds complexity to the evaluation task, so this section will only focus on criteria related to intrinsic evaluation, which are more relevant to the task at hand.

Coherence and Readability: In extractive summarization, multiple sentences are extracted from the source document to form a summary. However, this can result in a lack of coherence if the context of each extracted sentence is not considered. Some works (Paulus

et al., 2018) only use readability as a criterion, where coherence is implied. Readability is a high-level criterion that judges rate based on various aspects, such as sentence coherence, fluency, naturalness, and the presence of dangling anaphors.

Informativeness: In text summarization, informativeness refers to the amount of information that is covered from the source document. An informative summary covers a lot of information from the original text. However, the challenge lies in keeping the length of the summary short while ensuring that it has sufficient coverage. The informativeness criterion is also known as coverage or content quality. Automatic metrics such as ROUGE (Lin, 2004) measures informativeness based on the recall of information between the reference summary and the system-generated summary.

Correctness: The accuracy of the information included in a summary is an important criterion that needs to be considered. This can be a challenge for advanced summarization techniques that use deep learning as there is a risk that the generated summary may contain incorrect information, such as wrong dates, numbers, and facts. There are some studies (Maynez et al., 2020b; Kryściński et al., 2020) that have proposed methods for evaluating the correctness of a summary, which utilize techniques such as textual entailment and weakly supervised learning. These methods can help ensure that the summary generated by the system is factually correct.

Compression: Compression refers to summarization methods that involve compressing the original text into a shorter version by removing redundant information. In some studies such as those by Clarke and Lapata (2008), Berg-Kirkpatrick et al. (2011), and Almeida and Martins (2013), compression is used as a criterion to evaluate the quality of the summary. This is typically measured as the ratio of the length of the compressed sentence to the length of the original sentence.

2.11.2 ROUGE

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) is an automatic evaluation metric commonly used for assessing the quality of generated summaries in NLP (Lin, 2004). ROUGE measures the overlap between the generated summary and one or more reference summaries using precision, recall, and F-measure scores.

There are two main types of ROUGE score: $ROUGE-N$ and $ROUGE-L$. $ROUGE-N$ measures the $n-gram$ overlap, i.e., the number of continuous $n-grams$ (sequences of n words) in the generated summary that overlap with the reference summary. $ROUGE-N$ can be calculated using the following formula

$$\text{ROUGE-}N(precision) = \frac{\sum_{S \in \text{reference summaries}} \sum_{n\text{-grams} \in S} \text{Count}_{\text{match}}(n\text{-grams})}{\sum_{S \in \text{generated summaries}} \sum_{n\text{-grams} \in S} \text{Count}(n\text{-grams})} \tag{2.6}$$

$$\text{ROUGE-}N(recall) = \frac{\sum_{S \in \text{reference summaries}} \sum_{n\text{-grams} \in S} \text{Count}_{\text{match}}(n\text{-grams})}{\sum_{S \in \text{reference summaries}} \sum_{n\text{-grams} \in S} \text{Count}(n\text{-grams})} \tag{2.7}$$

$$\text{ROUGE-}N(f-measure) = \frac{(1+\beta^2) \cdot \text{ROUGE-}n(precision) \cdot \text{ROUGE-}N(recall)}{\beta^2 \cdot \text{ROUGE-}n(precision) + \text{ROUGE-}N(recall)} \tag{2.8}$$

where $\text{Count}_{\text{match}}(n\text{-grams})$ is the count of n-grams that match between the generated summary and the reference summary, and $\text{Count}(n\text{-grams})$ is the count of all $n-grams$ in the reference summary.

$ROUGE-L$, on the other hand, measures the longest common subsequence (LCS) of words between the generated summary and the reference summary. The LCS is the longest sequence of words that appears in both the generated summary and the reference summary but with no consecutive matching. $ROUGE-L$ can be calculated using the following

formula:

$$\text{ROUGE-L(precision)} = \frac{\sum_{i=1}^{u} \text{LCS}_{\cup}(r_i, C)}{n} \tag{2.9}$$

$$\text{ROUGE-L(recall)} = \frac{\sum_{i=1}^{u} \text{LCS}_{\cup}(r_i, C)}{m} \tag{2.10}$$

$$\text{ROUGE-L(f-measure)} = \frac{(1+\beta^2) \cdot \text{ROUGE-l(recall)} \cdot \text{ROUGE-l(precision)}}{\beta^2 \cdot \text{ROUGE-l(precision)} + \text{ROUGE-l(recall)}} \tag{2.11}$$

The formulas for this metric use the variable r_i to represent a sentence from a reference summary that is part of a larger set of summary sentences with a size of n sentences and m words. The variable C is used to represent the candidate summary.

ROUGE has been widely used in summarization tasks and has shown good correlation with human judgment (Radev et al., 2013). However, it has limitations, such as the fact that it only measures lexical overlap and does not consider the quality of the generated summary in terms of coherence, fluency, and other linguistic aspects.

2.11.3 BERTScore

BERTscore is a newer evaluation metric that aims to overcome some of the limitations of ROUGE by taking into account semantic similarity between the candidate summary and the reference summaries. BERTscore computes the cosine similarity between contextual embeddings of overlapping n-grams in the candidate summary and each reference summary, and then takes the harmonic mean of precision and recall of the overlapping n-grams. The contextual embeddings are obtained by passing the sentences through a pre-trained BERT model, which has shown to outperform previous state-of-the-art models in many natural language processing tasks including summarization. BERTscore has been found to correlate better with human judgments of summary quality than ROUGE, especially for

longer summaries and those that contain paraphrases or rephrased content.

The BERTscore formula for computing the F1 score is:

$$F_\beta = \frac{(1+\beta^2)\cdot \text{precision}\cdot \text{recall}}{(\beta^2\cdot \text{precision})+\text{recall}} \tag{2.12}$$

where β is set to 1 and the precision and recall are calculated as:

$$\text{precision} = \frac{1}{n}\sum_{i=1}^{n}\max_{j\in M_i}\text{sim}(w_i, w_j) \tag{2.13}$$

$$\text{recall} = \frac{1}{m}\sum_{j=1}^{m}\max_{i\in M_j}\text{sim}(w_i, w_j) \tag{2.14}$$

Here, n and m are the numbers of tokens in the candidate and reference summaries, respectively, and M_i and M_j are the sets of indices of the reference and candidate tokens that share the same n-gram as the i-th and j-th tokens, respectively. $\text{sim}(w_i, w_j)$ is the cosine similarity between the contextual embeddings of tokens i and j, which are obtained by passing the corresponding sentences through a pre-trained BERT model.

2.12 Summary

This chapter furnishes foundational understanding of the topics discussed in this research. It assesses the methodologies employed in natural language processing, primarily neural networks, and proceeds to elucidate the essentials of deep learning. Additionally, it presents an introduction to the core principles of prevalent deep learning structures in NLP and evaluation metrics.

Chapter 3

Related Work

3.1 Faithfulness in NLP

Natural Language Generation (NLG) has been advancing rapidly due to the development of deep learning techniques like pre-trained language models. This progress has resulted in better quality generated text that is more fluent, coherent, and can even be controlled for specific properties such as style, sentiment, and length. As a result, there has been significant progress in tasks such as summarization, dialogue generation, machine translation, and data-to-text generation. However, a major challenge with NLG is that the generated text can often contain inaccuracies or errors, making it unsuitable for many practical applications.

To address this problem, many researchers have proposed various methods for analyzing, evaluating, and optimizing the faithfulness of NLG models across different tasks. In this chapter we provide a comprehensive overview of the research progress on the faithfulness problem of NLG, including analysis, evaluation metrics, and optimization methods.

3.1.1 NLG Overview

NLG tasks can be categorized into three main types: text-to-text generation, data-to-text generation, and multimodality-to-text generation. Text-to-text generation tasks involve using existing texts as input to automatically produce new, coherent text as output. This includes applications such as dialogue generation, machine translation, question generation, and summarization (Allahyari et al., 2017). Data-to-text generation tasks generate

text automatically from numerical or structured data such as tables, key-value lists, and tuples. Examples of data-to-text generation applications include table-to-text generation, knowledge-graph-to-text generation, and meaning-to-text generation (e.g. AMR-to-text) (Song et al., 2018). Finally, multimodality-to-text generation tasks transfer the semantics in multimodal input, such as images or videos, into natural language texts. Image captioning, visual storytelling, and video summarization are some examples of multimodality-to-text generation applications (Huang et al., 2016).

NLG tasks can also be divided into open-ended and non-open-ended language generation based on input-output information transformation. Open-ended language generation tasks involve incomplete input, and the output semantics are not contained in the input. An example of an open-ended language generation task is story generation, where the model needs to create new information to complete the storyline planning and generate meaningful stories. In non-open-ended language generation tasks, the input usually provides complete or more information for the output. Machine translation is an example of a non-open-ended language generation task where the input provides complete semantics for the output. Paraphrase generation, where the input and output semantics are identical, but the language expression is different, is also considered a non-open-ended language generation task. Text summarization is another example of a non-open-ended language generation task, where the input usually provides more information than output, and the model needs to select the most important information to produce a summary output.

3.1.2 Development of NLG

The field of natural language generation (NLG) has a rich history dating back to the 1950s. Over time, NLG approaches have undergone four major stages of development, including template-based, statistical-based, neural-based, and pre-training-based methods.

Template-based: The earliest NLG systems utilized template-based approaches that involved designing modules for text generation using linguistic knowledge, including vo-

cabulary, grammar, syntax, and pragmatics. These systems typically included content planning, sentence planning, and text realization components to perform specific functions.

Statistical-based: Statistical-based NLG systems proposed a new approach to language modeling based on probability and statistics. The most popular statistical model, the N-gram language model, was often combined with template-based methods to improve the quality of generated text.

Neural-based: The advent of deep learning led to the development of neural-based NLG methods, which have become the dominant approach. These methods use various neural architectures, such as recurrent neural networks (RNNs), convolutional neural networks (CNNs), and self-attention transformer networks to model the relationship between vocabulary and context by end-to-end training.

Pretraining-based: The most recent stage of development in NLG is pre-training-based methods, which are based on the Transformer architecture and can capture the linguistic knowledge of vocabulary, syntax, and semantics. These models have been widely used for text generation in various applications, including text summarization, dialogue generation, data-to-text generation, and machine translation. However, while pre-training-based models generate fluent and grammatically correct text, they may still produce factual errors that contradict the input text.

The development of natural language generation (NLG) has faced four main issues throughout its various stages of development: fluency, informativeness, controllability, and faithfulness. The fluency challenge relates to whether the generated text is fluent, grammatical, and coherent. In contrast, the informativeness challenge refers to the model generating redundant, meaningless, and general content that lacks informativeness, diversity, and specificity. The controllability challenge refers to the generated text not satisfying the pre-given constraints, such as text style, attributes, and content. Finally, the faithfulness challenge relates to the generated content being inconsistent with the input information, containing hallucinations or non-factual information.

Although traditional template-based methods are reliable and can generate faithful texts, they are limited by the diversity and generality of the rules they rely on, which can result in issues with fluency and informativeness. Neural-based methods have the advantage of generating fluent and informative texts due to end-to-end training on large corpora, but they face challenges with controllability and faithfulness due to the introduction of the probability sampling mechanism. This challenge arises because the vocabulary size is large, usually in the order of 1000 to 50000, and the probability distribution inevitably contains a large number of long-tail words with low probability of occurrence. Additionally, the randomness of probability sampling can further reduce the controllability and faithfulness of the neural-based NLG model.

Recently, pre-training-based methods have emerged, which generate outstanding text in terms of fluency, informativeness, and even controllability. However, they still face challenges with faithfulness and may generate text that is inconsistent with the input information.

3.1.3 Faithfulness issues

The biggest challenge in NLG is the faithfulness problem, which limits the applicability of NLG algorithms in practical scenarios. State-of-the-art models have been shown to have faithfulness issues, which can be detrimental to the credibility and usability of systems, especially in tasks such as text summarization (e.g., up to 30% of generated summaries have been found to be unfaithful in previous studies (Cao et al., 2017; Falke et al., 2019; Kry'sci'nski et al., 2019; Pagnoni et al., 2021)), dialogue generation, machine translation, and table-to-text generation. Table 3.1 displays an instance of a text summarization that lacks faithfulness. The errors are represented by the color red. The faithfulness problem is present in almost all NLG tasks and can be defined differently depending on the task.

For non-open NLG tasks, such as text summarization and machine translation, the faithfulness problem relates to whether the generated content is factually consistent with the

Table 3.1: An example of errors that are not faithful.

Task	Source	Output
Abstractive Text Summarization	The first vaccine for Ebola was approved by the FDA in 2019 in the US, five years after the initial outbreak in 2014. To produce the vaccine, scientists had to sequence the DNA of Ebola, then identify possible vaccines, and finally show successful clinical trials. Scientists say a vaccine for COVID-19 is unlikely to be ready this year, although clinical trials have already started.	The first vaccine for Ebola was rejected in 2019. Scientists say a vaccine for Ebola is unlikely to be ready this year.

input information, which is referred to as factual consistency. On the other hand, for open-ended NLG tasks, such as news article generation, the model leverages knowledge from knowledge graphs or corpora to create new content, and the faithfulness problem refers to whether the generated content is factually consistent with world knowledge or commonsense, which is referred to as factual correctness. The research on faithfulness mainly focuses on non-open-ended tasks, and much effort has been devoted to optimizing faithfulness using automatic faithfulness evaluation metrics and meta-evaluations for these metrics.

If the output sequence of natural language generation (NLG) includes a range of tokens $(y_i, ..., y_j)$ that is not supported by the input sequence x, then it is defined as being unfaithful. There are two categories of faithfulness issues, which relate to factual consistency with the source sequence:

Intrinsic Error: When a fact is synthesized using information present in x, and it contradicts the source sequence (Maynez et al., 2020a).

Extrinsic Error: When a fact is neither supported nor contradicted by the source sequence (Maynez et al., 2020a).

3.1.4 Fine-grained categories of factual errors

The different types of errors that can occur during natural language generation (NLG) can be classified into three categories based on the type of error (Pagnoni et al., 2021): semantic frame errors, discourse errors, and content verifiable errors.

Semantic Frame Errors: These errors pertain to errors in the frame semantic and its core and non-core frame elements. This includes Predicate Errors (PredE) where the predicate is inconsistent with the source text, Entity Errors (EntE) where the primary arguments (such as entities) of the predicate are incorrect or have the wrong attributes, and Circumstance Errors (CircE) where the arguments and predicates interact in the wrong way (Palmer et al., 2005).

Discourse Errors: They capture errors in the linking of discourse segments, such as Co-reference Errors (CorefE) where pronouns or other references to previously mentioned entities are incorrect or have no clear antecedents, and Discourse Link Errors (LinkE) where there are incorrect links between different statements.

Content Verifiable Errors: The errors capture incorrect information that cannot be verified against the source text, including Out of Article Errors (OutE) where the information cannot be deduced from the original text (Maynez et al., 2020a), and Grammatical Errors (GramE) where the statement is not well-formed, making its meaning ambiguous or incomprehensible and cannot be verified against the source.

While all extrinsic errors are considered incorrect, recent studies (Maynez et al., 2020a; Cao et al., 2021) have shown that factual hallucinations (i.e. errors that are consistent with world knowledge but not inferable from the source text) can be beneficial in a summary by providing useful background information. Therefore, OutE errors can be further categorized into factual hallucinations and non-factual hallucinations. This hierarchical typology of faithfulness problems provides a more thorough means of categorizing the types of errors made by generation models, enabling deeper insights than simply categorizing content as faithful or unfaithful. By integrating these definitions, Table 3.2 illustrates the classification

of faithfulness issues.

Table 3.2: Error classification of text summarization using input from Table 3.1

Error Types			Examples
Intrinsic Error	Semantic Frame Errors	Predicate Error (PredE)	The Ebola vaccine was rejected by the FDA in 2019.
		Entity Error (EntE)	Scientists say a vaccine for Ebola is unlikely to be ready this year.
		Circumstance Error (CircE)	The first vaccine for Ebola was approved by the FDA in 2014.
	Discourse Errors	Co-reference Error (CorefE)	The first vaccine for Ebola was approved in 2019. They say a vaccine for COVID-19 is unlikely to be ready this year.
		Discourse Link Error (LinkE)	To produce the vaccine, scientists have to show successful human trials, then sequence the DNA of the virus.
Extrinsic Error	Factual		China has already started clinical trials of the COVID-19 vaccine.
	Non-Factual		China didn't start clinical trials of the COVID-19 vaccine.

3.1.5 Analysis

Models: Various studies (Pagnoni et al., 2021; Cao and Wang, 2021b) have conducted annotations with different levels of detail to examine the faithfulness performance of existing language generation models. The outcomes demonstrate that even the most advanced

pre-trained models still face significant issues with generating unfaithful content. Specifically, the results indicate that on the XSum dataset, all systems generate over 60% unfaithful summaries, while on the CNN/DM dataset, T5 and BART produce over 20% unfaithful summaries. These findings reveal both the gravity of the faithfulness problem of current models, and the substantial influence of different datasets.

Annotation: Annotation of NLG model faithfulness is challenging. Existing work typically categorizes the generated text as either faithful or unfaithful, although this binary approach has shown low inter-annotator agreement (Falke et al., 2019). Other researchers have attempted more granular annotation, such as Pagnoni et al. (2021), who collected human annotations from three independent annotators. However, their approach also resulted in low inter-annotator agreement, with Fleiss (1971) scores of 0.58 for faithful or not, and 0.39 for specific unfaithful error types. Tang et al. (2021) have compared different types of human annotations for faithfulness, and found that ranking-based Best-Worst Scaling annotations have higher reliability than rating-based annotations.

Causes: The accuracy of model-generated outcomes can be influenced by several variables, including the dataset used, the training methodology employed, and the expressiveness of the model.

- The data in the source and reference are not consistent – One of the primary causes of extrinsic hallucinations during generation is the data divergence between source and reference. For instance, in text summarization, summaries are usually written by journalists to introduce news articles. Hence, these summaries often contain additional information not present in the original document. This issue of divergence between source and target is prevalent in conditional text generation, as reported in previous studies (Dhingra et al., 2019; Kry'sci'nski et al., 2019; Wiseman et al., 2017), and may occur due to heuristic data collection or the nature of certain NLG tasks like table-to-text and dialogue generation. Despite the existence of this issue, existing models do not account for source-reference divergence, which makes them

susceptible to hallucinations. As a result, these models may produce texts that do not align with the input, yet yield high model log-likelihood. This explains why the same model can perform differently on different datasets, such as the difference in summarization quality between the XSum and CNN/DM datasets. The XSum dataset is collected heuristically by taking the introductory sentence of each article as its reference summary, leading to reference summaries that often contain hallucinations. In a study by Maynez et al. (2020), 76.9% of reference summaries were reported to contain unfaithful content. In contrast, the reference summaries of the CNN/DM datasets are all human-written and contain fewer hallucinations. Thus, the summarization model's faithfulness on the CNN/DM dataset is much better than that on the XSum dataset.

- The disparity in exposure between the training phase and the inference phase – According to Wang and Sennrich (2020), the discrepancy between training and inference, known as exposure bias (Ranzato et al., 2015), can cause hallucinations. The teacher-forcing training algorithm (Williams and Zipser, 1989), which is commonly used, can lead to this discrepancy, resulting in inaccurate outputs with factual hallucinations (Maynez et al., 2020a). Additionally, the model is optimized only for maximizing the log-likelihood of the reference summary at the word-level, which does not necessarily incentivize models to be faithful.

- Inadequate representation of the text – If a model has a poor input text representation, it can struggle with document level inference, which is often needed for tasks like abstraction and generation, and it can be prone to errors. For instance, in text summarization, the percentage of system-generated summaries with intrinsic hallucination was higher compared to gold summaries. This trend highlights the models' tendency to misrepresent information in the document due to the lack of document-level understanding and inference. To enhance text representation, using large pre-trained models is a common practice for downstream natural language generation

tasks. Pre-training can improve text generation by providing the model exposure to vast amounts of text, allowing it to integrate background knowledge with generation. However, according to Longpre et al. (2021), these models may rely too heavily on the parametric knowledge learned from large scale corpus over the provided input. Additionally, the dominant language model often prompts the decoder to generate common words to ensure fluent output.

3.2 Evaluation Metrics

Recent research has focused on evaluating the factual consistency of generated text, leading to the proposal of various new metrics for different natural language generation tasks. These metrics can be classified into three types: Entailment Score, QA Score, and Fact Score.

3.2.1 Entailment Score

To evaluate the faithfulness of generated texts, one of the popular approaches is to use NLI (Natural Language Inference) methods. These methods assess whether the generated text is entailed, neutral, or conflicting with the input text. The idea behind this approach is that the generated text should be entailed or at least not in conflict with the source text. Although an NLI model can predict three scores for entailment, neutral, and contradiction, most studies only use the entailment score to evaluate faithfulness. An NLI model $\mathcal{N}$ predicts the entailment score ,$\mathcal{N}(x,y)$, given a source text x as a premise and a generated text y as a hypothesis. The faithfulness of y given x is higher when $\mathcal{N}(x,y)$ is larger.

Evaluating the proposed metrics often involves reporting their correlations with human judgements. While traditional NLI tasks involve predicting entailment scores between individual sentences, applying this approach to text generation is complicated by the fact that the input text x can take on various forms and often comprises multiple sentences. Some entailment-based metrics propose ranking-based downstream tasks to demonstrate perfor-

mance (Falke et al., 2019). This method proposed is to aggregate entailment scores between the sentences of x and y to calculate a faithfulness score between them.

$$\frac{1}{|y|}\sum_{s_y \in y}\max_{s_x \in x}\mathcal{N}(s_x, s_y) \tag{3.1}$$

3.2.2 Question-Answering Score

Given that evaluating faithfulness requires logical inference based on factual information, it is natural to leverage the reasoning abilities of Question Answering (QA) models. Several recent studies such as QAGS (Wang et al., 2020) and FEQA (Durmus et al., 2020b) have proposed QA-based metrics for evaluating factual accuracy. As illustrated in Figure 3.1, these metrics typically consist of a Question Generation (QG) module and a QA module. The basic idea behind these metrics is to predict a matching score between the source answers (key information units from the source text) and the target answers (key information units from the generated text). The overall procedure for these metrics can be summarized as follows:

1. **Answer Selection:** Identify information units from the generated text that are considered target answers.

2. **Question Generation:** Using the generated text as context and the selected target answers as a condition, the QG module generates questions.

3. **Question Answering:** The QA module answers the questions using the source text as context to retrieve source answers.

4. **Answer Alignment Evaluation:** An answer alignment metric is used to calculate the matching score between source and target answers, which is then output as the final evaluation score.

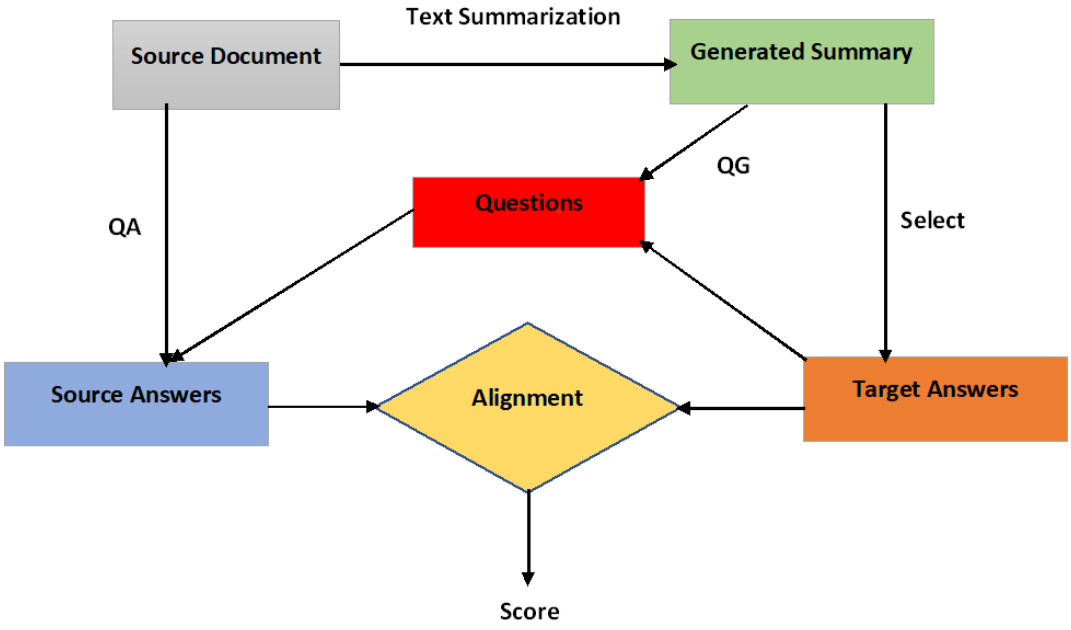

Figure 3.1: QA Metric Architecture

3.2.3 Fact Score

In the context of assessing the faithfulness of a generated text, factual inconsistencies can manifest either at the level of entities (Nan et al., 2021b), or at the level of relations (Sabet et al., 2020). Inconsistencies at the entity level occur when the generated text includes named entities that are not present in the source document. Inconsistencies at the relation level, on the other hand, arise when the entities are present in the source document but the relations between them are not reflected accurately in the generated text.

3.3 Improvement Approaches

Various optimization methods have been proposed to address the issue of faithfulness in a range of natural language processing tasks, such as abstractive summarization, dialogue generation, data-to-text generation, and machine translation. Although these methods are applicable across different tasks, they can be classified into several categories, including guidance, auxiliary, learning-based, post-editing, constrained decoding, and other techniques.

3.3.1 Guidance

Guidance is a straightforward and effective approach for improving the faithfulness and informativeness of summarization tasks. This method involves providing additional signals or inputs to the model that supplement the source document. The key considerations for implementing factual guidance are determining the type of information to feed into the model and the approach for doing so. One example of a factual guidance framework is shown in Figure 3.2, which is based on a simple sequence-to-sequence model. In this framework, two encoders process the original source and the extra guidance signals, and a decoder generates the final summary based on the hidden states of both encoders. Guidance signals may take various forms, such as keywords, important sentences, or other structures like relations or semantic graphs. Depending on the types of guidance signals, encoders may be based on a Transformer network (for signal of sequence structure) or a Graph Attention Network (for signal of graph structure) such as the one proposed by Veličković et al. (2017). GSum (Dou et al., 2021) is a general and extensible framework that can incorporate different types of external guidance signals to address unfaithful summaries. The guidance signals in GSum can be classified into three types: keywords, sentences, and relations.

Keywords: Keywords provide a concise representation of the crucial information contained in the source text, enabling summarization models to focus on the most important aspects and produce fewer factual errors. To leverage the benefits of keywords, Li et al. (2018a) proposed a Key Information Guide Network that encodes keywords into key information representation to guide the summarization process. Specifically, they first extract keywords using the TextRank algorithm and then encode them using a Bi-RNN network. During generation, the keyword representations are utilized in both the attention mechanism and pointer mechanism. Saito et al. (2020) further advanced this approach by introducing token-level saliency models called CIT, which combine a saliency model (composed of a Transformer encoder and a feed-forward layer) with a pre-trained sequence-to-sequence model. The saliency model produces scores for each token, and important tokens are iden-

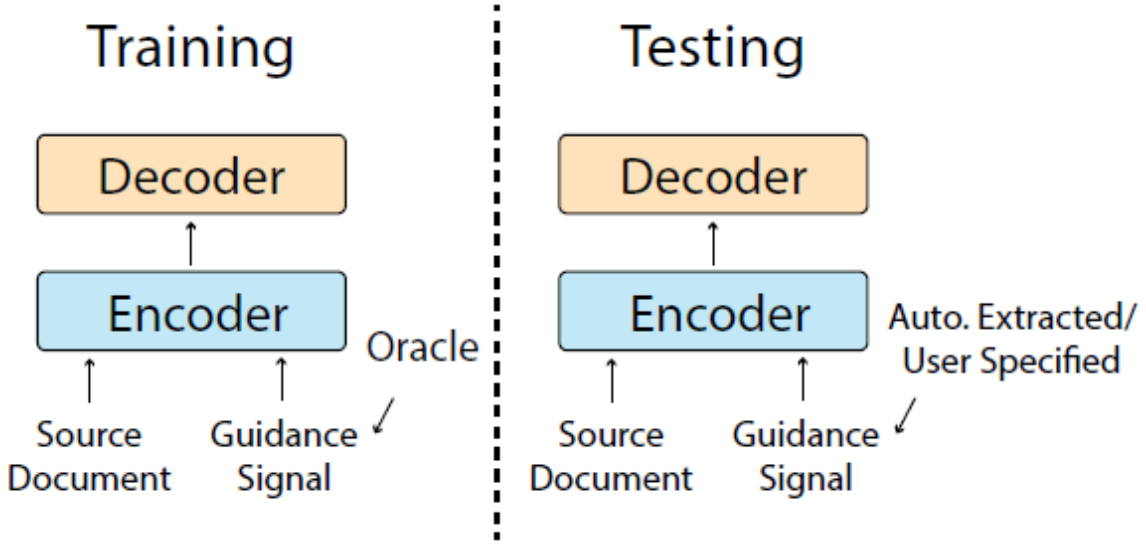

Figure 3.2: In factual guidance method, guidance selection typically involves using an oracle method during training and relying on automatically extracted guidance at test time. (Dou et al., 2021)

tified as K. The saliency model output and the source text are concatenated to create a new input $\hat{X} = concat(K, X)$ for the sequence-to-sequence model.

Sentence: To avoid repetition, sentence-level guidance can be used instead of relying solely on keywords. Re3Sum, proposed by Cao et al. (2018), is a method that retrieves existing summary sentences as candidate templates and uses an extended seq2seq framework to generate a summary that is template-aware. This framework converts both the source text X and the soft template R into hidden states with an RNN encoder. In the Rerank module, the saliency of R is measured based on its hidden state relevance to X.

In the Rewrite module, an RNN decoder combines the hidden states of X and R to generate a summary Y. Similarly, Song et al. (2020) used an extractor to select attractive sentences from the article, followed by a seq2seq-based abstractor to rewrite these sentences. The PORL-HG model combines the extractor with the abstractor using a reinforcement learning network that uses popularity and ROUGE scores as rewards to ensure that the generated headlines are both attractive and faithful.

Relation: According to Dou et al. (2021), the use of full sentences as guidance sig-

nals may contain irrelevant information, which could distract the model from the most important parts of the source text. To address this issue, some researchers have turned to using relation information in the form of relational triples as factual guidance. Cao et al. (2017) extracted fact descriptions from the source text using open information extraction and dependency parsing techniques. They proposed a dual-attention seq2seq framework that conditions the generation on both the source text and the extracted fact descriptions. Huang et al. (2020), introduced the ASGARD framework, which enhances the document encoder with a graph-structured encoder that maintains the global context and local characteristics of entities. They utilized <subject, predicate, object> triples extracted by OpenIE to construct a knowledge graph, which is used as extra factual guidance during summary generation. Most works extract relations from source documents using OpenIE and represent them as graph structures to improve seq2seq models. However, these OpenIE-based graphs only contain sparse relations between partial words and may not cover the overall semantic meaning of the source article. Wu et al. (2021) proposed BASS, which introduces a unified semantic graph to enhance the performance of multi-document summarization. The model extracts phrases and their relations from sentences using a two-stage merging process, and encodes graph structures in both the encoding and decoding processes, using the graph adjacent matrix as a self-attention mask and a graph-propagate attention mechanism to guide the decoding process.

3.3.2 Auxiliary

Instead of explicitly improving factual consistency through guidance methods, auxiliary task-based methods use correlated tasks to enhance the performance of summarization systems in an implicit way. These methods can be implemented using reinforcement learning, multi-task learning, and re-ranking frameworks, as illustrated in Figure 3.3. In the reinforcement learning framework, a score model is designed to optimize the factual consistency of the generated summaries by obtaining a reward. In the multi-task framework,

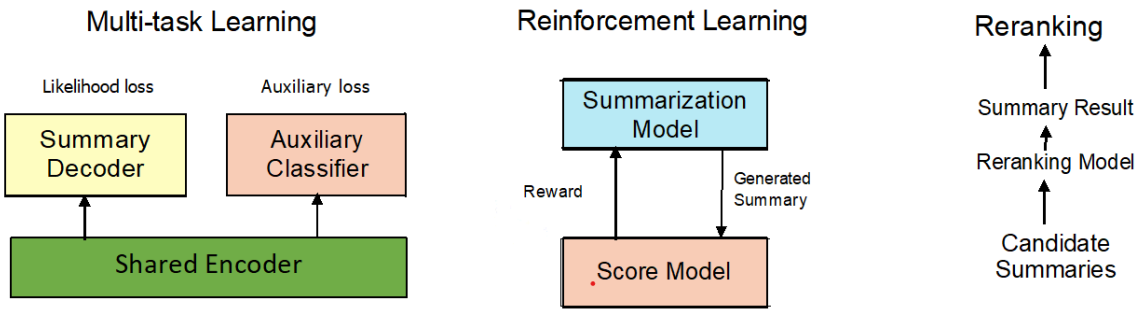

Figure 3.3: Auxiliary methods can be implemented through three different frameworks, namely multi-task learning, reinforcement learning, and re-ranking.

a task-specific layer is added to the shared-weight encoder, allowing the summarization model and auxiliary model to share the same semantic representations but have different learning objectives. The related auxiliary task complements the summarization task and improves the original summarization system's performance. The re-ranking framework generates several candidate summaries, scores them based on auxiliary tasks, and selects the best summary. Various common auxiliary tasks are used to enhance the faithfulness of abstractive summarization and will be described further.

Entailment Objective: NLI, or Natural Language Inference, is an entailment task in which a hypothesis sentence is classified as entailed, neutral, or contradicting a premise sentence. Previous studies (Barrantes et al., 2020) have demonstrated that incorporating NLI tasks into summarization models can improve their faithfulness. This can be achieved through multi-task learning, RL reward, or re-ranking summary candidates. Li et al. (2018) were the first to integrate entailment knowledge into abstractive summarization. They proposed an entailment-aware encoder in a multi-task learning framework and an entailment-aware decoder in an RL framework with entailment rewards. They used shared weight encoders trained on both the summarization and entailment tasks, where entailment prediction was considered as an auxiliary task for summary generation. Falke et al. (2019) proposed a re-ranking approach based on the idea that all information in a summary should be entailed by the source document, and Barrantes et al. (2020) applied an adversarial NLI dataset to further train the NLI model. Fabbri et al. (2021a) proposed a query-based sum-

marization model that employs NLI scores as reinforcement learning rewards to improve factual consistency.

Question Answering Objective: In order to generate summaries that are factually consistent, it is not only necessary to have a comprehensive understanding of the source text, but also to distinguish between important and irrelevant information. Therefore, it is natural to use a question answering (QA) model to evaluate a summarization model's comprehension and discrimination abilities. QA-based methods typically calculate a QA score by comparing answers extracted from the source text and the generated summaries, and then use the QA score as a reward in either the RL framework or the re-ranking framework. Building on this approach, Nan et al. (2021a) integrated a QA model into the seq2seq architecture using a novel contrastive learning method. They generated candidate summaries, sorted them into positive and negative samples based on the QA score, and then used contrastive learning to improve the models' faithfulness.

3.3.3 Learning

Cao and Wang (2021a) discovered that the maximum likelihood training method, which is frequently used, has limited ability in distinguishing between accurate and inaccurate generations. To address this issue, a possible solution is to devise novel learning objectives that enhance the preference for factual summaries over inconsistent ones. To this end, Contrastive Learning (CL) has emerged as a promising paradigm that was first proposed for visual tasks and is now widely used in numerous NLP tasks. The central premise of CL is to train the model to produce representations of similar samples that are close to each other, while keeping the dissimilar ones apart. The critical aspect of CL is the generation of positive and negative samples. In visual tasks, positive samples are commonly generated by rotating, resizing, distorting the original image, while other images serve as negative samples. In this section, we will introduce an effective technique for generating positive and negative samples in summarization tasks and how to integrate them into the CL framework.

Cao and Wang (2021a) introduced a task-specific formulation of contrastive learning (CLIFF) to train a summarizer to distinguish between factual and non-factual summaries. CLIFF employs three techniques for creating positive samples, including synonym substitution, word replacement, and back-translation. In contrast, previous works often used other samples in the same batch as negative samples, but Cao and Wang (2021a) argued that this method is not effective since it is easy to distinguish between positive and negative samples. To address this issue, CLIFF uses four strategies for generating negative samples. The first approach, called entity swap, mimics intrinsic errors by swapping named entities in the references with other randomly selected entities of the same type in the source text. The second technique is called mask-and-fill with BART, which replaces each named entity in a reference with a [MASK] and then generates new entities using BART. The third approach, called source-conditioned regeneration, involves feeding the text before each entity in the reference along with the original source into BART to generate negative samples. The final approach, called system generation, selects summaries generated by the system with low probability as negative samples.

Once the positive samples (P) and negative samples (N) have been created, CLIFF utilizes the formula 3.2 to optimize the contrastive learning objective. This objective is then combined with a conventional cross-entropy loss (L_{CE}) to create the final training objective, as shown in the formula 3.3. The representations h_i, h_j, and h_k correspond to the summaries y_i, y_j, and y_k. Additionally, the function sim calculates the cosine similarity between the summary representations. τ, which is a temperature parameter, is assigned to a value of 1.0 and λ is a scalar value that is set to 1.0.

$$LCL = \frac{-1}{\binom{|P|}{2}} \sum_{y_i, y_j \in P; y_i \neq y_j} \log \frac{\exp(\text{sim}(h_i, h_j)/\tau)}{\sum_{y_k \in P \cup N; y_k \neq y_i} \exp(\text{sim}(h_i, h_k)/\tau)} \tag{3.2}$$

$$L = L_{CE} + \lambda L_{CL} \tag{3.3}$$

3.3.4 Post-Editing

The previous methods described require model structure modifications or additional sample construction processes to improve factual consistency, which could potentially impact the informativeness of summary results as measured by metrics such as ROUGE scores. Alternatively, post-editing based methods aim to enhance factual consistency by incorporating a corrector into the system-generated summaries. These methods treat the generated summaries as drafts, which are then reviewed and edited to correct any factual errors and form the final summaries. This process closely mirrors the human writing process, where individuals typically write a first draft and then revise and edit it to improve its quality.

To address the challenge of factual inconsistencies, Dong et al. (2020b) proposed a suite of factual correction models called SpanFact that leverage knowledge learned from question answering models. SpanFact involves correcting system-generated summaries through span selection and correction, taking into account entity-level corrections in an iterative manner. Specifically, SpanFact masks each entity in the system summary at each time step and queries a QA model to replace the incorrect entities with the correct ones based on the source document. The corrected entity is then integrated into the updated summary for use in the next iteration. According to human evaluation, SpanFact can correct approximately 26% of unfaithful summaries while preserving the accuracy of otherwise correct summaries.

In contrast, Cao et al. (2020b) simplified the post-editing procedures by directly training a seq2seq rewrite model on artificial unfaithful summaries as a corrector. They constructed a weakly-supervised training dataset based on text transformations that replace entities, numbers, numerals, and pronouns in the source documents with other tokens of the same type. The corrector's goal is to generate correct summaries based on the unfaithful summaries and source documents.

Although post-editing methods have shown effectiveness in improving the faithfulness

of abstractive summarization systems while maintaining their informativeness, they represent a more indirect solution to factual inconsistencies rather than a fundamental one.

3.3.5 Constrained Decoding

Guided decoding, a variation of beam search, ensures specific words and phrases are included in the output, providing control over generated tokens without altering the model structure or requiring additional training data. CAS (Constrained Abstractive Summarization) introduced by Mao et al. (2020) improves factual consistency of summarization by constraining token sets during dynamic beam search decoding, only ending the generation process when all constraints are met. Constrained sets focus on entities and noun phrases not present in unconstrained summaries, resulting in more accurate and faithful tokens generated during inference. Meanwhile, Aralikatte et al. (2021) proposed FAME, a Focus Attention Mechanism that combines a contextual representation with a dynamic source-conditioned lexical bias layer, encouraging the decoder to produce input-document-faithful tokens.

3.4 Elementary Discourse Units

Elementary Discourse Units (EDUs) are fundamental building blocks in discourse analysis that represent independent units of meaning in a text. EDUs are defined as the smallest segment of text that convey a complete thought or idea. The concept of EDUs has been widely used in Natural Language Processing (NLP) and text summarization to improve the accuracy and coherence of generated summaries.

EDUs were first introduced by Mann and Thompson (1988), who defined them as "the smallest unit of discourse structure with an independent pragmatic function." Since then, EDUs have been used in several studies to capture the underlying structure of text and identify discourse relations between sentences.

One of the key advantages of using EDUs in NLP and text summarization is their abil-

ity to capture the meaning of a text beyond the sentence level. By grouping sentences into EDUs, it becomes possible to identify discourse relations between them, such as elaboration, contrast, and causality. These relations can then be used to generate summaries that capture the overall meaning and structure of the input text.

Here are a few examples of Elementary Discourse Units (EDUs):

- ”John went to the store.” - This is a simple EDU that conveys a complete thought or idea.
- ”Although it was raining, the game continued.” - This is an example of a complex EDU that contains a subordinate clause (”although it was raining”).
- ”The cat sat on the mat. The dog barked.” - These are two separate EDUs that are related to each other through their discourse relation, which could be elaboration (i.e., the dog barked in response to the cat sitting on the mat).
- ”The new restaurant opened last week. It has already received several positive reviews.” - These two sentences could be combined into a single EDU that conveys the idea that the new restaurant has been successful since its opening.
- ”The company announced record profits. However, its stock price fell.” - These two sentences are related through a discourse relation of contrast, indicating that although the company had record profits, its stock price fell.

These examples demonstrate how EDUs can be used to break down text into smaller units that convey a complete thought or idea. By identifying these units, it becomes possible to analyze the discourse structure of input text and generate summaries that capture the overall meaning and structure of the text.

There are some problems related to pre-trained text summarization models working with sentences, such as BERT. They usually generate extra information and are nearly extractive. Additionally, since these language models work with sentences rather than documents, they

usually miss the long relationship between tokens during the whole document. Smaller units such as EDUs or discourse units recently used in the summarization techniques to come up with these setbacks. Several studies have shown the importance of using EDUs in text summarization.

For example, in a study by Ji and Eisenstein (2014), the authors used EDUs to model the discourse structure of input text and generate summaries. They divided the input text into EDUs and used a graph-based approach to identify the most important EDUs. The identified EDUs were then used to generate summaries that were more coherent and readable than those generated without EDUs.

Another study by Filippova et al. (2015) used EDUs to identify sentence pairs that expressed similar or contrasting information. They first divided the input text into EDUs and then used a clustering algorithm to group the EDUs into similar or contrasting clusters. The identified clusters were then used to generate summaries that were more faithful to the input text.

Xu et al. (2020) (DISCOBERT (Discourse-Aware Neural Extractive Text Summarization)) proposed a novel approach to automatic text summarization that incorporates the discourse structure of the input text to improve the quality of the generated summaries. The authors used a neural network architecture and there is an encoder on top of BERT to encode the RST dependency graph and coreference. It captures the dependency in both level of paragraph and document. It is trained to predict the relevance of each sentence in the input text to the overall meaning of the document. The proposed model consists of several components, including a discourse-aware sentence encoder and a hierarchical attention mechanism that combines local and global context information to generate the final summary. To capture the discourse structure of the input text, the authors use a discourse parser to identify discourse relations between sentences. These relations include elaboration, contrast, and causality, among others. The sentences are then encoded with a discourse-aware encoder that takes into account the local and global context of the sentence, as well as its dis-

course relation with neighboring sentences. The hierarchical attention mechanism is used to generate a summary from the encoded sentences. It consists of two levels of attention: a local attention mechanism that focuses on the most relevant words within each sentence, and a global attention mechanism that combines the information from all the sentences in the document to generate the final summary. The authors evaluate their proposed model on several benchmark datasets and compare it to several state-of-the-art models. The results demonstrate that the proposed approach outperforms previous models in improving the coherence and faithfulness of generated summaries on several metrics, including ROUGE scores and human evaluation.

EDUSUM (Li et al., 2020) presents a novel approach to abstractive summarization that takes into account a method that uses Elementary Discourse Units (EDUs) to represent the discourse structure of input text. The proposed method consists of several steps: EDU selection, EDU fusion, encoding, decoding, and copy mechanism. In the EDU selection step, the authors first use a discourse parser to identify the EDUs in the input text and their discourse relations. Then, they use a heuristic method based on the discourse relation graph to select the most important EDUs for inclusion in the summary. In the EDU fusion step, the authors merge adjacent EDUs that share the same discourse relation to form larger discourse segments. This step is intended to capture the coherence and structure of the input text at the discourse level. To enhance the EDU selection process, reinforcement learning is utilized with the reward based on the ROUGE score of the EDU fusion result. In the encoding step, the authors encode the selected EDUs or fused segments into a fixed-length vector representation using a bidirectional long short-term memory (BiLSTM) network. The encoding step generates a sequence of EDU embeddings that capture the meaning of each EDU or fused segment. In the decoding step, the authors use a decoder network that is based on a unidirectional LSTM to generate the summary using the encoded EDU embeddings as input. The decoder network uses an attention mechanism that weights the importance of each EDU or fused segment based on its relevance to the current summary generation step.

Finally, to handle out-of-vocabulary (OOV) words that are not represented in the EDUs, the authors use a copy mechanism that allows the decoder to copy words directly from the input text. The copy mechanism is applied based on the attention distribution over the input words. The results show that the proposed method achieves better results than previous models in terms of both informativeness and fluency, suggesting the effectiveness of using EDUs to represent the underlying discourse structure of input text, and the EDU selection and fusion steps to improve coherence and structure of the summaries.

Xiao et al. (2020) showed that by incorporating discourse information, the model is able to capture the underlying structure and coherence of the input text, allowing it to capture semantic relations and making human like summarization. Since this method uses discourse tree attention matrix in the transformer, it leads to having less parameter. Furthermore, discourse information can help the model to identify relationships and dependencies between sentences, allowing it to better understand the context of each sentence and the role it plays in the overall discourse. This can potentially reduce the complexity of the model by improving its ability to identify the most informative sentences for the summary without having to consider every possible sentence.

3.5 Summary

In this section, we present faithfulness problems in Text Summarization. We systematically organize and analyze the assessment of faithfulness, the metrics used to evaluate it, and the optimization techniques employed to address the issue. We also show the way EDUs could be integral in text summarization tasks through related studies.

Chapter 4

Improving faithfulness in BART with guided summarization using EDUs

4.1 Introduction

Recently, encoder-decoder models such as PEGASUS (Zhang et al., 2020a) and BART (Lewis et al., 2020) have achieved state-of-the-art results on various summarization datasets. Despite their success, these models still suffer from hallucination when used for abstractive summarization. Consequently, researchers such as Li et al. (2018a) and Dou et al. (2021) have explored the guidance mechanism (GSUM) as a means to enhance these summarization models. This mechanism involves using a guidance signal to determine which tokens should be generated. The guidance signal takes various forms, with structural information from the source document being a common type.

Our inspiration comes from a previous manual evaluation work by Hardy and Vlachos (2019), where they discovered that highlighted content can assist human evaluations and reduce variation in results. We conclude that, as EDUs are semantically simpler units than sentences, encoding them is more accurate and can improve the faithfulness of the generated summaries. In this chapter, we propose that the selection of important EDUs can improve the decision-making process of the decoder. Consequently, we propose new guidances including integral EDUs. Our primary objectives in this study are twofold:

1. To extract important EDUs that can enhance faithfulness of abstractive summarization.

2. To guide an abstractive summarization system using the extracted EDUs.

4.2 Methodology

Several studies have explored the use of EDUs to enhance abstractive summarization, similar to our own research (Li et al., 2020; Xiao et al., 2020) . We employed a joint-training approach for the guidance mechanism. We accomplish this by applying an EDU extractor model (DISCOBERT) (Xu et al., 2020), a classification model (GSUM) (Dou et al., 2021), that can extract important EDUs from the source document. The extracted key-phrases were then utilized as constraints for the summarizer model during inference.

Our approach involves enhancing the baseline summarizer model (BART) (Lewis et al., 2020) by utilizing important EDUs that are extracted from DISCOBERT (Xu et al., 2020) to guide its predictions. Then, we fine-tune the summarizer model, for which we utilized BART (Lewis et al., 2020), applying the extracted EDUs to guide and constrain the summarizer model's predictions.

BART: BART(Bidirectional and Auto-Regressive Transformer) (Lewis et al., 2020) is a powerful sequence-to-sequence model which is valuable for natural language processing tasks such as summarization and text generation. BART employs a cutting-edge in-filling technique and a process of shuffling sentences to optimize its performance. The model is trained on a denoising autoencoder task, where it is tasked with reconstructing the original uncorrupted sequence from a corrupted version of it. This pretraining method allows the model to learn rich representations of the input text, which can be fine-tuned for a variety of natural language processing tasks. The BART architecture consists of an encoder and a decoder, each composed of a stack of transformer layers. The encoder takes as input a sequence of tokens, and produces a sequence of hidden representations. The decoder takes as input a target sequence, and generates a sequence of tokens by attending to the encoder's hidden representations. The model is trained to minimize the reconstruction loss between the original uncorrupted sequence and the sequence generated by the decoder. One key

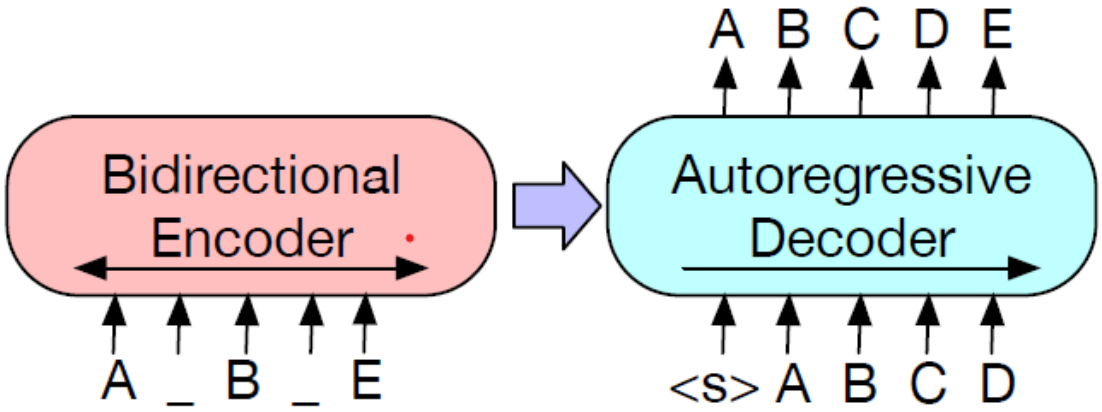

Figure 4.1: BART (Lewis et al., 2020)

difference between BART and other transformer-based models is that BART uses a bidirectional decoder. This means that during decoding, the model can attend to both the left and right context of the target token. This allows the model to generate more fluent and coherent sequences. BART also includes a set of additional features to further improve its performance. These include a learned mask token, which is used during pretraining to randomly mask tokens in the input sequence, and a sentence delimiter token, which is used during decoding to indicate the end of a sentence. Arbitrary noise transformations are permitted since encoder inputs don't have to match decoder outputs. As we can see in Figure 4.1, a document has been tampered with by substituting sections of text with mask symbols. The distorted document (on the left) is encoded using a bidirectional model, and the likelihood of the original document (on the right) is determined with an autoregressive decoder. During fine-tuning, both the encoder and decoder are fed an uncorrupted document, and the final hidden state of the decoder's representations are utilized. Additionally, BART includes a token-wise prediction mechanism that allows the model to make predictions at the sub-word level.

Alongside these justifications, it is chosen in this research as the baseline model, since it is a powerful language model that has been widely used for both generating faithful summaries (Akiyama et al., 2021; Chen et al., 2021; Nan et al., 2021a) and correcting

errors and hallucinations (Cao et al., 2020a) in generated summaries. However, like any other language model, BART is not perfect and can sometimes generate summaries that contain hallucinations or information that is not present in the input text. We address this issue in this research.

Guidance: In the process of abstractive summarization, a source document X which is composed of multiple sentences $x_1, x_2, ..., x_{|X|}$, is usually fed into an encoder that produces representations. The representations are then input to a decoder that generates the summary y one word at a time. During training, the model parameters θ are optimized to maximize the conditional likelihood of the outputs, which are obtained from a parallel training corpus $\langle \mathcal{X}, \mathcal{Y} \rangle$:

$$\arg\max_{\theta} \sum_{(x^i, y^i) \in \langle \mathcal{X}, \mathcal{Y} \rangle} \log p(y^i \mid x^i; \theta) \tag{4.1}$$

The act of providing additional input to a model alongside the source document X can be referred to as guidance. This supplementary signal, denoted as g, serves to support the model during training and inference. The information that is utilized to calculate g and the approach for integrating this information into the model can differ, but they are both part of the broader framework.

$$\arg\max_{\theta} \sum_{(x^i, y^i, g^i) \in \langle \mathcal{X}, \mathcal{Y}, \mathcal{G} \rangle} \log p(y^i \mid x^i, g^i; \theta) \tag{4.2}$$

During testing, the guidance signal can be defined in one of two ways: either manually by an interested user, or automatically predicted by an automated system based on input X. In the training phase, obtaining manual guidance can be too costly. Therefore, it is better to concentrate on two methods for generating guidance signals: automatic prediction using x and oracle extraction which involves deducing a value g from both x and y that is most likely to be beneficial in generating y. In theory, automatic prediction has the benefit of aligning with the training and testing conditions of a system that will also receive automatic

predictions during testing. However, using oracle guidance provides a significant advantage by producing highly informative guidance signals that prompt the model to give them more attention during testing.

Important EDUs Extraction: DISCOBERT (Discourse-Aware Neural Extractive Text Summarization) (Xu et al., 2020) is a pre-trained language model based on the popular BERT architecture (Devlin et al., 2019). However, DISCOBERT was specifically designed to capture the nuances of discourse structure and is intended to be used for discourse-related natural language processing tasks.

In recent times, state-of-the-art text summarization models have adopted BERT (Devlin et al., 2019) for document encoding. Nonetheless, extractive models based on sentences often lead to the inclusion of phrases that are redundant or uninformative in the extracted summaries. Moreover, BERT, which is pre-trained on sentence pairs instead of documents, fails to capture long-range dependencies across a document. Instead of sentences, DISCOBERT selects sub-sentential discourse units for extractive summarization with greater granularity. To capture long-range dependencies, DISCOBERT constructs structural discourse graphs based on RST trees and coreference mentions, encoded using Graph Convolutional Networks. The experiments demonstrate that DISCOBERT outperforms other BERT-based models on popular summarization benchmarks by a significant margin.

The DISCOBERT architecture consists of a transformer-based neural network with 12 layers, and it is pre-trained on a large corpus of text using a masked language modeling objective. During training, a certain percentage of tokens in the input sequence are randomly masked, and the model is trained to predict the original token given the context of the masked token. In addition to this pre-training step, DISCOBERT also includes a novel sentence boundary detection algorithm that takes advantage of the discourse structure of the input text. The model is able to identify the boundaries between sentences based on the discourse relation between the last token of one sentence and the first token of the next sentence. This information is used to create sentence embeddings that are incorporated into

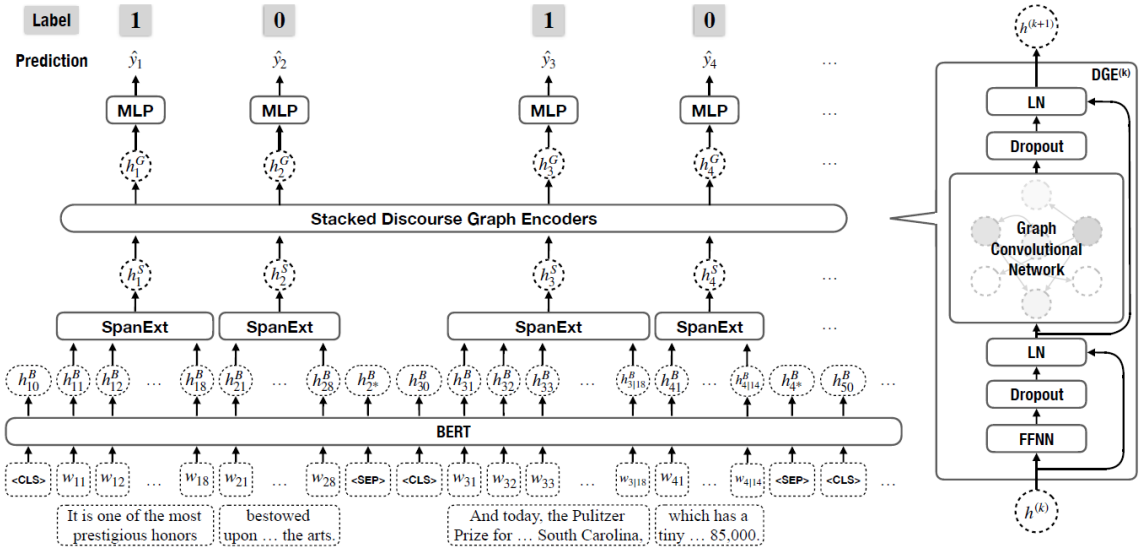

Figure 4.2: DISCOBERT (Xu et al., 2020)

the overall encoding of the input text. The sentence embeddings are generated by a separate transformer-based encoder that is specifically trained to predict the discourse relation between pairs of sentences. The model uses a binary classification task to predict whether a given pair of sentences has a coherent discourse relation, such as elaboration or contrast, or not.

Here is a breakdown of how DISCOBERT works and how it extracts important discourses as it is shown in Figure 4.2:

1. Model architecture of DISCOBERT: DISCOBERT is a deep neural network model that is based on the Transformer architecture, similar to other popular language models such as BERT and GPT-2. However, unlike these models, DISCOBERT is designed to capture the discourse-level structure of text by leveraging Stacked Discourse Graph Encoders (DGEs).

2. The Stacked Discourse Graph Encoders contain k stacked DGE blocks: DISCOBERT has k stacked DGE blocks that are used to process the input text in a hierarchical manner. Each DGE block is responsible for capturing the discourse-level structure of the input text and is designed to extract meaningful discourse-level representations.

3. The architecture of each Discourse Graph Encoder (DGE) block: The architecture of each DGE block consists of three key components: (i) a discourse graph construction module that builds a discourse graph from the input text, (ii) a discourse graph encoding module that encodes the graph structure and input text to produce discourse-level representations, and (iii) a graph attention mechanism that aggregates information from neighboring nodes in the discourse graph.

4. Final classification step: After the input text has been processed through the k stacked DGE blocks, the final classification step involves a simple linear layer that maps the discourse-level representations to the desired output classes.

Important Sentence Extraction: Specific subsets of source sentences will be emphasized using extractive models. In the training phase, to obtain the guidance signal, oracle extraction technique is used with a greedy search algorithm (Liu and Lapata, 2019) to identify the sentences in the source document that achieve the highest ROUGE scores with the reference. Then, these selected sentences are as our guidance signal g. During testing, a pretrained extractive summarization model, MatchSum (Zhong et al., 2020), which is the extractive model that improves the candidate summaries generated by BertExt (Liu and Lapata, 2019) using reranking approach, is applied for automatic prediction.

GSum: Flexible neural abstractive summarization models are known to produce coherent summaries, but their outputs can be unfaithful and difficult to control. Previous studies have attempted to enhance faithfulness and controllability by providing various types of guidance, but it remains unclear how these strategies compare to one another. To address this, a general and adaptable guided summarization framework (GSum) (Dou et al., 2021) was proposed that can effectively incorporate different forms of external guidance such as important sentences, keywords (Li et al., 2018a), key relational triples in the form of (subject, relation, object) (Jin et al., 2020; Zhu et al., 2020), and retrieved summaries (Cao et al., 2018) from the source document. Experiments on multiple datasets demonstrate that the model achieves state-of-the-art performance according to ROUGE, particularly when using

highlighted sentences as the proposed guidance. Also the guided model generates more faithful summaries and that different types of guidance can lead to qualitatively distinct summaries, thus providing a level of controllability to the learned models.

The general architecture is depicted in Figure 4.3. The model is fed with both the source documents and several types of guidance signals. The framework is flexible and could accommodate any types of guidance signals. It utilizes the Transformer model (Vaswani et al., 2017), which is composed of both encoder and decoder components. It instantiates this model with either BERT or BART.

The architecture comprises two separate encoders for the input source document and the guidance signals, respectively. Similar to the Transformer model, each encoder has $N_{enc}+1$ layers, and each layer contains a self-attention block and a feed-forward block:

$$\begin{aligned} x &= LN(x + SELFATTN(x)), \\ x &= LN(x + FEEDFORWARD(x)), \end{aligned} \tag{4.3}$$

where LN denotes layer normalization.

The encoding of the source document and guidance signals is performed independently without any interaction between the two. To reduce computation and memory requirements, the parameters of the word embedding layers and the bottom N_{enc} layers between the two encoders are shared. It is hypothesized that the differences between the source document and guidance signals are high-level and are captured at the top layers of the encoders.

Unlike the standard Transformer, the decoder attends to both the source document and guidance signal, rather than just a single input. Specifically, the decoder consists of N_{dec} layers, each comprising four blocks. Following the self-attention block, the decoder attends to the guidance signals to produce the corresponding representations. This allows the guidance signal to provide information to the decoder on which part of the source document to focus on. Subsequently, the decoder attends to the entire source document based on the guidance-aware representations. Finally, the output representation is passed through the

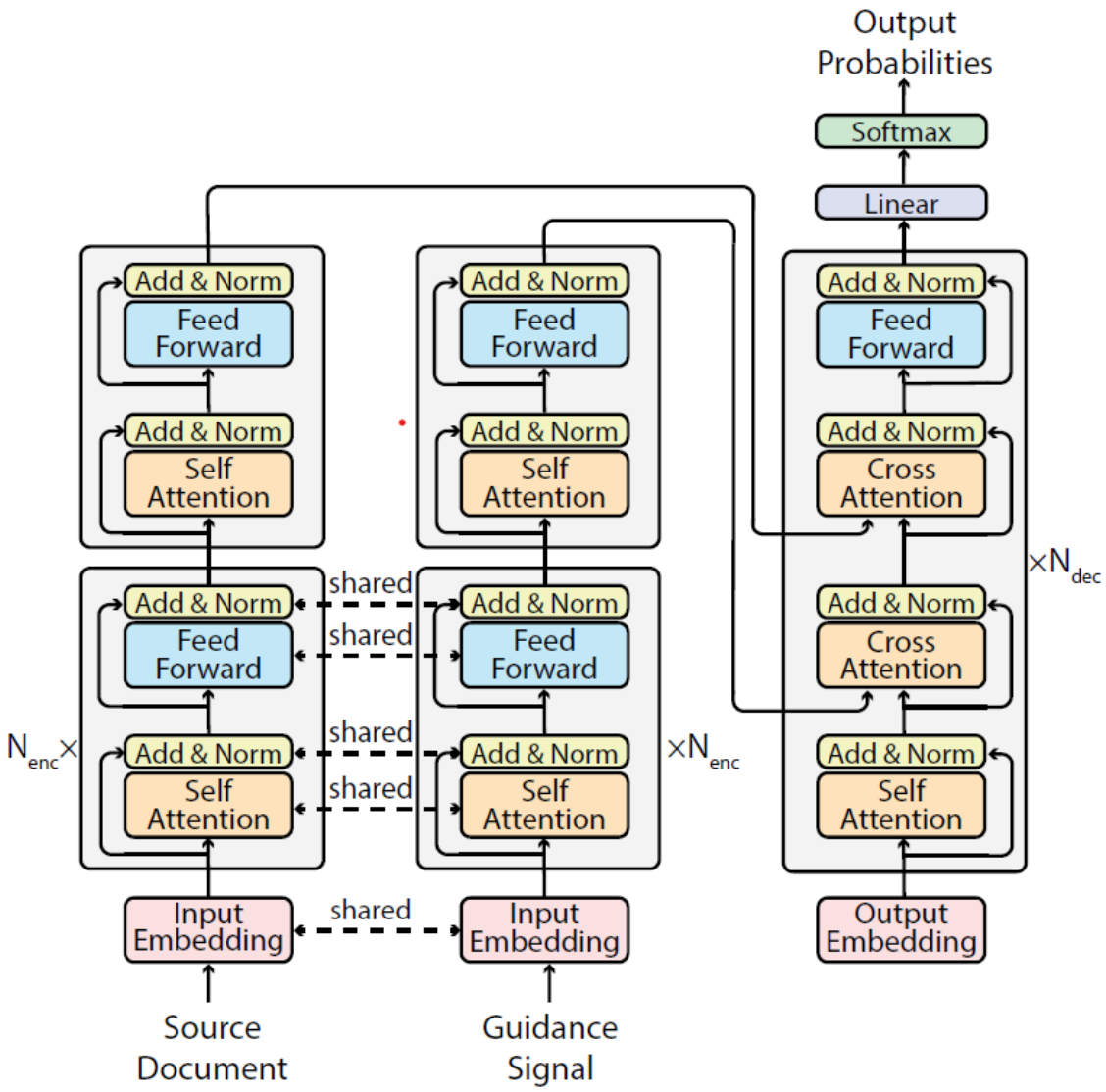

Figure 4.3: GSum (Dou et al., 2021)

feed-forward block:

$$\begin{aligned} y &= LN(y + SELFATTN(y)); \\ y &= LN(y + CROSSATTN(y, g)); \\ y &= LN(y + CROSSATTN(y, x)); \\ y &= LN(y + FEEDFORWARD(y)) : \end{aligned} \tag{4.4}$$

The second cross-attention block enables the model to supplement the input guidance signal by providing additional information, such as identifying the name of an entity by exploring co-reference chains.

Model: Figure 4.4 illustrates the improved architecture of the study. Extracted EDUs are EDUs labeld with the values zero and one. We select the EDUs from each document that have the label valued with one, which means that they are important EDUs. In our study GSum is instantiated with BART abstractive summarization. In the next chapter we will show the result of applying one guidance or multiple guidances simultaneously.

4.3 Summary

In this chapter, we explained the proposed architecture and explained the rationale for using EDUs as guidance signals to improve the decision-making process of the decoder, enhance faithfulness, and reduce hallucination in text summarization, specifically BART.

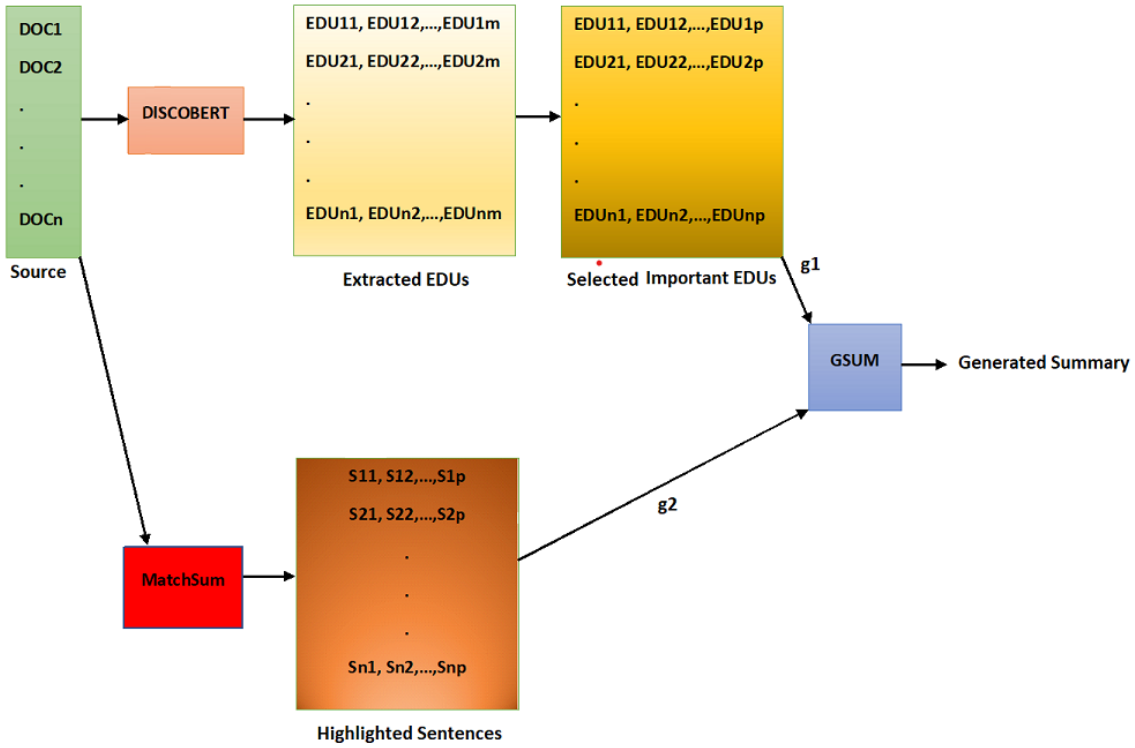

Figure 4.4: Improved architecture

Chapter 5

Experimental findings and outcomes

5.1 Introduction

This chapter presents details on the experimental settings, dataset descriptions, and evaluation metrics used. We also analyze and discuss the results of each experiment.

5.2 Data

In this study we make our experiments with CNN/Daily Mail dataset. It is a large-scale text summarization dataset comprising news articles and corresponding summaries from two leading news organizations. Its diverse range of topics and writing styles, along with the inclusion of human-written summaries, make it a comprehensive source for training and testing models. Additionally, the dataset's size and challenging nature contribute to better performance in real-world applications.

5.3 Implementation

Our summarization models are constructed with BART (Lewis et al., 2020) as the baseline summarizer using its default hyperparameter settings. GSum (Dou et al., 2021) instantiated with BART consists of 24 encoding layers, with the top layer initialized with pretrained parameters and separately trained for each encoder. While the first cross-attention block of the decoder is randomly initialized, the second cross-attention block is initialized with pretrained parameters. We make two guidances. At training time we have Oracle extracted guidance. We employ DISCOBERT (Xu et al., 2020) to select important EDUs.

Highlighted sentences during testing time will be extracted applying MatchSum (Zhong et al., 2020).

We leverage Byte Pair Encoding (BPE) tokenization to represent source, target, and guidance. By representing words as a combination of subword units, BPE can effectively handle rare or unseen words that are not present in the vocabulary. This is especially useful in text summarization tasks where the model needs to summarize a large corpus of text that contains domain-specific or rare words. Moreover, BPE can capture meaningful morphological and syntactic information in the subword units, which can further improve the model's performance on tasks such as text generation and summarization. BPE can also reduce the vocabulary size, making the model more computationally efficient and easier to train. For these reasons, BPE has become a popular technique in text summarization and is widely used in state-of-the-art summarization models.

We use different guidances in our experiment such as highlighted sentences, Important EDUs, combination of highlighted sentences with Important EDUs, and combination of EDUs. After determining the guidance on the CNN/DM dataset, we fine-tune our model on BART with oracle-extracted guidance. We then predict the guidance during testing.

5.4 Automatic Evaluation

Our evaluation process involved assessing the effectiveness of various types of guidance signals on the CNN/DM dataset with BART.

ROUGE: ROUGE (Lin, 2004) is used for automatic evaluation to determine the similarity between source and system summaries. In fact, better ROUGE score emphasizes that the generated summary includes more information from the source document.

BERTScore: Utilizing contextual embeddings, BERTScore (Zhang et al., 2020b) is a metric used for automatic evaluation of text generation. This method calculates the similarity score between each token in the candidate sentence and the reference sentence. It is effective in matching paraphrases and capturing distant dependencies and ordering; then, it

correlates better with human judgments of text quality. Then, BERTScore is less likely to be misled by surface-level similarities or paraphrases that do not convey the same meaning as the original text.

Textual Entailment: Measuring factual consistency is often done using Textual Entailment (Dagan et al., 2005) , which is a widely-used approach in various studies, such as those conducted by Falke et al. (2019) and (Maynez et al., 2020a). The main idea is that a summary should contain information that is either entailed by or neutral to the source document, while avoiding any contradictions. This score can be calculated in three methods: sentence-to-sentence (s2s), document-to-sentence (d2s), and top-to-sentence (t2s). The s2s method verifies whether each sentence in the summary is entailed by any sentence in the source. The d2s method ensures that each sentence in the summary is entailed by the source document as a whole. Finally, the t2s method determines if each summary sentence is entailed by the top k (=3) most similar sentences in paragraph, which are identified through cosine-similarities of sentence embeddings.

Question Generation & Question Answering (QGQA): Durmus et al. (2020a) and Wang et al. (2020) introduced the QGQA framework, which has been utilized in subsequent works (Maynez et al., 2020a; Dong et al., 2020a). This framework aims to detect factual inconsistencies between a summary and its source by generating questions about the summary and its source, and comparing the answers. The more matching answers, the more faithful the summary is considered. The QGQA framework consists of three models: an answer candidate selection (AS) model to identify important text spans from the summary, a question generation (QG) model to generate questions using the answer candidates, and a question answering (QA) model to answer the questions using both the source document and the generated text. Finally, the faithfulness score is computed based on the similarity of the corresponding answers.

Sentence Similarity (SentSim): To measure faithfulness, SentSim is based on the idea that the summary should convey the same information as the source document but using

different wording. This approach requires SentSim to identify similar concepts expressed with different words and distinguish between similar and contrasting or contradicting information. Here is the following sentence similarity strategy to assess faithfulness: 1) divide the source document and summary into individual sentences, 2) match each summary sentence with the most similar source sentence to compute precision, and 3) match each source sentence with the most similar summary sentence to compute recall.

5.5 Results

In this research, we utilized the high-performance computing resources provided by Compute Canada's Alliance to fine-tuned our model using 4 x NVIDIA A100 with 40 GB for Narval cluster and 4 x NVIDIA V100 with 32 GB for Cedar cluster. With its advanced hardware and software capabilities, Alliance allowed us to efficiently run large-scale experiments and optimize our deep learning models for text summarization. This access to state-of-the-art computing infrastructure significantly enhanced the speed and accuracy of our experiments, enabling us to analyze and evaluate our results with greater confidence. Specifically, we employed the Narval and Cedar clusters.

We compare the result of Abstractive Text Summarization using our proposed guidance with Dou et al. (2021) that used highlighted sentences as guidance. EDUs for each document, combination of highlighted sentences with EDUs for each document, and combination of EDUs for each document are our proposed guidances. Also we compare the results with two other models worked for improvement of faithfulness in text summarization such as CLIFF (Cao and Wang, 2021a), a contrastive learning summarization approach, and SEASON (Wang et al., 2022), a guided summarization method. We discussed CLIFF in Chapter 3. SEASON uses the allocation of salience expectation, a flexible guidance in mapping sentences to various salience degrees, to guide abstractive summarization and overcome the limitations of extractive summaries as guidance.

The evaluation metrics are calculated with a 95% interval confidence, meaning that the

Table 5.1: Comparison of summarization using ROUGE (with 95% confidence interval)

Model	ROUGE-1	ROUGE-2	ROUGE-L
BART + Sentences(G)	0.45	0.22	0.42
BART + EDUs(G)	**0.54**	**0.31**	**0.5**
BART + Sentences,EDUs(G)	0.52	0.29	0.49
BART + Concatenated EDUs(G)	**0.54**	**0.31**	**0.5**
BART	0.44	0.21	0.41
CLIFF	0.44	0.21	0.41
SEASON	0.46	0.23	0.43

Table 5.2: Comparison of summarization using different evaluation metrics (with 95% confidence interval)

Model	BERTScore	Entailement	QGQA	SentSim
BART + Sentences(G)	0.89	0.925	0.067	0.85
BART + EDUs(G)	**0.91**	0.916	0.084	0.738
BART + Sentences,EDUs(G)	0.9	**0.929**	**0.093**	**0.854**
BART + Concatenated EDUs(G)	0.9	0.917	0.086	0.743
BART	0.88	0.916	0.049	0.578
CLIFF	0.89	0.884	0.051	0.634
SEASON	0.89	0.914	0.083	0.761

reported scores have a 95% probability of falling within a certain range. This range is calculated from the sample data and provides an estimate of the plausible values for the true score in the population. A wider confidence interval indicates more uncertainty in the estimate, while a narrower one indicates more precision. By reporting evaluation metrics with a 95% confidence interval, we can quantify the level of uncertainty in our estimates and provide a range of plausible values for the true score

As shown in Table 5.1 the guided BART model with EDUs and concatenated EDUs as guidance performs the best across all three ROUGE metrics, with a score of 0.54 and 0.5 for ROUGE-1 and ROUGE-L respectively, and 0.31 for ROUGE-2. The BART model without guidance, CLIFF, and SEASON also performed relatively well, but did not outperform the BART model with EDU guidance. Then, guiding BART with important EDUs make the

model more able to capture the salient information and produce summaries that are more similar to the reference summaries.

In terms of faithfulness evaluation, the results in Table 5.2 suggest that the BART model with EDU guidance has the highest BERTScore and has more similarity between the candidate and reference summaries based on contextual embeddings. However, the BART model with Sentences and EDUs guidance has the highest scores for Entailment, QGQA, and SentSim, which evaluate the semantic coherence, answer generation, and sentence similarity of the candidate summaries with the source document, respectively.

Overall, guiding a text summarization model like BART with EDUs enhances the faithfulness of the generated summary.

5.6 Summary

This chapter elaborates on our experimentation with the guided summarization using EDUs. We explain the experimental setup, working with a larger dataset, the evaluation metrics, and the results of the experiments. Additionally, we analyze the outcomes and their implications.

Chapter 6

Conclusion

This thesis focuses on improving the faithfulness quality of BART summarization. We discuss the challenges posed by the overwhelming growth of textual information on the internet and the need for intelligent approaches to process and summarize large amounts of information. The two types of summarization approaches, extractive and abstractive, are explained. The use of deep learning approaches and pre-trained language models in improving the performance of summarization systems is discussed. The seq2seq model with an attention-based mechanism is described, along with the transformer model.

The challenges of abstractive summarization include hallucination and unfaithfulness, which may lead to inaccurate or irrelevant information being presented in generated summaries. To address these issues, we implemented a mechanism that employs a guidance signal to regulate the important EDUs and conducted two experiments to evaluate its effectiveness. Our results indicate that our approach outperforms previous models using other guidance signals, and our guided summarization based on EDUs produces more informative and faithful summaries. The use of pre-trained language models and deep learning architectures has significantly improved the performance of summarization systems. However, our research provides a promising approach to enhance the quality of automatic summarization and improve its faithfulness. Our work has the potential to impact future research and practical applications in this field, as summarization is becoming increasingly important due to the expansion of the written documents.

6.1 Future Work

While our proposed approach using important EDUs as a guidance signal has shown promising results in improving faithfulness, there is still room for improvement. One potential avenue for further exploration is incorporating additional and diverse sources of guidance to lead to more informative summaries, such as discourse markers, sentiment analysis, topic modeling or other linguistic features or representations of the text that may help to better capture the intended meaning of the source text. Additionally, continued research and development in this area could lead to more effective and accurate summarization models that better capture the key information from the source text while remaining faithful to its original meaning.

Bibliography

Kazuki Akiyama, Akihiro Tamura, and Takashi Ninomiya. Hie-BART: Document summarization with hierarchical BART. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 159–165, Online, June 2021. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2021.naacl-srw.21.

Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D Trippe, Juan B Gutierrez, and Krys Kochut. Text summarization techniques: a brief survey. *arXiv preprint arXiv:1707.02268*, 2017.

Manuel Almeida and Andr'e FT Martins. Fast and robust compressive summarization with dual decomposition and multi-task learning. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 196–206, 2013. URL http://www.newdesign.aclweb.org/anthology/P/P13/P13-1020.pdf.

Rahul Aralikatte, Shashi Narayan, Joshua Maynez, Sascha Rothe, and Ryan T. McDonald. Focus attention: Promoting faithfulness and diversity in summarization. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers)*, pages 6078–6095. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.acl-long.474. URL https://doi.org/10.18653/v1/2021.acl-long.474.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Mario Barrantes, Benedikt Herudek, and Richard Wang. Adversarial nli for factual correctness in text summarisation models. *CoRR*, abs/2005.11739, 2020. URL https://arxiv.org/abs/2005.11739.

Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. Jointly learning to extract and compress. In *Proc. of ACL*, pages 481–490, 2011. URL https://www.aclweb.org/anthology/P11-1049.

Meng Cao, Yue Dong, Jiapeng Wu, and Jackie Chi Kit Cheung. Factual error correction for abstractive summarization models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6251–6258, Online, November 2020a. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-main.501.

Meng Cao, Yue Dong, Jiapeng Wu, and Jackie Chi Kit Cheung. Factual error correction for abstractive summarization models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6251–6258, 2020b.

Meng Cao, Yue Dong, and Jackie Chi Kit Cheung. Inspecting the factuality of hallucinated entities in abstractive summarization. *arXiv preprint arXiv:2109.09784*, 2021. URL http://arxiv.org/abs/2109.09784.

Shuyang Cao and Lu Wang. CLIFF: Contrastive learning for improving faithfulness and factuality in abstractive summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6633–6649, Online and Punta Cana, Dominican Republic, November 2021a. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.532. URL https://aclanthology.org/2021.emnlp-main.532.

Shuyang Cao and Lu Wang. Cliff: Contrastive learning for improving faithfulness and factuality in abstractive summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6633–6649. Association for Computational Linguistics, 2021b. URL https://aclanthology.org/2021.emnlp-main.532/.

Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. Faithful to the original: Fact aware neural abstractive summarization. *arXiv preprint arXiv:1711.04434*, 2017. URL http://arxiv.org/abs/1711.04434.

Ziqiang Cao, Wenjie Li, Sujian Li, and Furu Wei. Retrieve, rerank and rewrite: Soft template based neural summarization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 152–161, 2018.

Jaime Carbonell and Jade Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336. ACM, 1998. doi: 10.1145/290941.291025. URL papers2://publication/uuid/1FA33AEC-2C9E-4149-B740-02A7C6C24B93.

Sihao Chen, Fan Zhang, Kazoo Sone, and Dan Roth. Improving faithfulness in abstractive summarization with contrast candidate generation and selection. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5935–5941, Online, June 2021. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2021.naacl-main.469.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

James Clarke and Mirella Lapata. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, 31:399–429, 2008. URL http://www.jair.org/media/3112/live-3112-21036-jair.pdf.

Ido Dagan, Oren Glickman, and Bernardo Magnini. The pascal recognizing textual entailment challenge. In *Proceedings of the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment*, pages 177–190. Springer, 2005.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. URL http://arxiv.org/abs/1810.04805.

Bhuwan Dhingra, Manaal Faruqui, Ankur Parikh, Ming-Wei Chang, Dipanjan Das, and William Cohen. Handling divergent reference texts when evaluating table-to-text generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4884–4895, Florence, Italy, 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1483. URL https://aclanthology.org/P19-1483.

Yue Dong, Shuohang Wang, Zhe Gan, Yu Cheng, Jackie Chi Kit Cheung, and Jingjing Liu. Multi-fact correction in abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9320–9331, Online, November 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.749. URL https://aclanthology.org/2020.emnlp-main.749.

Yue Dong, Shuohang Wang, Zhe Gan, Yu Cheng, Jackie Chi Kit Cheung, and Jingjing Liu. Multi-fact correction in abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9320–9331, 2020b.

Zi-Yi Dou, Pengfei Liu, Hiroaki Hayashi, Zhengbao Jiang, and Graham Neubig. Gsum: A general framework for guided neural abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4830–4842, 2021.

Esin Durmus, He He, and Mona Diab. FEQA: A question answering evaluation framework for faithfulness assessment in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5055–5070, Online, July 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.454. URL https://aclanthology.org/2020.acl-main.454.

Esin Durmus, He He, and Mona Diab. Feqa: A question answering evaluation framework for faithfulness assessment in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5055–5070, 2020b. doi: 10.18653/v1/2020.acl-main.454. URL http://arxiv.org/abs/2005.03754.

H. P. Edmundson. New methods in automatic extracting. *Journal of the ACM (JACM)*, 16(2):264–285, 1969. URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.83.5638&rep=rep1&type=pdf.

Alexander R. Fabbri, Chien-Sheng Wu, Wenhao Liu, and Caiming Xiong. Qafacteval: Improved qa-based factual consistency evaluation for summarization. *arXiv preprint arXiv:2112.08542*, cs, 2021a. URL http://arxiv.org/abs/2112.08542.

Tobias Falke, Leonardo F. R. Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. Ranking generated summaries by correctness: An interesting but challenging application for natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2214–2220, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1213. URL https://aclanthology.org/P19-1213.

Katja Filippova and Michael Strube. Sentence fusion via dependency graph compression. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 177–185, Honolulu, Hawaii, October 2008. Association for Computational Linguistics. URL https://aclanthology.org/D08-1019.

Katja Filippova, Enrique Alfonseca, Carlos A Colmenares, Łukasz Kaiser, and Oriol Vinyals. Sentence compression by deletion with lstms. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 360–368, 2015.

Joseph L Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378, 1971.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*, volume 1. MIT press, 2016.

Philip John Gorinski and Mirella Lapata. Movie script summarization as graph-based scene extraction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1066–1076, 2015. URL http://en.wikipedia.orghttp://anthology.aclweb.org/N/N15/N15-1113.pdf%0Ahttp://www.aclweb.org/anthology/N15-1113.

David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. English gigaword. *Linguistic Data Consortium*, 4(1):34, 2003.

Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5-6):602–610, 2005.

Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Juergen Schmidhuber. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232, 2017.

Martin T Hagan, Howard B Demuth, and Mark H Beale. *Neural network design*. PWS publishing, 2 edition, 1996.

Shashi Narayan Hardy and Andreas Vlachos. HighRES: Highlight-based Referenceless Evaluation of Summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, June 2019. URL http://arxiv.org/abs/1906.01361.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. URL http://arxiv.org/abs/1606.08415.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper/2015/file/afdec7005cc9f14302cd0474fd0f3c96-Paper.pdf.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Luyang Huang, Lingfei Wu, and Lu Wang. Knowledge graph-augmented abstractive summarization with semantic-driven cloze reward. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5094–5107, 2020.

Ting-Hao Huang, Francis Ferraro, Nasrin Mostafazadeh, Ishan Misra, Aishwarya Agrawal, Jacob Devlin, Ross Girshick, Xiaodong He, Pushmeet Kohli, Dhruv Batra, et al. Visual storytelling. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1233–1239, 2016.

Yangfeng Ji and Jacob Eisenstein. Representation learning for text-level discourse parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13–24. Association for Computational Linguistics, 2014.

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

Wojciech Kry'sci'nski, Bryan McCann, Caiming Xiong, and Richard Socher. Evaluating the factual consistency of abstractive text summarization. *arXiv preprint arXiv:1910.12840*, 2019. URL http://arxiv.org/abs/1910.12840.

Wojciech Kryściński, Bryan McCann, Caiming Xiong, and Richard Socher. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, 2020.

Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1007. URL https://aclanthology.org/P18-1007.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553): 436–444, 2015.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. Technical report, arXiv preprint arXiv:1910.13461, 2020.

Chenliang Li, Weiran Xu, Si Li, and Sheng Gao. Guiding generation for abstractive text summarization based on key information guide network. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 55–60, 2018a.

Haoran Li, Junnan Zhu, Jiajun Zhang, and Chengqing Zong. Ensure the correctness of the summary: Incorporate entailment knowledge into abstractive sentence summarization. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1430–1441, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL https://aclanthology.org/C18-1121.

Zhenwen Li, Wenhao Wu, and Sujian Li. Composing elementary discourse units in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6191–6196, Online, 2020. Association for Computational Linguistics.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Proceedings of the workshop on Text Summarization Branches Out, Post-Conference Workshop of ACL 2004*, number 1, pages 25–26, 2004. URL papers2://publication/uuid/5DDA0BB8-E59F-44C1-88E6-2AD316DAEF85.

Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A Smith. Toward abstractive summarization using semantic representations. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1077–1086, 2015.

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

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. URL http://arxiv.org/abs/1907.11692.

Shayne Longpre, Kartik Perisetla, Anthony Chen, Nikhil Ramesh, Chris DuBois, and Sameer Singh. Entity-based knowledge conflicts in question answering. *arXiv preprint arXiv:2109.05052*, 2021.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1412–1421, 2015.

William C Mann and Sandra A Thompson. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281, 1988.

Yuning Mao, Xiang Ren, Heng Ji, and Jiawei Han. Constrained abstractive summarization: Preserving factual consistency with constrained generation. *arXiv preprint arXiv:2010.12723*, 2020.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online, July 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.173. URL https://aclanthology.org/2020.acl-main.173.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919. Association for Computational Linguistics, July 2020b. doi: 10.18653/v1/2020.acl-main.173. URL https://aclanthology.org/2020.acl-main.173/.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013a.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26:3111–3119, 2013b.

David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

Feng Nan, Cicero Nogueira dos Santos, Henghui Zhu, Patrick Ng, Kathleen McKeown, Ramesh Nallapati, Dejiao Zhang, Zhiguo Wang, Andrew O. Arnold, and Bing Xiang. Improving factual consistency of abstractive summarization via question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6881–6894, Online, August 2021a. Association for Computational Linguistics. URL https://aclanthology.org/2021.acl-long.542.

Feng Nan, Ramesh Nallapati, Zhiguo Wang, Cicero dos Santos, Henghui Zhu, Dejiao Zhang, Kathleen McKeown, and Bing Xiang. Entity-level factual consistency of abstractive text summarization. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2727–2733, 2021b.

Feng Nan, Cícero Nogueira dos Santos, Henghui Zhu, Patrick Ng, Kathleen R. McKeown, Ramesh Nallapati, Dejiao Zhang, Zhiguo Wang, Andrew O. Arnold, and Bing Xiang. Improving factual consistency of abstractive summarization via question answering. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 6881–6894. Association for Computational Linguistics, 2021a. doi: 10.18653/v1/2021.acl-long.536. URL https://doi.org/10.18653/v1/2021.acl-long.536.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1206. URL https://aclanthology.org/D18-1206.

Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4812–4829, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.383. URL https://aclanthology.org/2021.naacl-main.383.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106, 2005.

Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. In *Proceedings of the 6th International Conference on Learning Representations*, 2018. URL http://arxiv.org/abs/1705.04304.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Matthew E Peters, Mark Neumann, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, 2018. URL http://allennlp.org/elmo.

Dragomir Radev, Doo Soon Tam, Sonja Winkelmann, and David Zajic. Evaluation challenges in large-scale document summarization. In *Proceedings of the ACL 2013 workshop on Models for Large Scale NLP*, pages 1–6, 2013.

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

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015.

Pengjie Ren, Furu Wei, Zhumin Chen, Jun Ma, and Ming Zhou. A redundancy-aware sentence regression framework for extractive summarization. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 33–43, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee. URL https://aclanthology.org/C16-1004.

Alexander M. Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 379–389, 2015.

Masoud Jalili Sabet, Philipp Dufter, François Yvon, and Hinrich Schütze. Simalign: High quality word alignments without parallel training data using static and contextualized embeddings. *arXiv preprint arXiv:2004.08728*, 2020.

Itsumi Saito, Kyosuke Nishida, Kosuke Nishida, and Junji Tomita. Abstractive summarization with combination of pre-trained sequence-to-sequence and saliency models. *arXiv preprint arXiv:2003.13028*, 2020.

Mike Schuster and Kaisuke Nakajima. Japanese and korean voice search. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2012.

Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

Abigail See, Peter J Liu, and Christopher D Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, 2017.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. A graph-to-sequence model for amr-to-text generation. *arXiv preprint arXiv:1805.02473*, 2018.

Yun-Zhu Song, Hong-Han Shuai, Sung-Lin Yeh, Yi-Lun Wu, Lun-Wei Ku, and Wen-Chih Peng. Attractive or faithful? popularity-reinforced learning for inspired headline generation. *arXiv preprint arXiv:2002.02095*, 2020.

Karen Sparck Jones and Julia R. Galliers. *Evaluating Natural Language Processing Systems: An Analysis and Review*. Springer Science & Business Media, 1995.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

Duyu Tang, Bing Qin, and Ting Liu. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1422–1432, 2015.

Xiangru Tang, Alexander R Fabbri, Ziming Mao, Griffin Adams, Borui Wang, Haoran Li, Yashar Mehdad, and Dragomir Radev. Investigating crowdsourcing protocols for evaluating the factual consistency of summaries. *arXiv preprint arXiv:2109.09195*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Alex Wang, Kyunghyun Cho, and Mike Lewis. Asking and answering questions to evaluate the factual consistency of summaries. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5008–5020, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.450. URL https://aclanthology.org/2020.acl-main.450.

Chaojun Wang and Rico Sennrich. On exposure bias, hallucination and domain shift in neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3544–3552, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.326. URL https://aclanthology.org/2020.acl-main.326.

Fei Wang, Kaiqiang Song, Hongming Zhang, Lifeng Jin, Sangwoo Cho, Wenlin Yao, Xiaoyang Wang, Muhao Chen, and Dong Yu. Salience allocation as guidance for abstractive summarization. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 2022.

Ronald J Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

Sam Wiseman, Stuart M Shieber, and Alexander M Rush. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on EMNLP*, pages 2253–2263, 2017. doi: 10.18653/v1/D17-1239. URL http://www.aclweb.org/anthology/D17-1239.

Wenhao Wu, Wei Li, Xinyan Xiao, Jiachen Liu, Ziqiang Cao, Sujian Li, Hua Wu, and Haifeng Wang. BASS: Boosting abstractive summarization with unified semantic graph. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6052–6067, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.472. URL https://aclanthology.org/2021.acl-long.472.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. Technical report, Google, 2016. URL http://arxiv.org/abs/1609.08144.

Wen Xiao, Patrick Huber, and Giuseppe Carenini. Do we really need that many parameters in transformer for extractive summarization? discourse can help! In *Proceedings of the First Workshop on Computational Approaches to Discourse*, pages 124–134, Online, 2020. Association for Computational Linguistics.

Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. Discourse-aware neural extractive text summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5021–5031, Online, 2020. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11441–11451. PMLR, 2020a.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2020b.

Ye Zhang and Byron Wallace. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *arXiv preprint arXiv:1510.03820*, 2015.

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

Chenguang Zhu, William Hinthorn, Ruochen Xu, Qingkai Zeng, Michael Zeng, Xuedong Huang, and Meng Jiang. Boosting factual correctness of abstractive summarization with knowledge graph. *arXiv preprint arXiv:2004.14900*, 2020.