Text Encoding and Decoding from Global Perspectives

Thesis submitted in accordance with the requirements of the University of Liverpool for the degree of Doctor in Philosophy by

Ye Ma

September 2022

Abstract

As an important application scenario of deep learning, Natural Language Processing (NLP) is receiving more and more attention and developing rapidly. Learning representation for words or documents via neural networks is gradually replacing feature engineering in almost all text-related applications. On the other hand, how to decode these representations or encodings is also very vital for sequence-to-sequence text generation tasks such as Neural Abstractive Summarization (NAS), Neural Machine Translation (NMT), etc. Towards a more comprehensive representation and decoding strategy, this dissertation explores several global perspectives that previous studies ignored. We treat *global* as a relative concept that indicates higher-level knowledge conducive to enriching representation or improving decoding. However, its specific definition may vary in different tasks.

In text representation or encoding, *global* refers to relatively higher-level context information. There usually are three natural contextual relationships for mapping words or documents into latent space, namely (1) co-occurrence relationships between words, (2) coherence relationships between sentences, and (3) subordinate relationships between documents/sentences and their words. Beyond these naturally occurring contexts, there are possibly hidden context relationships between dependent documents from the perspective of the whole corpus (i.e., the global perspective). Although we often assume that documents in a corpus are independent of each other, the assumption may not be valid for some corpora like news corpora, since events reported by news documents interact in the real world. To capture the global-contextual information, we construct a news network for the whole corpus to model the latent relationships between news. A network embedding algorithm is then designed to produce news representations based on the above-mentioned subordinate relationship and news dependency. Besides, such a cross-document relationship plays a vital role in some specific tasks which need to represent or encode a cluster of multiple documents, e.g., Multi-document Summarization (MDS). Some studies concatenate all documents as a flat sequence, which is detrimental to modeling the cross-document and long-term dependency. To alleviate the two problems, we design a Parallel Hierarchical Transformer (PHT), whose local and global attention mechanisms can simultaneously capture cross-token and cross-document relationships.

On the other hand, *global* in text decoding refers to a higher-level optimum, i.e., the global optimum relative to the local optimum. Under the fact that the neural text gener-

ator is almost impossible to generate the whole sentence at once, the heuristic algorithm – beam search has been the natural choice for text decoding. Inevitably, beam search often gets stuck of local optimum as it decodes word-by-word. Although global optimum is hard to touch directly, it is feasible to conduct a one-shot prediction of how the global optimal hypothesis attends to the source tokens. A global scoring mechanism is then proposed to evaluate generated sentences at each step based on the predicted global attention distribution, thus calibrating beam search stepwise to return a hypothesis that can assign attention distribution to the source in a more-near global optimal manner. Decoding with global awareness improves the local optimum problem to enhance the generation quality significantly, and it can be developed and used in various text generation fields.

Key Words: Text representation, Sequence-to-sequence generation, Decoding strategy, Transformer, Global attention, Global context.

Acknowledgements

I am extremely grateful for the advice and support of my supervisor Professor Lu Zong who have helped guide me on both academic and personal decisions in the past several years. Her guidance has helped me in all the time of my research and in writing of my papers.

Besides, I extend my thanks to all my co-supervisors and co-authors: Professor Jionglong Su, Professor Kaizhu Huang, Mr Yikang Yang, Dr Peiwan Wang, Mr Zixun Lan. They have offered me an intensively positive attitude and the technical and writing support.

Finally, I would also like to thank Xi'an Jiaotong-Liverpool University for providing a Ph.D. scholarship.

Publications & Patents

Publications

- Ye Ma, Zixun Lan, Lu Zong, Kaizhu Huang, "Global-aware Beam Search for Neural Abstractive Summarization", Advances in Neural Information Processing Systems (NeurIPS), 2021, 34: 16545-16557.
- Ye Ma, Lu Zong, Yikang Yang, Jionglong Su, "News2vec: News network embedding with subnode information", Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2019, 11: 4843-4852.
- Ye Ma and Lu Zong, "Neural Abstractive Multi-Document Summarization: Hierarchical or Flat Structure?", Proceedings of the Second International Workshop of Discourse Processing, 2020, 12: 29-37.
- Peiwan Wang, Lu Zong, Ye Ma, "An integrated early warning system for stock market turbulence", Expert Systems with Applications, 2020, 153: 113463.
- Ye Ma, Lu Zong, Peiwan Wang, "A novel distributed representation of news (drnews) for stock market predictions", 2020.
- Ye Ma, Lu Zong, "Integrated Node Encoder for Labelled Textual Networks", 2020.

Patents

- Ye Ma, Lu Zong, Jionglong Su, "A construction method and application of news feature vector", CN201910397143.3.
- Ye Ma, Lu Zong, "Method and system of generating summary by hierarchical transformer based on multi text", CN202010609274.6.
- Ye Ma, Lu Zong, "Abstract neural network generation method based on known attention distribution", CN202010610681.9.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Text encoding and decoding is the backbone of model sequence-to-sequence, which first maps the sequence into a latent space and then decodes it as a different sequence. The concept of encoding in NLP is similar to representation or embedding. In this thesis, all of them mainly refer to the dense and continuous vectors, which represent unstructured text data numerically by the neural networks to facilitate the subsequent mathematical calculations. Moreover, encoding or representation can be regarded as points embedded in an abstract and high-dimensional latent space, which is the origin of the name *embedding*. This dissertation generally promotes distributional text representation and sequence-to-sequence with global information, and their developments are introduced as follows.

To be exact, distributional representation denotes an approach to represent text relying on the hypothesis that objects in the same contexts tend to have similar embeddings. It should be mentioned that distributional representation is different from distributed representation. The latter refers to a form of text representation (i.e., the continuous and low-dimensional vectors), which is exactly opposite with discrete representation. In other words, distributional representation could either be distributed or discrete. A typical distributional representation is the word co-occurrence matrix $\boldsymbol{M} \in \mathbb{N}^{|V_W| \times |V_C|}$ where $|V_W|$ and $|V_C|$ indicate the size of word vocabulary and context vocabulary, and contexts $c \in V_C$ for word $w \in V_W$ are often defined as the words surrounding it in the window-sized L. This matrix counts the number of occurrences of any word pair $\#(w, c)$. Each row could be regarded as a distributional word representation that is sparse, high-dimensional, and discrete. Its distributional attribution is not changed even though we transform these sparse-and-discrete vectors to the dense-and-continuous vectors (i.e., distributed represen-

tations) by means of dimension reduction such as well-known Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA). Besides, a fashionable way is to train a neural network to estimate the distributed representations like GloVe [72] and Word2vec [68]. Compared with traditional dimensionality reduction methods, the neural estimation can stably achieve distributed vectors of higher quality with lower computational costs [47]. Overall, the vector attributions of distributed and discrete representations are contrary, and these attributions are independent of whether they belong to the distributional representation.

On the other hand, distributed and distributional representation has been the natural choice for learning-based NLP tasks since they help deal with two major challenges for deep networks. Firstly, it is much more efficient for the deep network to calculate distributed vectors than discrete vectors. The co-occurrence matrix often has millions of dimensions of sparse word vectors. It should be catastrophic to operate multiple linear or non-linear transformations to such vectors. By comparison, the dimension-reduced distributed vectors are more convenient to be computed. Secondly, the pre-trained distributional representation is high in solving text-related deep learning tasks because deep learning heavily relies on large volumes of datasets. In contrast, distributional representation can be trained on a large corpus in an unsupervised way. Since context relationships naturally exist in documents, we can train distributional representations by unsupervised learning to mine text knowledge as much as possible. This knowledge then improves downstream supervised tasks where labeled examples are limited. This process is exactly the classical two-stage learning paradigm in NLP: pre-training first and fine-tuning next. Many pre-training word embeddings and models based on the distributional hypothesis have become landmark researches in NLP including Word2vec [68], GloVe [72], ELMo [74], BERT [20], etc. In Chapter 3, we train neural text embeddings on a huge news corpus to mine the latent-but-valuable context information between news documents, which brings a new challenge: we need to make these invisible contexts concrete in advance instead of leveraging the existing contexts directly like word embeddings.

Text encoding or representation aims to understand text and map it in a latent space, while text decoding converts these hidden vectors back to readable sentences which follow different distributions from that of input sentences. Sequence-to-sequence model is often composed of the encoder and decoder, and could be regarded as the conditional text generation $p(\boldsymbol{y}|\boldsymbol{x})$. Two neural networks with different parameters serve as the encoder and decoder to estimate the probabilities of target sequences $\boldsymbol{y} = (y_1, \cdots, y_T)$ based on

the input sources $\boldsymbol{x} = (x_1, \cdots, x_N)$. Since it is unreasonable to assume that words are independent, $p(\boldsymbol{y}|\boldsymbol{x})$ should be factorized as $= \prod_{t=1}^{T} p(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t})$ where each $p(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t})$ is the probability over all words in the vocabulary.

Sequence to sequence learning with neural networks sprang up between 2013 - 2014. The representative work [89] uses the multilayered Long Short-Term Memory (LSTM) to encode the English sentences to fixed-sized vectors and then uses another deep LSTM to decode French sentences from these vectors. Bahdanau et al. [3] argue that using a fixed vector to represent the whole input limits the translation performance. They propose an attentive mechanism to align the decoded token to the corresponding token in the input sentence at each step. Instead of decoding the sentence from a fixed vector, the attention model allows the decoder to obtain a dynamic vector as the generation proceeds. Meanwhile, the learned attention weights can be used to explain the contribution of each input word to output. Another bottleneck is the problem of out-of-vocabulary (OOV). Most time, the encoder and the decoder share the same word vocabulary, whose size is limited. Otherwise, the softmax operation would take up too much space to run. In this case, the words out of vocabulary have to be replaced by a unified special token *UNK*, disabling the decoder to generate these OOV words. Attention models could also deal with this problem by treating the attention distribution of OOV words as their probability distribution so that these OOV words have the chance to be generated together with words in the vocabulary [95, 85]. It is like that OOV words in the input are copied to the output directly rather than chosen from the original fixed-sized vocabulary. Though such treatment alleviates the OOV problem to some extent, it reduces the instability and efficiency of training. The more reasonable way is using a subword tokenizer, which splits OOV words into sub-words stored in the vocabulary [86].

The attention mechanism has been an auxiliary module of the RNN family for a long time, implying that this kind of model still suffers from the defects of RNN, including excessive time complexity in teacher forcing training and limited gains as the model deepens. The emergence of Transformer [91] solves the problems above and marks that NLP has officially entered a real era of deep learning. Transformer abandons any forms of recurrent networks and adopts multilayered attention models to construct the backbone that allows more parallelization and achieves a new state-of-the-art sequence-to-sequence (seq2seq) translation. The intuitive advantage of Transformer over the RNN family is that it has stronger representation and generalization ability as the model goes deeper. At the same time, like RNN, Transformer is also designed based on two valuable prior knowledge of

modeling sequential data. The first is the translation invariance, that is, the same set of parameters is used to process the input of any time step. Besides, the model should integrate sequential information into the final representation. That is why the recurrent structure has become the natural choice in the past. To abandon recursion but preserve the prior knowledge, the Transformer adds additional positional encodings into token embeddings.

Although Transformer has already been the mainstream architecture in modeling sequential data, it was questioned when it first came out. Specifically, compared with RNN, the training of Transformer takes up more memory space and is not relatively stable. However, it is destined to be the preferred model framework in the era of pre-training thanks to its strong representation ability and less sequential operations. Moreover, some studies have improved the inherent disadvantages of Transformer. For example, OpenAi proposes a sparse Transformer [13] to save the memory space required to run the model. Based on this technique, they trained oversized auto-regressive language models (i.e., the family of GPT [7, 78]) by stacking many layers of sparse Transformer. On the other hand, some studies have found that the difficulty of transformer training lies in the normalization after the residual connection, which inspires them to make some improvements there to improve the stability of loss decline, including pre-normalization [104] and ReZero [2]. Generally, the backbone of the sequence-to-sequence model has been changed from RNN to Transformer and then to pre-training Transformer. However, it should be mentioned that not all pre-training Transformer models can be regarded as seq2seq models. Strictly speaking, the seq2seq model should consist of both encoder and decoder. Still, some pre-training models only adopt Transformer-encoder like BERT [20], or only use Transformer-decoder like GPT [7, 78]. Since this thesis focuses on improving the seq2seq model (Chapter 4) and its decoding strategy (Chapter 5) with global information, we mainly discuss pre-trained Transformer encoder-decoder in Chapter 2 including T5 [79], BART [48], PEGASUS [109], etc.

This dissertation intends to improve text representation (or encoding) and text decoding from global perspectives. This global concept is a widely used improvement direction, but actually, it has various meanings which depend on specific cases. In this thesis, we propose some novel global perspectives that few studies explored previously to alleviate some problems of locality bias, such as local context problems in text representation (Chapter 3 & 4) or local optimum problem in text decoding (Chapter 5). Although there is no specific definition of global information, it essentially reflects higher-level knowledge compared with

the knowledge learned by the existing model. Taking the studied problem as an example, the local context of a news document are often its inner words, which only include limited information because they does not capture latent connections between news events. To obtain more context knowledge than that located in a single document, we construct a news network on the whole news corpus. These additional contexts, namely global contexts, represent higher-level knowledge as it comes from relationships among news documents instead of that inside the single document. The definition of global information is also changed in text decoding. Firstly, since the decoding is a step-wise process, the local information often refers to the generated tokens before the current decoding step, getting the decoding stuck of the local optimum. Under this premise, we define global information as many characteristics that can describe the complete generated sentences. In general, we emphasize that the global perspective is a relative concept and is hard to give a unified definition. Besides, since our global perspectives are inspired by the problems in the existing methods of text representation and text decoding, they will be the major discussions in Chapter 2.

1.1 Summary of Remaining Chapters

Chapter 2 Background. This chapter is divided into four parts. We firstly introduce LSTM [35] and Transformer [91], which are the most commonly-used sequential models in many NLP fields, including but not limited to text representation and text generation. Subsequently, we review text representation approaches, sequence-to-sequence models, and decoding strategies. There are generally three kinds of representing text numerically according to the divergences in techniques or purposes: discrete representation, distributional representation, and distributed representation. These three representation forms are not mutually exclusive, i.e., one text representation may belong to multiple forms. On the other hand, sequence-to-sequence [89] was first proposed in 2013 - 2014 and is widely applied in many source-based text generation tasks, e.g., machine translation, abstractive summarization, image captioning, etc. This thesis summarizes sequence-to-sequence development and emphatically introduces the popular pre-trained seq2seq models. Finally, the decoding strategy revolves around sampling search and deterministic search.

Chapter 3 News Network Embedding based on Local and Global Context. This chapter introduces a novel distributed news representation based on the distributional hypothesis that news with similar contexts should be represented as two adjacent points

in the latent space, which is extended from Harris's hypothesis that words occurring in the same contexts tend to have similar meanings [32]. On the other hand, since representing news could be considered as a document embedding task, we can use words within the document as its contexts to train the document vector [45]. However, we argue that this context fails to capture the latent relationships between news, making the final news representation monotonous. To enrich news embedding, We treat key entities and actions inside a news document as the local context and the hidden associations among news as the global context. We construct a large graph for the whole news corpus to achieve news embeddings based on both local and global contexts. Specifically, the news network comprises two kinds of nodes: news nodes and entity or action nodes. News nodes are not directly connected but only indirectly linked by their shared entity or action nodes. Besides, the edge weights refer to the importance of entities or actions to the connected news documents. We further enrich the news embedding with various label information, including news type, sentiment, length, and so on, by decomposing a news node into many subnodes. An inductive network embedding approach is then adopted to learn the representations of subnodes which are later composed as the news embedding. We applied the converged news embedding into downstream tasks and observed that this additional information is conducive to financial prediction and news recommendation.

Chapter 4 Parallel Hierarchical Transformer with Local and Global Cross Attention. This chapter introduces a hierarchical transformer encoder-decoder model, which enjoys a distinct that there operate two cross attention mechanisms in parallel, namely local cross attention and global cross attention. This design is to process long text sequences or multi-documents with relatively lower memory usage and faster inference speed. Specifically, we split a long document or a large document cluster into many chunks or paragraphs, and then we used a shared encoder to represent them respectively. The shared encoder outputs the context-aware word embeddings for each paragraph and fuses these dense word vectors as the paragraph embedding, which would also be fed into the decoder. Subsequently, the decoder attends to the source locally and globally through the cross attention models. The local cross attention is to model how the decoder pays attention to the tokens of each paragraph. At the same time, the global one captures the attention distribution given by the decoder to the paragraphs. To ensure the translation invariance (i.e., the parametric function is independent of the number of paragraphs) and facilitate subsequent computation, we use these global attention weights as a pooling function to integrate these local ones for all paragraphs. We applied the Parallel Hier-

archical Transformer (PHT) in the Multi-document Summarization. Compared with Flat Transformer, our architecture is more conducive to capturing cross-document relationships, saving more memory, and enjoying faster inference speed.

Chapter 5 Decoding with Awareness of Global Attention Distribution. This chapter develops a calibrated beam-based algorithm with awareness of the global attention distribution for neural abstractive summarization, aiming to improve the local optimality problem of the original beam search in a rigorous way. Specifically, a novel global protocol is proposed based on the attention distribution to stipulate how a global optimal hypothesis should attend to the source. A global scoring mechanism is then developed to regulate beam search by aligning the local attention distribution with the predicted global attention distribution. This novel design enjoys a distinctive property, i.e., the global attention distribution could be predicted before inference, enabling step-wise improvements on the beam search through the global scoring mechanism. Extensive experiments on nine datasets show that the global (attention)-aware inference significantly improves state-of-the-art summarization models even using empirical hyper-parameters. The algorithm is also proven robust as it remains to generate meaningful texts with corrupted attention distributions. Although this chapter mainly focuses on the global-aware inference in summarization, this decoding strategy can be generalized to more source-based text generation tasks (not just text-to-text tasks), thanks to the fact that attention models are widely used to represent various unstructured data [24, 92].

1.2 Contributions

This thesis takes three chapters (namely Chapter $3-5$) to introduce how to use the global concept to enrich distributional news embedding, restructure Flat Transformer [91] and calibrate beam search. Our contributions are summarized as follows.

In Chapter 3, we propose a distributional news embedding approach News2vec whose main advantages are threefold,

- This work explores novel context relationships for distributional news embedding, that is, the latent connections between news. Previous works often assume each news document as an independent individual and use contexts inside news to train their embeddings, such as the subordinate relationships between news and their inner words. We argue that this assumption may not be reasonable since news events

should affect each other in the real world. Therefore, we treat the hidden connections between news as global contexts, distinguished from local contexts which describe the subordinate relationships between news and their inner words. To facilitate joint modeling of both local and global contexts, we embrace the two contexts into a large network constructed on the whole news corpus.

- Inspired by Subword [5] which can enrich word embeddings and solve out-of-vocabulary (OOV) problems conveniently, we propose Subnode model for network embedding. Subnode splits the news node into many news characteristics subnodes, which share the same advantages as Subword. First, Subnode could enrich news embeddings with subnode information such as news categories, news polarities, etc. Besides, it offers an easy solution to create embeddings for unseen (news) nodes outside the trained network.

- By visualizing our news embeddings and applying them to news-related downstream tasks, we confirm that News2vec can integrate global contexts and many subnode features into the final news vectors. Besides, this additional information beyond local contexts can further promote news-based stock trend prediction and news recommendation.

In Chapter 4, we restructure vanilla Transformer as Parallel Hierarchical Transformer by introducing local and global cross attention.

- It is widely known that vanilla Transformer [91] has two main aspects worthy of improvement. The first is that it is challenging to learn the long-term dependency typically when the token length is over 2000 [60]. The second is that its training takes up too much memory, especially when the input length becomes longer. These two issues become particularly important in the large-scale multi-document summarization, because we need to concatenate all documents as a flat sequence and then feed it into the Transformer model. To deal with the two problems, the hierarchical architecture with parallel cross attention mechanisms is designed to represent local information (cross-token relationships in a document) and global information (cross-paragraph relationships). The hierarchical structure uses a shared encoder to capture many short-term dependencies truncated from the long-term dependency. Besides, the parallel local and global cross attention are conductive to the inference speed.

- The effectiveness of the Parallel Hierarchical Transformer model is investigated relative to a variety of aspects, in terms of the ability to capture cross-document relationships, computational efficiency, and improvements on the summarization quality. Primarily, we design a simple and effective experiment to examine the model's capacity to capture cross-document relationships.

In Chapter 5, we develop a decoding strategy named global-aware inference, which is a calibrated beam-based algorithm with awareness of the global attention distribution. Its contributions are presented as follows.

- We argue that the limitation of beam search roots from its defect in finding the global optimal hypothesis. We improve the algorithm by proposing a global protocol to regulate beam search step-by-step. This paper is the first to predict and deploy the global attention distribution to calibrate the inference in a rigorous way, thus returning a hypothesis that attends to source tokens in a more near-global optimal manner. In contrast, previous works [102, 27, 55, 85, 56] try to use attention distributions to improve beam search, but ignore that the global attention distribution is predictable.

- A novel global scoring mechanism is designed to evaluate the generated sequences at each step based on the local attention distribution and the predicted global attention distribution. As theoretically justified, its major component can be elegantly integrated into beam search in the form of a probability so that there is merely $\mathcal{O}(\mathcal{K})$ time complexity increased in each step.

- The proposed algorithm with global awareness manifests a robust and plug-and-play property in enhancing beam search for neural abstractive summarization. Without requiring any model or parameter modification, the global-aware inference shows excellent performance in generating meaningful texts, even if the attention distribution is corrupted or not of its own. Further, it is identified that summaries generated by global-aware inference are both higher-quality and different from beam search hypotheses. More interestingly, we find that the generation style of a dataset could be transferred by the designated global attention distribution. For instance, summaries of higher abstractness for CNN/DM could be generated by only replacing its global attention distribution with a highly abstractive distribution during inference.

- On the empirical side, we show that the proposed global-aware inference can stably and significantly boost two state-of-the-art summarization models BART [48] and PEGASUS [109] to produce higher quality summaries on nine datasets, even if only the empirical hyper-parameters are used.

Chapter 2

Background

This chapter introduces LSTMs and Transformers as the preliminaries before reviewing text representation and sequence-to-sequence generation. This is because many studies in the two fields are based on LSTM [35] and Transformer [91] or their variants.

2.1 Preliminaries

2.1.1 LSTM

Long short-term memory [35] is a significant variation of recurrent neural network (RNN) [81] used in the field of deep learning. It is usually used to process sequences of data, such as entire sentences or videos, instead of single data points like words and images. LSTMs partially solve the vanishing gradient problem of the classical vanilla RNNs by introducing memory cells, and it has dominated various sequential tasks until the emergence of Transformer [91]. In the following part, we will introduce the vanilla RNN, the long short-term memory mechanism, and its variations such as Bidirectional-LSTM [84] and GRU [14].

Recurrent neural network

Recurrent neural network (RNN) is one of several basic neural networks. As shown in Figure 2.1, it processes the sequential data recursively by keeping feeding the outputed hidden states h_t to the next time step. It is worth noting that RNN uses the same set of parameters to estimate the probabilities of data of all time steps, which brings a big advantage that it can easily deal with variable-length sequences of inputs. On the other

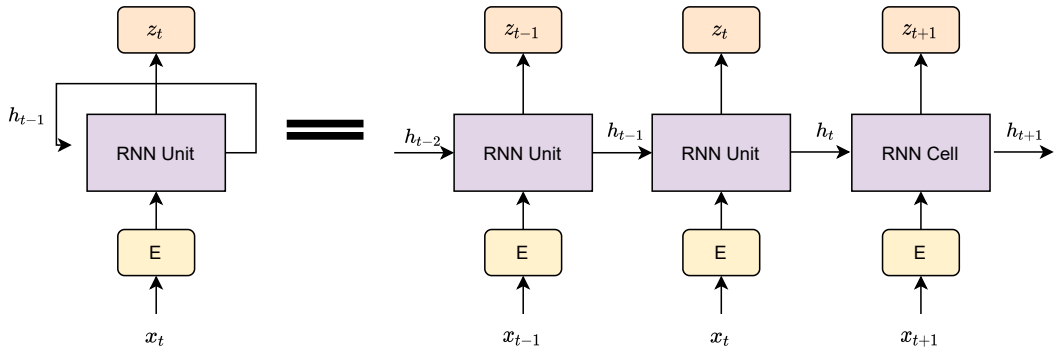


Figure 2.1: Recurrent neural network.

hand, the recurrent mechanism is a sequential operation so that it has the natural choice for modeling temporal data.

Sentences $\boldsymbol{x} = (x_1, x_2, \cdots, x_T)$ are very common time series and the word x_t indicates the data point at the t_{th} time step. Since words are unstructured data and we need to represent them numerically, there is an embedding layer ahead of the RNN unit to convert words into dense vectors. Normally, each word is firstly assigned with a unique position index and is converted a vector $\boldsymbol{e}_t \in \mathbb{R}^{d_{in}}$ of the embedding matrix $\boldsymbol{E} \in \mathbb{R}^{|\mathcal{V}| \times d_{in}}$ by looking up the corresponding index, where $|\mathcal{V}|$ is the vocabulary size and d_{in} the dimension of the input vectors. The RNN unit could be formulated as:

$$\boldsymbol{h}_t = \tanh(\boldsymbol{W}_h \boldsymbol{e}_t + \boldsymbol{U}_h \boldsymbol{h}_{t-1} + \boldsymbol{b}_h) \tag{2.1}$$

$$\boldsymbol{z}_t = \sigma(\boldsymbol{W}_z \boldsymbol{h}_t + \boldsymbol{b}_z) \tag{2.2}$$

where $\boldsymbol{W}_h \in \mathbb{R}^{d_h \times d_{in}}$, $\boldsymbol{U}_h \in \mathbb{R}^{d_h \times d_h}$ and $\boldsymbol{b}_h \in \mathbb{R}^{d_h}$ are learnable parameter matrices and vectors. Therefore, at each step, the input vector $\boldsymbol{w}_t$ and the latest hidden states $\boldsymbol{h}_{t-1}$ are firstly transformed linearly with the bias $\boldsymbol{b}$. The transformed results are then projected to a new vector space by the non-linear activation function (often use $\tanh(\cdot)$ function) to achieve the updated hidden vector $\boldsymbol{h}_t \in \mathbb{R}^{d_h}$. The hidden states are updated and delivered iteratively, carrying the previous information to enrich the representation of the local input. If we need the output $\boldsymbol{z}_t$ at each step, these hidden vectors are required to pass through a fully-connected layer with learnable parameters $\boldsymbol{W}_z$ and $\boldsymbol{b}_z$. The reason

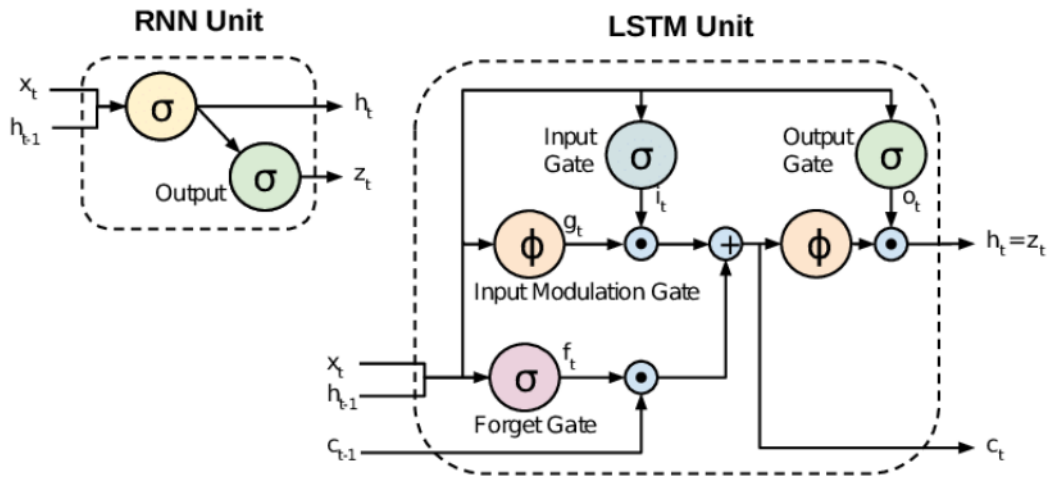


Figure 2.2: RNN unit versus LSTM unit.

why hidden vectors are not outputted directly is that hidden vectors and output vectors should belong to different semantic spaces since they have different functions. Such an operation could enhance the expressive ability of RNN significantly. Since the transmission is unidirectional, the output of each step only incorporates partial information of the current position and the left of that. This perceptual field is in line with the left-to-right reading behaviors of humans. On the other hand, an equally common practice is to output the vector only at the last step, which exactly represents the whole sequence. This operation is often used to represent the whole sequence, such as sentence-level text classification.

Long short-term memory mechanism

Although RNNs, in theory, can track any long-term dependency in the input sequence, it suffers from the problem that the gradients of back-propagation would be vanished (i.e., they are zeros) or be exploded (i.e., they tend to infinite). Its variant model LSTMs can solve the problem partially. We firstly introduce the detailed structure of LSTMs before explaining why the long short-term memory could solve mathematically.

The differences between the RNN unit and LSTM unit are diagrammed in Figure 2.2. Primarily, LSTMs still follow the recursive flow but are equipped with a memory cell, an input gate, an output gate, and a forget gate. The cell preserves the long-term information,

and the three gates work to regulate which part of information should be forgotten or continued to be transmitted. Extended from the RNN unit, the detailed computational process of the LSTM cell is described as follows,

$$\boldsymbol{f}_t = \sigma\left(\boldsymbol{W}_f \boldsymbol{e}_t + \boldsymbol{U}_f \boldsymbol{h}_{t-1} + \boldsymbol{b}_f\right) \tag{2.3}$$

$$\boldsymbol{i}_t = \sigma\left(\boldsymbol{W}_i \boldsymbol{e}_t + \boldsymbol{U}_i \boldsymbol{h}_{t-1} + \boldsymbol{b}_i\right) \tag{2.4}$$

$$\boldsymbol{o}_t = \sigma\left(\boldsymbol{W}_o \boldsymbol{e}_t + \boldsymbol{U}_o \boldsymbol{h}_{t-1} + \boldsymbol{b}_o\right) \tag{2.5}$$

$$\widetilde{\boldsymbol{c}}_t = \tanh\left(\boldsymbol{W}_c \boldsymbol{e}_t + \boldsymbol{U}_c \boldsymbol{h}_{t-1} + \boldsymbol{b}_c\right) \tag{2.6}$$

$$\boldsymbol{c}_t = \boldsymbol{i}_t \circ \widetilde{\boldsymbol{c}}_t + \boldsymbol{f}_t \circ \boldsymbol{c}_{t-1} \tag{2.7}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \circ \tanh(\boldsymbol{c}_t) \tag{2.8}$$

where the initial states are $\boldsymbol{c}_t = \boldsymbol{0}$ and $\boldsymbol{h}_0 = \boldsymbol{0}$ and the operator $\circ$ indicates the element-wise product. There are three sigmoid$(\cdot)$ activation functions for the gate vectors $\boldsymbol{f}_t$, $\boldsymbol{i}_t$ and $\boldsymbol{o}_t$ and their values are thus between 0 and 1, which can better indicate how much information passes through. For example, a value of 0 means *no information passed*, and a value of 1 means *all information passed*. Besides, an output gate regulates the transformation from the memory vectors to the hidden states. It should be mentioned that the output of LSTM unit z_t is the hidden vector h_t equivalently. This is because the hidden states of LSTMs have experienced multiple non-linear transformations and are enough to represent a variety of information.

Why LSTM?

The advantage of LSTMs over RNNs, intuitively, is the former could regulate the information transmission at each time step, which is a very important function for real-world time series because events at different time nodes should have different importance. Taking sentences as an example, each work plays a different role in representing the sentence.

When using the RNN unit to represent a useless word, its parameters are expected to be trained to make $\boldsymbol{h}_{t+1} = h_t$. It is tough to obtain such parameters, especially since RNNs need the same set of parameters to deal with all time steps. In contrast, the LSTM unit is well qualified for the work by making the input gate $\boldsymbol{i}_t$ close to $\mathbf{0}$. In other words, this advantage helps LSTMs to have a stronger representational ability.

Further, LSTMs alleviate the problem of gradient vanishment in RNNs training. Reviewing the RNNs formula Eq. 2.1, we define the error at step t as l_t, then its gradient is calculated as follows:

$$\frac{\partial l_t}{\partial \boldsymbol{U}} = \sum_{i=0}^{i=t} \frac{\partial l_t}{\partial \boldsymbol{z}_t} \frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{h}_t} \frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_i} \frac{\partial \boldsymbol{h}_i}{\partial \boldsymbol{U}} \tag{2.9}$$

$$\frac{\partial \boldsymbol{h_t}}{\partial \boldsymbol{h}_i} = \prod_{k=i+1}^{k=t} \frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{h}_{k-1}} \tag{2.10}$$

$$\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{h}_{k-1}} = \tanh'(\boldsymbol{W}_h \boldsymbol{e}_k + \boldsymbol{U}_h \boldsymbol{h}_{k-1} + \boldsymbol{b}_h) \boldsymbol{U}_h \tag{2.11}$$

It should be mentioned that $\tanh'(\cdot) = 1 - \tanh^2(\cdot)$ so that $\tanh'(\cdot) \in [0, 1]$. In this case, if $\boldsymbol{U}$ is smaller than 1, the cumulative multiplication value $\frac{\partial \boldsymbol{h_t}}{\partial \boldsymbol{h}_i}$ will tend to be zero as i becomes far from t, namely gradient vanishment. In other words, it is very hard for vanilla RNNs to learn the long-term dependencies. Gradient explosion, on the other hand, appears when $\boldsymbol{U}$ is larger then 1.

Moving back to LSTMs, $\frac{\partial \boldsymbol{c}_k}{\partial \boldsymbol{c}_{k-1}}$ can be simplified to $\Delta_k + \boldsymbol{f}_k$, whose value is case by case. That is to say, gradients could be larger than 1 at some steps and less than 1 at other steps, depending on whether the neural network wants to preserve or forget this part of information. RNNs fail to do so since the gradient of each step depends on $\boldsymbol{U}$, which is shared by all time steps. Overall, LSTMs have two advantages over RNNs with respect to backpropagation. The first is that its additive-updated strategy of memory-cell states makes the gradient propagation more reasonable. Besides, The gating unit can decide how many gradients to forget, and they can take different values at different times. These values are learned through hidden states and input functions.

Bidirectional LSTMs

We often use RNNs or LSTMs to process left-to-right sequences because this order is in line with human behaviors. However, when the machine creates a representation for the

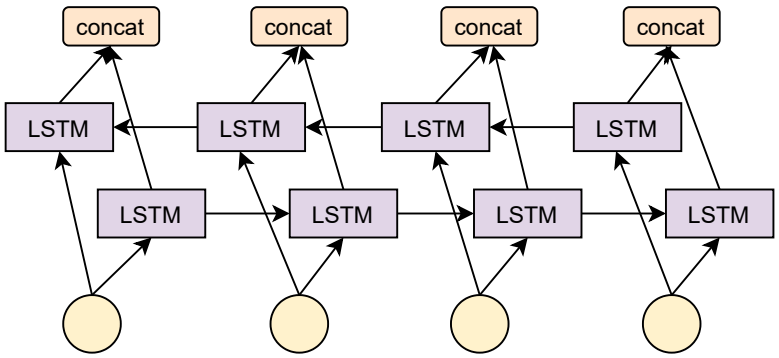


Figure 2.3: Bidirectional LSTM.

word at step t, this order means it can only represent the contextual relationships between the current word and previous words. This limits the network to extract enough context information to form strong representations. To allow the LSTMs output $\boldsymbol{z}_t$ to know the contexts after t, Bi-RNNs [84] are proposed to transmit hidden states in both left-to-right and right-to-left order. Specifically, the output of a Bi-LSTM layer $\boldsymbol{z}_t$ is concatenated by the forward and reverse hidden states, namely $\overrightarrow{\boldsymbol{z}}_t$ and $\overleftarrow{\boldsymbol{z}}_t$, as shown in Figure 2.3

Bidirectional LSTMs can better understand and represent text, whether word-level or document-level representations. It should be mentioned that when representing the whole sentence using Bi-LSTMs, its final vector should be concatenated by the last hidden states of the forward and reverse directions.

Gated Recurrent Unit (GRU)

Reviewing the LSTMs formula, it seems that h_t is not needed to be transmitted to the next time step because c_t has included the optimized hidden states. Therefore, Cho et al. propose GRU to simplify the LSTM unit [14].

$$\boldsymbol{u}_t = \sigma\left(\boldsymbol{W}_u \boldsymbol{e}_t + \boldsymbol{U}_u \boldsymbol{h}_{t-1} + \boldsymbol{b}_u\right) \tag{2.12}$$

$$\boldsymbol{r}_t = \sigma\left(\boldsymbol{W}_r \boldsymbol{e}_t + \boldsymbol{U}_r \boldsymbol{h}_{t-1} + \boldsymbol{b}_r\right) \tag{2.13}$$

$$\widetilde{\boldsymbol{h}}_t = \tanh\left(\boldsymbol{W}\boldsymbol{e}_t + \boldsymbol{U}[\boldsymbol{r}_t \circ \boldsymbol{h}_{t-1}] + \boldsymbol{b}_c\right) \tag{2.14}$$

$$\boldsymbol{h}_t = (\mathbf{1} - \boldsymbol{u}_t) \circ \widetilde{\boldsymbol{h}}_t + \boldsymbol{u}_t \circ \boldsymbol{h}_{t-1} \tag{2.15}$$

where $\boldsymbol{h}_t$ is similar to the memory cell states $\boldsymbol{c}_t$ of LSTMS and r_t corresponds to the output gate $\boldsymbol{o}_t$. The difference is that r_t acts on the last hidden states $\boldsymbol{h}_{t-1}$ (i.e., a pre-filter) in GRUs but on the current cell states $\boldsymbol{c}_t$ (i.e., a post-filter). Besides, GRUs reduce the forget gate $\boldsymbol{f}_t$ and the input gate $\boldsymbol{i}_t$ to a single update gate $\boldsymbol{u}_t$ which is a trade-off operation. This is reasonable because the two gates are opposite exactly. Overall, GRU is an alternative to LSTM, with a more concise structure and competitive performance.

2.1.2 Transformer

Before Transformer [91], gated RNNs such as LSTM and GRUs almost dominated the NLP applications. Although the additional attention mechanism often appears in these models, it only serves as an auxiliary module to promote the expressive ability and to provide some explanations for feature importance, and RNN is still compulsory to represent the critical sequential information. However, the emergency of Transformer highlights the fact that the attention mechanism itself can also capture the sequential relationships and match the performance of RNNs with attention, that is, *Attention is all you need* [91].

Transformer is not only an alternative to the RNN family but has actually increasingly become the choice mode of NLP problems. The core reasons are that Transformers allow more parallelization significantly and have stronger representational ability somehow. It enjoys a very important distinct that it does not need to process sequential data in order while still representing these sequential relationships. This is because Transformers add additional positional embeddings to the word vectors so that words in the same position could have some similarities in their vector forms. Then, the self-attention module models the position-aware contexts for each word in parallel. It should be mentioned that parallelization is only achievable during teacher forcing training, where the whole sentence could be predicted in one shot with gold sentences available. However, there still needs to predict word by word from left to right in the Transformer decoding.

Benefiting from the parallelization and the powerful expressive ability as the layer goes deeper, it becomes possible to train language models on larger datasets than before.

Its emergence brings NLP to the era of pre-training, and many unsupervised pre-trained Transformer models [20, 7] raise performances of almost all NLP downstream tasks to unprecedented levels. In this paper, we mainly introduce the basic structure of Transformer [91] and give details as to why choose self-attention and some interesting designs in the far-reaching paper.[1]

Transformer encoder-decoder structure

Transformer was firstly introduced by the paper *Attention is all you need* in 2017, where it follows the sequence-to-sequence architecture (i.e., encoder and decoder) and is applied into the machine translation. As claimed in the paper title, it abandons the recurrent structure to obtain more parallelization. To also represent the sequential relationships, it designs a positional embedding $\boldsymbol{p}_t$ to add into the token embedding $\boldsymbol{e}_t$. This is to, for example, make two different words have some vector similarities if they are at the same position of sentences. Also, it can tell the decoder which position of the word should be generated during inference. These position-aware token embeddings are then fed into Transformer.

The core component of both Transformer encoder-decoder is the Multi-head Attention, which is used to calculate three attention modules in Transformer, namely the (bi-directional) self-attention, cross attention, and uni-directional (or auto-regressive) self-attention. Firstly, the multi-head attention can be uniformly represented as:

$$head_i = \text{softmax}\left(\frac{(\boldsymbol{Q}\boldsymbol{W}^Q)_i(\boldsymbol{K}\boldsymbol{W}^K)_i^\top}{\sqrt{d_{head}}}\right)(\boldsymbol{V}\boldsymbol{W}^V)_i \tag{2.16}$$

$$\text{MultiHead}(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \text{Concat}(head_1,\cdots,head_h)\boldsymbol{W}^C \tag{2.17}$$

where $\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V} \in \mathbb{R}^{n\times d}$ are complete query, key and value matrices. Each is stacked by a set of word vectors. Therefore, the first dimension n is the sequence length, but note that their lengths can be different. The multi-head module firstly uses three projection matrices $\boldsymbol{W}^Q,\boldsymbol{W}^K,\boldsymbol{W}^V \in \mathbb{R}^{d\times d}$ to map them into corresponding semantic spaces. They are then split on the second dimension, for example $\boldsymbol{Q}\boldsymbol{W}^Q$ is split to several $(\boldsymbol{Q}\boldsymbol{W}^Q)_i \in \mathbb{R}^{n\times d_{head}}$. In this case, the context matrix on the i_{th} head is denoted as $head_i$, and each head can carry different context semantics from each other. These context heads are then concatenated and fused by the linear transformation. The aforementioned whole process is available in

[1] Figure 2.4 and 2.5 are copied from the original paper [91].

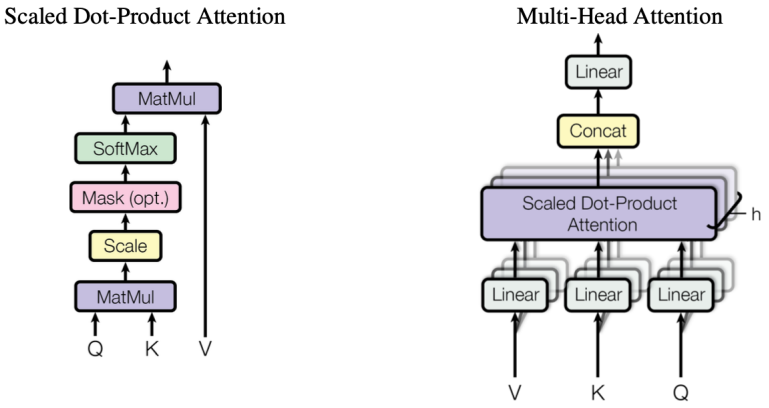


Figure 2.4: Multi-head attention module.

Figure 2.4.

In the encoder, the model only uses self-attention. That is to say, the initial three input matrices ($\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}$) of the multi-head attention are the same at each encoder layer, namely the output of the last layer or the position-enriched token embeddings at the beginning. Besides, the attention matrix here (i.e., the output after the softmax operation) is fully connected, meaning each word can capture the complete context information from both left-to-right and right-to-left. Its outputs are then fed into a feed-forward network to obtain position-wise representations, which act on every position independently and identically. Its inner structure is two linear transformations with the Relu activation function,

$$\mathrm{FFN}(\boldsymbol{x}) = \max(0, \boldsymbol{x}\boldsymbol{W}_1 + \boldsymbol{b}_1)\boldsymbol{W}_2 + \boldsymbol{b}_2 \tag{2.18}$$

The output of the feed-forward function is then added up with the layer input to be the output of that layer after the layer normalization. Transformer encoder is exactly a recurrent operation of the computational flow mentioned above, as shown on the left side of Figure 2.5

In the decoder, each layer mainly consists of two multi-head attention modules, namely the auto-regressive multi-head attention (i.e., the masked multi-head attention in Figure 2.5) and the cross multi-head attention to connect the encoder and decoder. Specifically, auto-regressive indicates that the decoder generates sentences word by word, and its output of each step is a probability distribution over the vocabulary. Since every probability

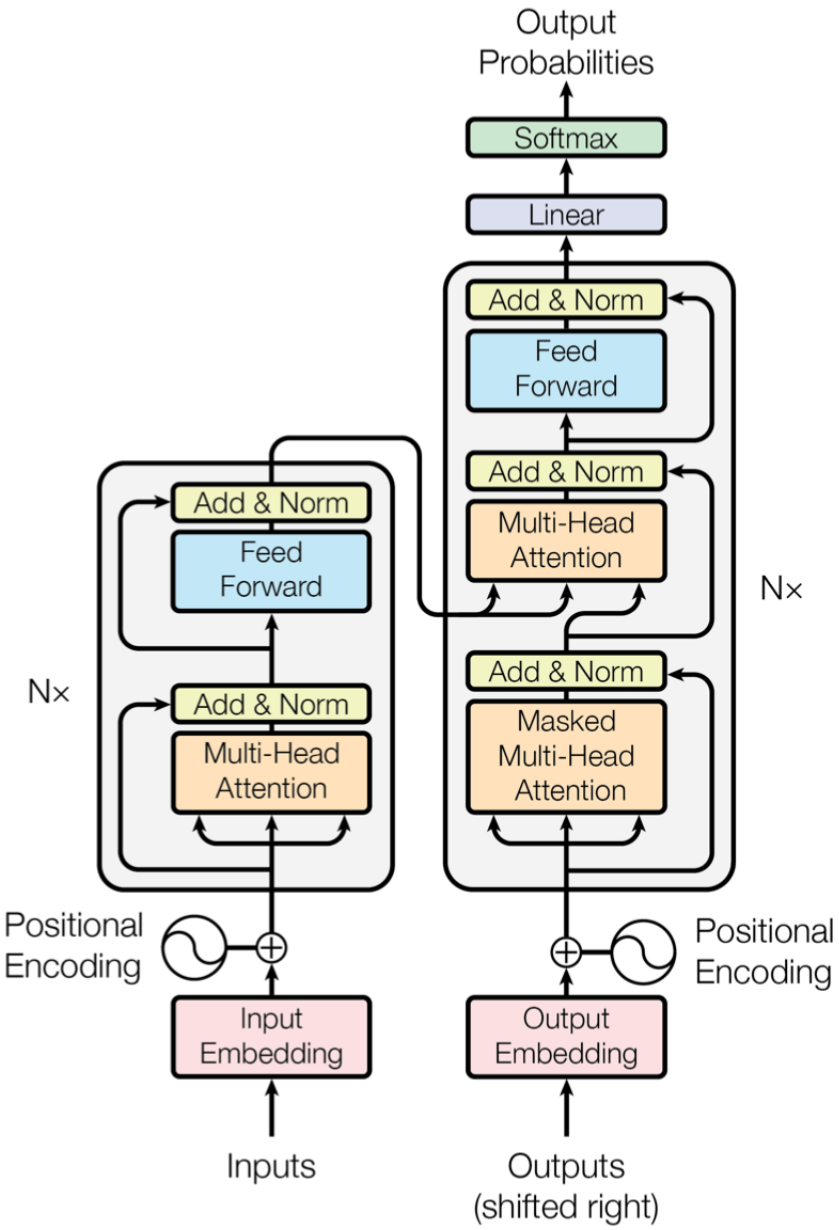


Figure 2.5: Transformer encode and decoder.

distribution is calculated based on the words generated previously, the first multi-head attention should be masked to make each word attend to previous information only. Similar to the self-attention in the encoder, here $\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}$ are still the same and refer to the output of the last layer or the position-enriched token embeddings at the beginning. The difference is that its attention matrix is masked halfway along the diagonal. As to the hinder cross multi-head attention, its query becomes different from the key and value. The query is the output of the auto-regressive self-attention module, while both key and value are the encoder output. This cross attention is to model how the decoder attends to the source tokens at each time step.

Why self-attention

This part has been discussed in the original paper [91]. In general, there are three reasons to choose attention as the model backbone.

- *Lower time complexity for common NLP tasks.* The computational flow of Eq. 2.16 can be simply formalized as $(n \times d)(d \times n)(n \times d) \longrightarrow n^2 \times d$, so the complexity per layer is $\mathcal{O}(n^2 \cdot d)$. By contrast, that of recurrent is $\mathcal{O}(n \cdot d^2)$ and convolution is $\mathcal{O}(k \cdot n \cdot d^2)$. For common NLP task, their sentence lengths n are often smaller than 256, while their dimensions d are usually set to greater to 512. Therefore, transformer layer has lower complexity in most cases.

- *None of the sequential operations during training.* As described before, Transformer leverages the positional encodings to make up sequential relationships lost due to abandoning recurrent. Transformer still needs the recurrent operation during inference, where it takes the last predicted word as the following step input and uses the same set of parameters to calculate them. The distinct of Transformer is that it can disable this operation at the training stage when the gold targets are available. It designs an interesting attention masking mechanism to model the recurrent operation and can output the whole sentence at once during training. Therefore, the sequential operation complexity of Transformer is only $\mathcal{O}(1)$, much smaller than the $\mathcal{O}(n)$ complexity of conventional recurrent networks.

- *Wider perceptual field.* The original paper uses the Maximum Path Length to measure the perceptual field. For example, the maximum path length of RNNs is $\mathcal{O}_n$, which means there need n steps to model the first token and the last token jointly. In other

words, RNNs only process local information at each step and can not form global modeling until the final step. Similar situations also exist in convolution networks, but they form layer-wise rather than step-wise. By contrast, Transformer has a global perceptual field at each layer. Moreover, its input data of each time step can directly obtain the information from any other time step without step-by-step or layer-by-layer propagation. Abstractly, Transformer regards input sentences as special graphs which take words as nodes and are fully connected. In this case, any two words can be adjacent nodes to each other, and the special graph network (i.e., Transformer) aggregates the information of the whole graph at each layer.

Why scaled attention

A thoughtful design of Transformer self-attention is that it scales the attention logits before softmax by dividing it by $\sqrt{d}$, as put in Eq. 2.16. This design has improved the training stability of Transformer both theoretically and experimentally and has been the widely-used mechanism in the dot-product attentive models. From a theoretical perspective, it maintains the input vectors and output vectors of attention models in similar level of variance and at the same time prevent gradient vanishment when appearing an extremely high logit value.

Firstly, the dot-product attention can be simply denoted as:

$$\hat{y}_i = \text{softmax}(\frac{\exp(x_i)}{\sum_i \exp(x_i)}) \tag{2.19}$$

Since the exponential function is a concave function, it is easy to prove that the final attention distribution $\hat{\boldsymbol{y}}$ would be near a one-hot vector once the logit x_i is larger than others significantly. On the other hand, the gradient of softmax function $\hat{\boldsymbol{y}} = g(\boldsymbol{x})$ can be calculated as

$$\frac{\partial g(\mathbf{x})}{\partial \mathbf{x}} = \text{diag}(\hat{\boldsymbol{y}}) - \hat{\boldsymbol{y}}\hat{\boldsymbol{y}}^\top \quad \in \mathbb{R}^{d\times d} \tag{2.20}$$

$$\frac{\partial g(\mathbf{x})}{\partial \mathbf{x}} = \begin{bmatrix} \hat{y}_1 & 0 & \cdots & 0 \\ 0 & \hat{y}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \hat{y}_d \end{bmatrix} - \begin{bmatrix} \hat{y}_1^2 & \hat{y}_1\hat{y}_2 & \cdots & \hat{y}_1\hat{y}_d \\ \hat{y}_2\hat{y}_1 & \hat{y}_2^2 & \cdots & \hat{y}_2\hat{y}_d \\ \vdots & \vdots & \ddots & \vdots \\ \hat{y}_d\hat{y}_1 & \hat{y}_d\hat{y}_2 & \cdots & \hat{y}_d^2 \end{bmatrix} \tag{2.21}$$

Then we assume $\hat{\boldsymbol{y}} \approx [1, 0, \cdots, 0, 0]$ where y_1 is the extreme value. In this case, the above

matrix becomes

$$\frac{\partial g(\mathbf{x})}{\partial \mathbf{x}} \approx \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} - \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} = \mathbf{0} \tag{2.22}$$

Therefore, the gradient disappears to 0 when a certain order of magnitude of the input is large, making it difficult to update the parameters.

So far, we have explained why the scaling can help to alleviate gradient vanishment, but it seems that we only need to allocate a positive value larger than 1 instead of a specific scaled ratio like the sqrt of model dimension. This special ratio is actually to meet another expectation to the neural network. That is, the input hidden states of each layer are better to have a similar variance as their outputs, and this variance is often set to 1. According to the assumption, the inputs of softmax function, namely elements q, k in the query and key vector, respectively, should follow independent standard normal distributions $\mathcal{N}(0, 1)$. Then the softmax function firstly calculates the sum of their element-wise products, i.e., $\sum_i^d q_i \cdot v_i$. At this time, the variance is $\sum_i^d D(q_i)D(v_i)$ which is exactly equal to d when the two random variables are independent and are of mean value 0. With the scaling operation of $\sqrt{d}$, the output variance is reduced to the input variance 1 again. Finally, it is conducive to reducing gradient vanishment and explosion that the variances of inputs and outputs are kept at a similar level. In short, gradient explosion means that a small input disturbance causes a large fluctuation of the output, and the gradient vanishing shows that even a large input change can not cause the change of the output. Obviously, if the variance of input and output can be maintained at the same and low level, these two extremes will be greatly reduced.

2.2 Text Representation

This section includes two parts to review Discrete Representation and Distributional Representation, respectively. We are not going to introduce Distributed Representation alone because its definition is so broad (i.e., the dense, low dimensional continuous vector) that almost all vectors produced by neural networks can be called distributed representation. Since this thesis studies neural text embedding, the distributionally representing methods

Table 2.1: Example of one hot representations.

I	[1, 0, 0, 0, 0, 0, 0, 0]
Like	[0, 1, 0, 0, 0, 0, 0, 0]
Natural	[0, 0, 0, 0, 0, 0, 1, 0]
Language	[0, 0, 0, 1, 0, 0, 0, 0]
Processing	[0, 0, 0, 0, 0, 0, 0, 1]

presented next all belong to the distributed representation.

2.2.1 Discrete Representation

This is to represent text as a discrete vector, which is interpretable but often sparse and high-dimensional.

One-hot representation

This is an extremely sparse vector and is the most straightforward way to represent words. We firstly set a fixed-sized vocabulary where each word has a fixed index. The one-hot word vector is the same length as the vocabulary, and all of its elements are zeros except for the position of the corresponding word where the value is 1.

This is a very simple and straightforward word representation and is also the base of many neural word embedding methods. However, as the word vector to use directly, the information it contains is very limited. Specifically, any two one-hot word vectors are always orthogonal so that it fails to capture the semantic similarity between words. Besides, it reflects the information that is limited in whether the word appears and lacks further insights, such as whether the word is important.

Table 2.1 provides an example for one-hot representations. We assume there are only eight words in the demo vocabulary so that the lengths of one-hot vectors are also eight. The vocabulary is ordered, and here *I* is the first word in the vocabulary, *Like* is the second, and *Processing* is the last one. Normally, the vocabulary is sorted according to the frequency of words in the whole corpus.

Bag of words (BOW)

BOW is essentially the summation of all one-hot vectors of the document and is thus the document representation. In the BOW vector, each value denotes the frequency of the

Table 2.2: BOW representations of *I Like Natural Language Processing*.

Word frequency BOW	[1, 1, 0, 1, 0, 0, 1, 1]
TF-IDF BOW	[0.2, 0.5, 0, 1.2, 0, 0, 1.5, 2]

corresponding word in the document. Therefore, it can partially reflect which words are more important for the document, even though the word frequency may not be the best measure for word importance. Besides, the pattern of word frequency enables the BOW to measure the similarity between documents.

Actually, BOW is not limited to the word frequency. To form more reasonable representations for documents, we often replace the word frequency with the term frequency-inverse document frequency (tf-idf). Specifically, although word frequency can represent word importance to some extent, it may mistakenly regard some high-frequency stop words (e.g., *the, a, etc.*) as important words. TF-IDF, on the premise of considering the word frequency, takes the frequency of the word in the whole corpus (i.e., idf) into the measurement index. An important word for the document should first be highly-frequent and then be lowly-frequent in other documents. For instance, *the* is often very common in every document, which means that it is destined not to have a high tf-idf score. By contrast, even if a proper noun does not frequently appear in a document, it may have a higher tf-idf score because it is rare in other documents. Similar to one-hot representation, we provide two examples for word frequency BOW and tf-idf BOW in Table 2.2.

Finally, there are still many limitations in BOW. Firstly, the sparse vector is not conducive to saving and calculating the semantic information of the text. Besides, as a sentence representation, BOW fails to capture the word order information and cross-word context information, hindering it from representing semantic similarity sufficiently.

N-Gram

Each word is independent in the bag of words representation mentioned above, ignoring the word order problem. Adding an n-gram feature can obtain local context information to enrich text representation.

In short, traditional BOW splits documents into many words and combines them into a vocabulary after removing repetitions, while N-gram splits documents using a sliding window. For example, bi-gram would split the sentence *I Like Natural Language Processing* into *I Like*, *Like Natural*, *Natural Language* and *Language Processing* . Similarly, these

bi-grams are firstly sorted in a vocabulary and assigned with fixed indices. Bags of bi-grams are then constructed by counting their frequencies of appearing in a document. Its advantage is that these n-grams consider the word order information and local contexts. Taking the word *play* as an example, traditional BOW only counts its number but ignores that this word may have different meanings under different contexts such as *play basketball* versus *play piano*. But what follows is the size explosion of the vocabulary, bringing much computation burden to the input and output layer.

2.2.2 Distributional Representation

In the last part, we introduce the discrete text representation. We can see their construction is straightforward and explainable, but it is not reasonable to mathematically model them directly due to over-sparsity. More often than not, these words or n-grams of one-hot forms are used as the original inputs to create the distributional representations, which are often distributed and can be better to capture semantic similarities between words or sentences.

Word2vec

As mentioned before, the distributional hypothesis refers to that words with similar contexts should have similar word vectors. In other words, if we input two semantically similar words into the context predictor, their outputs should be similar. Based on this, Mikolov et al. [68] proposed two context classifiers of different architectures to obtain distributed word vectors. Notably, these classifiers are not to predict the context words, really, but to get the trained parametric matrix, i.e., the word embedding matrix.

The two context classifiers are CBOW (Continuous Bag-of-words) and Skip-gram, which are presented in Figure 2.6 (it is copied from the original paper [68]). Both structures consist of a word embedding matrix and a context projection matrix, and they have the same shapes normally. Multiplying the one-hot word vectors with the word embedding matrix, these words are converted as dense and low-dimensional vectors. In the beginning, these vectors are randomly initialized, and they are updated constantly by gradient descent until they are able to predict contexts well. For CBOW, the more accurate description is that it uses the context words to predict the ontology word. However, it is also reasonable to explain its output word as the context word of the input words. Anyway, both CBOW and skip-gram take the projection matrix in the input layer as the final word embeddings, and that in the output layer is used to measure the similarity between the input words and

their contexts. Since the context matrix often has the same shape as the embedding matrix and there are usually nearly a million words in the embedding vocabulary, it is tough to calculate the probability distribution over the common softmax function efficiently. To reduce computational costs, Mikolov et al. adopted hierarchical softmax [68] firstly and then proposed negative sampling. The latter is a more widely-used training approach when it comes to Word2vec.

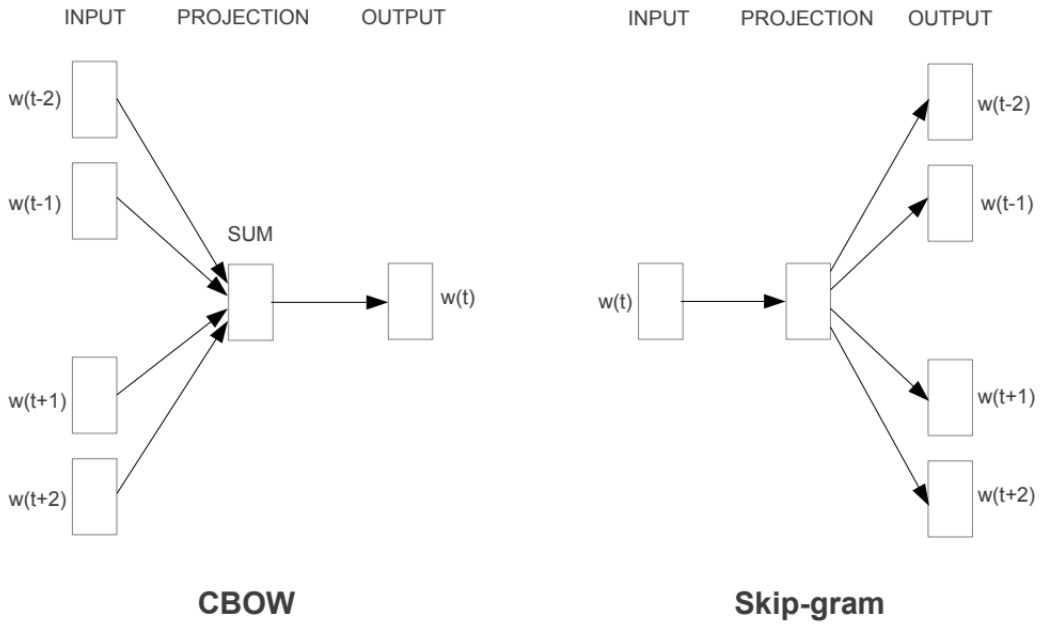


Figure 2.6: CBOW and Skip-gram

This thesis gives more details to the Skip-gram model with negative sampling (SGNS), the most widely-used word embedding model. In the skip-gram model, there are a corpus of words $w \in V_W$ and a set of context words $c \in V_C$, where V_W, V_C refer to word vocabulary and context vocabulary, respectively. It should be mentioned that both vocabularies often have the same size and can be regarded as the same object. We distinguish them here in order to explain its principle better. Subsequently, for a word w_i in the corpus, its contexts are the words surrounding it in a fixed-sized window $w_{i-L}, \cdots, w_{i-1}, w_{i+1}, \cdots, w_{i+L}$. We collect all pairs (w, c) of observed words and corresponding contexts into the cluster D. Next, we define $\#(w, c)$ as the number of times that one pair (w, c) appears in D, $\#c =$

$\sum_{w' \in V_W} \#(w', c)$ and $\#w = \sum_{c' \in V_C} \#(w, c')$ as the frequency of c and w appearing in D. By the way, we can construct the co-occurrence matrix $\mathcal{M}$ with $\#(w, c)$.

As mentioned before, the skip-gram model is composed of a word embedding matrix and a context projection matrix, and both are parametric. With them, the words and contexts are converted to dense vectors $\boldsymbol{w}, \boldsymbol{c} \in \mathbb{R}^d$. Under the distributional hypothesis, we need to make words with similar contexts closer in the embedding space and farther when their contexts are different. Then, we assume there are two words w_j, w_k, and the latter has a context word c. To extend their vector distance, we need to maximize the distance between dot-products $\boldsymbol{w}_j \cdot \boldsymbol{c}$ and $\boldsymbol{w}_k \cdot \boldsymbol{c}$, where w_j is the negative word of c. Identically, the negative sampling approach needs to sample the in-context words and out-of-context words as positive and negative samples for each word. Subsequently, it uses $\sigma(\boldsymbol{w} \cdot \boldsymbol{c})$ to calculate the probability scores for samples. The σ function scales the score into range $[0, 1]$, and normally SGNS trains parametric matrices to force the score of positive samples to 1, and negative samples to 0. Overall, the objective of SGNS for single word pair (w, c) is to maximize

$$\log \sigma(\boldsymbol{w} \cdot \boldsymbol{c}) + k \cdot \mathbb{E}_{c_N \sim P_D} \left[\log \sigma \left(-\boldsymbol{w} \cdot \boldsymbol{c}_N \right) \right] \tag{2.23}$$

where c_N is the out-of-context words, and the function increases as $\boldsymbol{w} \cdot \boldsymbol{c}$ becomes larger and $\boldsymbol{w} \cdot \boldsymbol{c}_N$ becomes smaller. In practice, the expectation is calculated by the average of negative pairs after k times samplings. These negative samples are drawn from D based on the empirical unigram distribution $P_D(c) = \frac{\#(c)}{|D|}$, i.e., the distribution over word frequencies. The total objective then sums over the single objectives of observed (w, c) pairs in the corpus,

$$\mathcal{L} = \sum_{w \in V_W} \sum_{c \in V_C} \#(w, c) \left(\log \sigma(\boldsymbol{w} \cdot \boldsymbol{c}) + k \cdot \mathbb{E}_{c_N \sim P_D} \left[\log \sigma \left(-\boldsymbol{w} \cdot \boldsymbol{c}_N \right) \right] \right) \tag{2.24}$$

This objective is trained over all observed pairs using stochastic gradient updates, finally making different words with similar contexts can have similar semantic vectors.

There are some special processing in the algorithm stated in the original paper [68]. In the beginning, the algorithm firstly traverses the whole corpus and randomly drops some words according to the probability distribution over word frequency. These dropped words are often some high-frequency stop words like *a, the, he, etc.*. In addition, the negative pairs are exactly sampled according to the 3/4 power of word frequency distribution.

Word2vec as implicit matrix factorization

The last part introduces a classical and common-used word2vec model, namely skip-gram with negative sampling (SGNS). Although it is clear that the SGNS's training objectives follow the distributional hypothesis - maximizing the dot product between frequently occurring word context pairs and minimizing the dot product between random word context pairs. However, little is known about why it is expected to produce good word representation. Levy et al. [47] found that it implicitly decomposes a word co-occurrence matrix, whose cell is the pointwise mutual information (PMI) of each word and context pair after being shifted by a global constant. This interesting and far-reaching corollary provides a reasonable explanation for why SGNS can produce high-quality word vectors and, at the same time, inspires follow-up studies about word embeddings such as GloVe [72].

Following previous notations, the Eq. 2.24 can be rewritten as:

$$\begin{aligned}\mathcal{L} &= \sum_{w\in V_W}\sum_{c\in V_C} \#(w,c)(\log\sigma(\boldsymbol{w}\cdot\boldsymbol{c})) + \sum_{w\in V_W}\sum_{c\in V_C} \#(w,c)\left(k\cdot\mathbb{E}_{c_N\sim P_D}\left[\log\sigma\left(-\boldsymbol{w}\cdot\boldsymbol{c}_N\right)\right]\right) \\ &= \sum_{w\in V_W}\sum_{c\in V_C} \#(w,c)(\log\sigma(\boldsymbol{w}\cdot\boldsymbol{c})) + \sum_{w\in V_W} \#(w)\left(k\cdot\mathbb{E}_{c_N\sim P_D}\left[\log\sigma\left(-\boldsymbol{w}\cdot\boldsymbol{c}_N\right)\right]\right)\end{aligned} \tag{2.25}$$

Then we extend the expectation term with the unigram distribution $P_D(c) = \frac{\#(c)}{|D|}$,

$$\begin{aligned}&\mathbb{E}_{c_N\sim P_D}\left[\log\sigma\left(-\boldsymbol{w}\cdot\boldsymbol{c}_N\right)\right] = \textstyle\sum_{c_N\in V_C}\frac{\#(c_N)}{|D|}\log\sigma\left(-\boldsymbol{w}\cdot\boldsymbol{c}_N\right) \\ &= \tfrac{\#(c)}{|D|}\log\sigma(-\boldsymbol{w}\cdot\boldsymbol{c}) + \textstyle\sum_{c_N\in V_C\setminus\{c\}}\frac{\#(c_N)}{|D|}\log\sigma\left(-\boldsymbol{w}\cdot\boldsymbol{c}_N\right)\end{aligned} \tag{2.26}$$

The last two equations indicate that the objective function ℓ for a specific (w,c) pair is:

$$\ell = \#(w,c)\log\sigma(\boldsymbol{w}\cdot\boldsymbol{c}) + k\cdot\#(w)\cdot\frac{\#(c)}{|D|}\log\sigma(-\boldsymbol{w}\cdot\boldsymbol{c}) \tag{2.27}$$

We then let $x = \boldsymbol{w}\cdot\boldsymbol{c}$ and calculate the partial derivative of the objective with respect to x, i.e.,

$$\frac{\partial\ell}{\partial x} = \#(w,c)\cdot\sigma(-x) - k\cdot\#(w)\cdot\frac{\#(c)}{|D|}\cdot\sigma(x) \tag{2.28}$$

We make the derivative equal to 0 to optimize the objective and can obtain:

$$e^{2x} - \left(\frac{\#(w,c)}{k\cdot\#(w)\cdot\frac{\#(c)}{|D|}} - 1\right)e^x - \frac{\#(w,c)}{k\cdot\#(w)\cdot\frac{\#(c)}{|D|}} = 0 \tag{2.29}$$

We then replace e^x with y, and this equation can be regarded as an univariate quadratic equation with respect to y. Its one solution is $y = -1$ which is incorrect obviously since $e^x > 0$, and another solution is:

$$y = \frac{\#(w,c)}{k \cdot \#(w) \cdot \frac{\#(c)}{|D|}} = \frac{\#(w,c) \cdot |D|}{\#w \cdot \#(c)} \cdot \frac{1}{k} \tag{2.30}$$

With y to e^x and x to $\boldsymbol{w} \cdot \boldsymbol{c}$, we can get

$$\boldsymbol{w} \cdot \boldsymbol{c} = \log\left(\frac{\#(w,c) \cdot |D|}{\#(w) \cdot \#(c)} \cdot \frac{1}{k}\right) = \log\left(\frac{\#(w,c) \cdot |D|}{\#(w) \cdot \#(c)}\right) - \log k \tag{2.31}$$

and $\log\left(\frac{\#(w,c)\cdot|D|}{\#(w)\cdot\#(c)}\right)$ is exactly the point-wise mutual information (PMI) of the pair (w, c). That is to say,

$$M_{ij}^{\text{SGNS}} = \boldsymbol{w}_i \cdot \boldsymbol{c}_j = PMI\left(w_i, c_j\right) - \log k \tag{2.32}$$

PMI is often used to measure the correlation between two objects, whose canonical form is defined as

$$\text{PMI}(x,y) = \log \frac{P(x,y)}{P(x)P(y)} \tag{2.33}$$

Normally, it can be used to represent the association between two words using the joint probability of their presence together (i.e., the two words are contextual to each other). In practice, each element of the PMI matrix can be estimated by the observations in the corpus.

Overall, although we can not interpret the trained word vectors $\boldsymbol{w}$ and context vectors $\boldsymbol{c}$ respectively, it is very interesting that their dot-products can be explained by the point-wise mutual information. Meanwhile, since PMI is a widely-recognized method to measure the association between words, it strongly confirms the superiority of the SGNS model from a theoretical point of view.

Finally, since SGNS is to train the word and context vectors to reach a situation in which their dot-products can equal the shifted PMI matrix, why not leverage it to deduce the word vectors we expected directly. Given the corpus and the pre-set size of the context window, we can achieve the word-context pairs (w, c) and construct the PMI matrix easily. The matrix is huge because its length and width are equal to vocabulary size, but we can use some dimensionality reductions to reduce its width and thus get word2vec embeddings in a learning-free fashion. However, in practice, SVD suffers from many problems compared

with learning methods, such as existing unobserved values [42] and being hard to weight different (w, c) pairs differently [88]. Therefore, a more reasonable way is to use the neural networks to directly learn the results of shifted PMI matrix, which is exactly the basic idea used in GloVe [72].

GloVe

As mentioned above, although levy et al. [47] found the *secret* of why skip-gram models perform well on creating word embeddings, it is still very hard to reproduce the result when they tried to use the theoretically optimal solution to derive word vectors. The gap lies at that the learning method performs more stably in estimating the dimension-reduced results. On the other hand, the word2vec models represented by skip-gram belong to local context window methods, which fail to take full advantage of global context information (i.e., the co-occurrence matrix of corpus). Instead, they optimize their objective contexts by contexts and thus often get stuck into the local optimum of partial contexts. Glove aims at combining the two methods, namely the global context matrix and the neural estimator.

The motivation and operation of GloVe are both straightforward, that is to use word vectors and context vectors to predict their co-occurrence probabilities. Similarly, let $\#(w, c)$ be the number of times that the word c appears in the context of word w, and $\#w = \sum_c \#(w, c)$ be the total number of contexts of word w. Thus, the conditional probability $P(c|w)$ can be estimated by $\frac{\#(w,c)}{\#w}$, i.e., the probability of word c exits in its context range give a word w. Subsequently, they train a neural regressions model to make $\boldsymbol{w} \cdot \boldsymbol{c} = \log \frac{\#(w,c)}{\#w}$ where $\boldsymbol{w}, \boldsymbol{c}$ are learnable word and context vectors. Since the term $\log \#w$ is independent of cm, the mapping function can be re-written as:

$$\boldsymbol{w} \cdot \boldsymbol{c} + b_w + b_c = \log \#(w, c) \tag{2.34}$$

Distributed sentence representation

Before, we mainly introduced word embeddings, but actually, more NLP tasks require sentence embeddings. There could be some straightforward ways from word representations to the sentence or document representations. Given the trained word embeddings, the most straightforward way is to average the word vectors to obtain the sentence vector after removing the stop words. Another effective approach is to calculate the weighted summation of word vectors, and the weights, for example, can be the normalized tf-idf

scores or other importance scores. These means are so-called inductive sentence embedding which only needs to learn representations of components and then reduce these vectors as the final vector we expected.

Another is the transductive sentence embedding, which indicates that each sentence or document is directly assigned with an independent vector. One typical operation is to take the inner words of a sentence as its contexts [45]. That is to say, let the context pair be (s, c) where s refers to a sentence or document and the context word c is exactly any word in the sentence. Correspondingly, the context c_N in the negative samples indicates any word out of the sentence or document. Its objective function and subsequent training process are similar to SGNS.

Compared with inductive learning, the disadvantages of transductive methods lie in the over-sized vocabulary and being hard to expand. Transductive embedding is more suitable for the small corpus where the size of document vocabulary (i.e., the number of documents) is limited. However, it would be hard to train the model when the size increases constantly. As to the inductive sentence embeddings, although there might be a larger vocabulary when the number of documents is not too many, its increase is finite as the number goes up because the total number of words is almost fixed in the real world. Another problem is that transductive methods fail to calculate embeddings for out-of-vocabulary sentences and often need to re-train the model on the new corpus. By contrast, it is easy to obtain embeddings for any new sentence using inductive models because its components have been represented. We only need to use or learn an aggregation function to reduce these component vectors as the final document vector.

ELMo and GPT

Since word embeddings can be used to form sentence embeddings naturally, it makes more sense to focus on the former. One main drawback of Word2vec and GloVe is that their word vectors are fixed even when the words are under different contexts. For instance, the word *play* should have different meanings in the two phrases *play basketball* and *play piano*, namely the word should have different vectors as its contexts change. However, the fact goes the opposite, and traditional neural word embeddings fail to catch the effects of immediate contexts to words completely. ELMo [74] took the first step to solve the problem systematically.

ELMo uses the bi-directional LSTM to represent words and their contexts. The two

directions ensure the model captures both left-to-right and right-to-left contexts. This operation seems to be used in many supervised NLP tasks, but note that ELMo leverages an unsupervised way, namely auto-regressive language modeling. Thus it can mine the contextual information sufficiently in a large corpus. When using the ELMo word embeddings, the two vectors trained in different directions are concatenated and fixed and then serve as the initial input embeddings of the models of downstream tasks. On the surface, ELMo just uses a common-used model to train a language model, but it indeed sends a very important message that unsupervised pre-training is useful. At the same time, another model, GPT [77] also sends a similar signal. GPT is very simple as it only uses a Transformer-decoder to train the language model, and the difference is it is built on countless parameters and trained on a huge corpus. Researchers found that the fine-tuned GPT model can be directed applied to NLP tasks by connecting an output layer. If we summarised the two ideas and factorized them into some essential factors, then ELMo should be pre-training and bi-directional modeling, and GPT be pre-training and Transformer. Both prove that pre-training is very important, and they provide a reasonable motivation to improve them, that is to combine all three factors.

BERT

To involve bi-direction and Transformer in the pre-training model, Transformer-encoder is a natural choice. However, it can not be trained as a traditional language model in an auto-regressive style. Hence, BERT [20] proposes a novel way to train a language model, called the non-autoregressive or masked language model. BERT is a milestone model which has achieved SOTA on multiple tasks. It marks that NLP has officially entered the era of pre-training, and its pre-training and fine-tuning have become a paradigm for performing NLP tasks.

BERT uses two pre-training objectives, namely masked language model (MLM) and next sentence prediction, to jointly train Transformer-encoder on a huge corpus. During training, it randomly masks 15% words in sentences and replaces them with special tokens. The MLM objective is exactly to use context representations to predict masked tokens. Every training sequence is composed of two partially-masked sentences. At 50% time, the second sentence is the next sentence of the first one in the document, but they are combined randomly for the rest of the time. BERT outputs not only context-aware word embeddings but also a special token to represent the whole sentence. Therefore, it can

be applied to word-level and sentence-level NLP classification tasks by concatenating an output layer above BERT. To better preserve the pre-training knowledge, the learning rate of BERT is often set up as a small value during fine-tuning.

2.3 Sequence-to-sequence Generation

2.3.1 Development of Sequence-to-sequence Models

The concept of sequence-to-sequence learning began to rise in 2014. Among them, the most widely-know model is the LSTM-based seq-to-seq model proposed by Sutskever et al. [89].

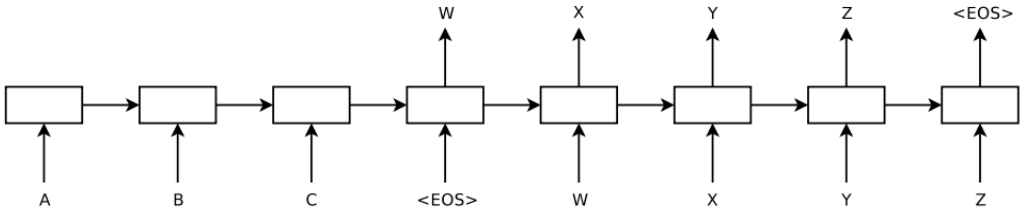


Figure 2.7: Architecture of seq-to-seq learning model.

Seq-to-seq model is to convert the input sequence to another different sequence. As shown in Figure 2.7, its structure should consist of an encoder and a decoder to encode inputs firstly and decode the encoding to the target sequences. Sutskever et al. [89] took Bi-LSTM as the encoder to represent the English sentence as a fixed vector, which then serves as the initial hidden states of the LSTM decoder to recover the encoded vector to a readable France sentence. In other words, the seq-to-seq model aims to maximize the probability of the target sequence $\boldsymbol{y}$ conditional on the source sentence $\boldsymbol{x}$, i.e., $\max p(\boldsymbol{y}|\boldsymbol{x})$. And the model is an auto-regressive generation model as the decoder forms the target sentences word by word. Therefore, the objective function can be further factorized as $\max \prod_t p(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t})$, where $\boldsymbol{y}_{<t}$ denotes the sentence having been generated before the step t, and y_t is the word generated at step t exactly. Such auto-regressive form fully considers the dependency between sequence components, which is basically in line with the physical truth. These dependency relationships are easily proved by some statistics of word frequencies. For example, some words have a higher probability of appearing

after some specific phrases like *hamburger* often after *KFC* and *eat*. This indicates that there is a stronger dependency for *hamburger* than many other words when the pre-phrase appears *KFC* and *eat*. Meanwhile, the left-to-right generation process is closer to the way humans read or express. Therefore, it is generally natural to choose the auto-regressive style to form sentences word by word instead of giving an additional assumption that all words are independent, even though the latter may have some speed advantages. Since the gold target sentences are known during training, it often inputs these gold words into the decoder to serve as generated ones before step t. It is exactly to align the last-token-deleted reference with the first-token-deleted reference and use the former to predict the latter. This training method is so-called teacher forcing training.

After LSTM-based seq-to-seq models, the next impressive structure is attention-auxiliary seq-to-seq model. One obvious disadvantage of LSTM seq-to-seq is that it always converts the whole source as a fixed vector, and all decoding steps are based on the same vector. Intuitively, different decoded words should depend on different parts of the source, namely a dynamic vector to represent the source step-wise. Inspired by this, the attentive model is exactly to assign weights to source words at different decoding steps dynamically. The decoder can generate words based on different source representations by calculating the weighted summation of source word vectors. This operation was first applied to the seq-to-seq translation model, which is very similar to human translation habits. Normally, instead of always attending to the whole source sentences, humans are more used to focusing on the word and its surroundings that need to be translated at that time and keep moving their attention constantly as the translation proceeds. To model the dynamic attention distribution, the attentive sequence-to-sequence model takes the generated sentence as the query to look up related parts in the source. In other words, it uses the neural network to estimate the similarity between the query representation and each source word representation. These similarity scores are then normalized as the attention probabilities to weight source words. The attentive mechanism has achieved great success and recognition, not only because it is derived from reasonable motivation and promotes a lot in the seq-to-seq tasks, but also because it provides partial interpretability for neural networks and can be generalized to many other tasks.

Overall, the attention mechanism mainly enhances expressive ability in the original attentive sequence-to-sequence model. However, it fails to solve inherent drawbacks of traditional seq-to-seq models such as the *UNK* problem. This central problem limits the piratical model effect in the early stage of seq-to-seq models. The reason that causes this

is that the training set is impossible to include all real-world words, especially some proper words like names, numbers, etc. Besides, the vocabulary size should not be too large so that some rare words will be actively expressed with *UNK* special token. In this case, the trained model may assign the highest probability to the *UNK* token at some decoding steps when the suitable words do not appear in the fixed vocabulary. After the attention model was proposed, researchers found that it can also be used to alleviate this problem except for enhancing representational ability. The pointer seq-to-seq networks [95, 85] are designed to copy the words in the source document as the generated words using an attention probability distribution. Instead of equipping with a fixed vocabulary, pointer networks have dynamic vocabulary whose extended parts are composed of words in the current source sentences. These extended words can not be allocated probabilities together with fixed-vocabulary words since the number of softmax units is fixed. Therefore, attention weights become the natural choice to estimate the probability distribution of these copied words. The pure pointer network can only copy words from the source sentence and can fail to produce novel words like humans. Therefore, at more time, the attention probability distribution should be combined with the original probability distribution. For the words belonging to the fixed vocabulary, their final probabilities depend on weighted summations of both probability distributions. As a branch of seq-to-seq models, Pointer networks always have a place when the target sequences are more composed of source words and do not require novelty. However, this network is difficult to be the mainstream seq-to-seq model because the main problem it solves (i.e., the issue of rare words) is covered by a highly efficient tokenizer system. Pointer network constructs a dynamic vocabulary, which is not conducive to training stability and efficiency. Thus, it can not continuously improve the generation quality even though it can avoid producing *UNK* token. By contrast, this problem can be better solved by only changing the tokenizer mechanism. Traditional tokenizer puts the complete words into the vocabulary set, and the vocabulary can easily break through millions if the corpus is large enough. This is unacceptable because it places a significant burden on computing and memory. Revisiting the inductive idea, words are further split into smaller components. For example, many words share the same prefix or suffix. Even if a rare word does not appear in the training corpus, its embedding is still accessible because its sub-words are available in the preset vocabulary. Such tokenizer, the so-called subword tokenizer [86, 5], can efficiently solve rare words and significantly reduce vocabulary size simultaneously. The pointer family is not the mainstream seq-to-seq model, and almost all pre-training seq-to-seq models do not use this structure. However,

it is undeniable that the pointer mechanism is a very effective technique and has been applied to many tasks such as edge pointer distribution in graph generation [112, 8].

Sequence-to-sequence models have been playing an important role in many generative tasks, especially translation and summarization. Besides, it has promoted the development of many new fields like image caption [96] which model structure is exactly derived from seq-to-seq but uses a convolution network as the encoder. Compared with transitional approaches like statistical translation or extractive summarization, such a neural generation model is generally considered more creative and produces more coherent sentences. At the same time, attentive models have been proved to have strong representation ability in multiple tasks such as classifications [106]. However, with available paired samples more and more, the researcher found that the existing attentive LSTM-based seq-to-seq models have two bottlenecks that are difficult to break through, making the marginal benefit of increasing data lower and lower. The first bottleneck is that even with teacher forcing training, LSTM still can not predict the whole target sequence at once, making efficient training impossible. Another is that the benefit of increasing the number of layers is insignificant compared with the additional burden it brings to training and inference. The emergence of Transformer [91] breaks through these two bottlenecks by replacing LSTM cells with the attention module completely.

Transformer, as introduced before, allows more parallelization during training. In short, it can output whole target sequences at once during teacher force training. Even if Transformer is superimposed continuously, its training speed is still guaranteed. A deep structure indicates more parameters, which means it has the ability to represent more various information. Meanwhile, it can process the global information at each layer. Therefore, it is reasonable that Transformer has a stronger representative ability than RNNs or some convolution networks, which can only capture local information to form a global view gradually. Without Transformer, NLP is difficult to really enter the era of pre-training. This is because we need pre-training models that can process a large amount of data efficiently and reliably. Two inherent bottlenecks of RNN make them destined to be the infrastructure of the pre-training model.

In the era of pre-training, model structures are dominated by Transformer and its variants. At most times, the vanilla Transformer encoder and decoder are good enough to deal with all situations. For instance, the vanilla Transformer-encoder often serves as the non-autoregressive language model structure, and vanilla Transformer-decoder or encoder-decoder is the first choice of generative pre-training models. Generally, we need

some Transformer variants to reduce the computational costs when the model is very large or the input sequences are very long. Common-used variants are sparse Transformer [13], restricted Transformer and linear Transformer. Compared with the changes in the model architecture, more changes come from pre-training objectives. The next part will give more details about the pre-training objectives of different seq-to-seq pre-training models.

2.3.2 Pre-trained Sequence-to-sequence Models

This part will introduce three notable Seq-to-seq Pre-training Models (PTMs), namely BART [48], T5 [79] and PEGASUS [109]. The three PTMs all use standard Transformer encoder-decoder and teacher forcing training. The differences lie in pre-training objectives and applied tasks.

BART

Before introducing BART, we firstly review BERT [20] and GPT [77] from the perspective o of generative models. BERT is widely known as non-autoregressive, which means it is not easily used for a generation because masked tokens are predicted independently. On the other hand, the model structure of GPT is the pure auto-regressive language model so that it can be applied to generative tasks straightforwardly. However, its obvious disadvantage is that its pre-training objective can not learn bi-directional interactions because words in GPT are only conditional on leftward contexts. Therefore, BART can be regarded as the generative model pre-trained with bi-directional contexts. Its pre-training schema has been shown in Figure 2.8 (intercepted from the original paper). The encoder input is the noised sentence, and the general objective is to recover the sentence. We can see its training objective fuses the features of both BERT and GPT. Compared with BERT, it adds a decoder to train sequences in the auto-regressive style. Thus it can serve as a text generator naturally. Compared with GPT, when predicting the missing tokens such as C, the dependencies include leftward contexts in the decoder inputs and the unmasked rightward context in the encoder inputs. It should be mentioned that masking tokens is not the sole way to noise sentences. There are totally five transformations for nosing inputs (see Figure 2.9), and they can be compose together.

As to the relevant fields, BART can not only do the same classification tasks as Bert but also seq-to-seq generation tasks, including summarization, generative QA, and dialogue. Besides, it can also be applied to the translation by replacing the pre-trained encoder

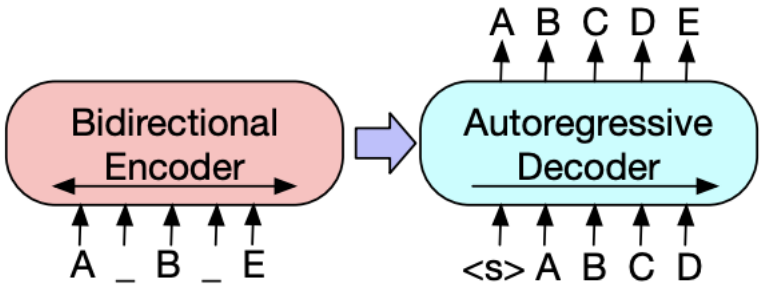


Figure 2.8: BART pre-training schema.

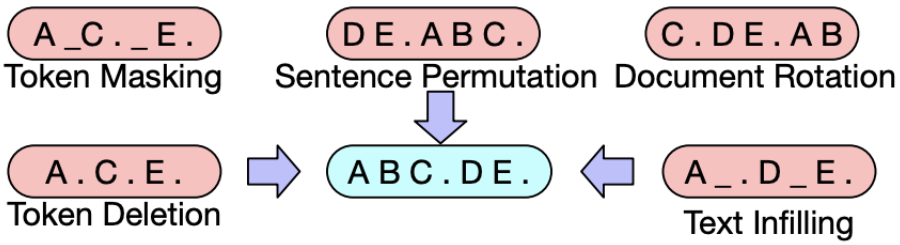


Figure 2.9: Five transformations for nosing BART inputs.

embedding layer with a new randomly initialized encoder. Generally, the final word representations of BART are the outputs of the decoder, and the final sentence representations are embeddings of the special ending token. The whole decoder inputs are known for the classification tasks, namely the input sentences without the last token. The initial decoder input is only the special start token for generation tasks. Newly generated words are then concatenated to the decoder input to promote the generation continuously.

T5

Before we introduced BART, we can see it can be applied to almost all NLP tasks regardless of classifications or generations. T5 (Text-to-Text Transfer Transformer) can do the same, but their ways are essentially different. BART maps different tasks into different distributions. BART needs to use different output layers (e.g., the number of softmax units are often different), loss functions, and hyper-parameters to deal with different downstream tasks. By contrast, T5 uses the same system to deal with all tasks, even including

regression tasks. This is because T5 transforms all tasks as the mode of text-to-text tasks. For example, the sentiment classification can be equivalent to using the text-to-text model to generate the word *good* or *bad*. Since T5 uses the same set of parameters to process multi-tasks, it needs a trigger to tell the model which task it needs to conduct. Figure 2.10 shows how T5 designs these triggers and puts them into the inputs. Notice that the figure is copied from the original paper.

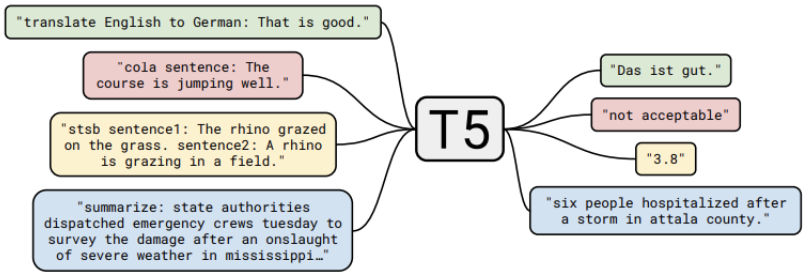


Figure 2.10: Examples of T5 applying to different tasks.

Overall, T5 is a combination of pre-training and multi-task learning. During training, T5 lets the decoder predict the masked tokens of encoder inputs, which is unsupervised similar to other PTMs. The difference is that T5 also adopts multi-task supervised learning in the pre-training stage. Moving to the fine-tuning stage, T5 also uses multi-task mode to concatenate all downstream tasks into a single task to conduct fine-tuning.

PEGASUS

Unlike the previous two seq-to-seq PTMs, PEGASUS [109] only focuses on one downstream task, namely text summarization. However, it provides a simple pre-training canonical form for almost all source-based text generation. PEGASUS firstly leverages an unsupervised way to extract important sentences from the source documents. Then it pre-trains the transformer encoder-decoder by using the source document to predict its important components. The key point of PEGASUS lies in de-noising, and this concept is very important in many tasks like image captioning or paraphrasing, etc. Generally, most source-based text generations do not need the target sentences to cover all source information in addition to the translation. Besides, many texts crawled from the Internet contain a lot of noise, so this pre-training objective for de-noising is very practical.

2.4 Decoding Strategies

Unlike classification tasks, there are many places worth exploring in the inference stage of generation tasks. We refer to these techniques to improve generation quality at inference time as decoding strategies. According to different generation objectives, we can roughly divide the decoding (search) strategy into two categories, namely sampling search and deterministic search. The former is a decoding method with randomness. It is suitable for generating creative text such as story generation, where there may be multiple optimums as long as the story is fluent and logical. By contrast, the latter is more common in source-based text generation like sequence-to-sequence generations, where the content described by these generated sentences can not be separated from the source. Therefore, sampling search is not favored here because its randomness may lead to content bias. In this case, we prefer to reduce randomness to generate sentences that are less likely to make mistakes. Overall, the deterministic search will return the sentence with the highest probability, i.e., to find the most likely sentence to be a pair with the given source document.

2.4.1 Sampling Search

Vanilla sampling search

At each decoding step, the decoder will output a probability distribution $p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{y}_{<t})$ whose length is equal to the vocabulary size. The vanilla sampling is to randomly choose a word based on the probability distribution and then treat the selected word as the next-step input to predict the next probability distribution.

Top-k sampling search

It is possible that vanilla sampling selects a corrupted word with low probability because it is a completely random process. Since the subsequent generation depends on the corrupted word, the errors will accumulate constantly. In general, any failed choice may make the whole generated sentence unsatisfactory. Therefore, although the probability of choosing corrupted words every time is very low, the total probability of not extracting corrupted words in all steps is not low.

To solve the problem mentioned above, one straightforward way is to firstly select top k candidates from vocabulary according to the probability distribution. The candidates'

probabilities are then re-scaled to a new distribution, and the final successor is sampled from these candidates based on the new distribution.

Top-p sampling search

The defect of top-k sampling is that the number of candidates is fixed for any probability distributions. This often leads to some corrupted words being still selected as candidates, especially when the probability distribution is mainly concentrated on the head words. Given a distribution $p(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t})$, we define its top-p vocabulary as $\mathcal{V}^{(p)} \subset \mathcal{V}$ which should meet the following condition:

$$\sum_{y_t \in \mathcal{V}^{(p)}} p(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}) \geq p \tag{2.35}$$

. In other words, p is a threshold in the probability distribution, and words with probabilities larger than the threshold are candidates from which the final word is sampled, while other words have no chance to be selected. In this case, the number of candidates is dynamic rather than fixed in top-k samplings. For example, if the majority of the probability mass is clustered in the head words, we need a smaller candidate set to exclude more tail words. Top-p samplings can do this easily by assigning a higher value to p.

2.4.2 Deterministic Search

Greedy search

Greedy search always chooses the word with the highest probability at each decoding step and stops when the selected word is the preset end token. Theoretically, when the global optimal solution is factorized to each step, it should be the local optimal solution, namely the word with the highest probability. This is actually in line with the way of teacher force training. However, although it enjoys lower $\mathcal{O}(|\mathcal{V}|)$ complexity, it often performs worse among all deterministic decoding strategies because it suffers from exposure bias most. The exposure bias [100, 80, 110] reveals the gap between teacher force training and decoding. Especially, teacher forcing training only needs to ensure local optimality because it is based on reference tokens, which means the compacted local optimal solutions are the global optimality exactly. By contrast, the local optimality in the greedy search is dependent on the previously generated words. Obviously, these local optimal solutions are

unreliable, and these errors will continue to accumulate. As long as there is a deviation in any step, it will eventually deviate from the global optimal solution.

Viterbi algorithm

Since the final objective of the greedy search is to find the sentence with the highest probability, the straightforward way is to construct all possible sentences whose number is the square of vocabulary size. Then the sentence with the highest probability will be the successor. The biggest problem is that this method has extremely high complexity $\mathcal{O}(\mathcal{N} \times |\mathcal{V}| \times |\mathcal{V}|)$ and is hard to apply in practice text generation.

Beam search

Beam search is an eclectic approach between greedy and Viterbi. Simplified speaking, it only preserves $\mathcal{K}$ sentences with the highest probabilities at each generation step. Compared with Viterbi, its complexity is significantly reduced to $\mathcal{O}(\mathcal{N} \times |\mathcal{K}| \times |\mathcal{K}|)$. At the same, it provides a buffer for the selections of local optimums. With the beam size increased to the vocabulary, beam search is exactly the Viterbi algorithm. However, studies found that the generation quality is degraded instead as the beam size goes up [16, 70, 67]. There is no standard explanation for this phenomenon. Still, it can be determined that there exists an inherent bias in teacher forcing training such that the global optimum learned by it may not be completely reliable. As a result of this phenomenon, we usually use smaller beam sizes to conduct beam search to guarantee both efficiency and quality. We will give more details about the procedure of beam search in Chapter 5.

Chapter 3

News Network Embedding based on Local and Global Context

3.1 Introduction

News, carrying a large amount of information, can often guide public opinions, affect people's behavior and drive the evolution of social events. In the era of information, news text is considered to be a part of big data that is continuously updated. In order to facilitate the performance of downstream NLP tasks, it is rather essential to find an efficient way to represent news as continuous vectors that contain the collective information of its inherited features and the inter-textual knowledge between different news.

The most straightforward approach to embed news is to directly treat it as textual data and apply text/sentence embedding models. To represent texts/sentences as vectors, one of the classic techniques is to perform numerical operations on the word vectors [45], which is recognized as a simple but powerful baseline [18]. SDAE [94] uses an auto-encoder to compress texts into a low-dimensional vector. Paragraph Vector [45] learns the distributed representation of sentences and documents by predicting their contexts, i.e., words in the sentence/document. Additionally, Skip-Thought [50], FastSent [34] and Quick-Though [63] learn distributed sentence-level representations through the coherence between sentences. BERT [20] proposes a powerful pre-trained sentence encoder which is trained on unsupervised datasets based on the masked language model and next sentence prediction. The major limitation of embedding approaches discussed above is that they produce representations and features that solely describe the local contextual information

at the document level. In the case of news representation, the connection between different news events (i.e., global contextual information) is also considered as crucial information that should be incorporated into the news embedding, as well as other labeled features such as the topic category and the polarity of the news.

With the argument that news stories describe events, news event embedding models are developed to offer an alternative solution for the vector representation of news. It is suggested to extract event tuples $E = (Actor, Predicate, Object)$ from headlines and learn vector representations by scoring the correct tuples higher than the corrupted ones [21, 22]. Event2vec [87] is constructed on a classified news event database from Wikipedia[1], where an event network is created by considering events, entities, event types and years as nodes of different types. The Event2vec model thus produces distributed vector representations of news events based on the network embedding mechanisms, which lacks the flexibility of generating vectors for nodes outside the trained network. In addition, the weights of some edges in the event network are determined by objective assumptions since types and years are not comparable.

In this study, the News2vec model is proposed to learn distributed representations of news, with the embedded vectors containing not only the semantic and labeled information, but also both local contexts (between the news event and its components) and global contexts (between different news events). We start by building the news network that connects the news nodes with their corresponding event element nodes in order to create the aggregated news context that describes both the semantic information and the background linkages between different news events (See Figure 3.1 for a simplified example of the news network). According to Figure 3.1, the two pieces of news $Alibaba\ reinvested\ Suning\ after\ three\ years$ and $Suning's\ second{-}half\ profit\ rose\ by\ 5\%$ are connected by sharing the same contextual element $Suning$. The verbal keywords $reinvest$ and $rise$ of the two different news are thus connected through their common node $Suning$ and become the latent contextual element of the other news. Based on the constructed news network, the biased random walk approach [29] is adopted to generate adequate number of connected node sequences which are later used in the network embedding. Inspired by the Subword model [4], we further represent each news node as a bag of news features including semantic features, categorical features, time features, etc.. The vector representation of the news feature is thus obtained based on the skip-gram

[1] `https://en.wikipedia.org/wiki/Portal:Currentevents`

architecture [68], and the news vectors are computed as the sum of its feature embeddings.

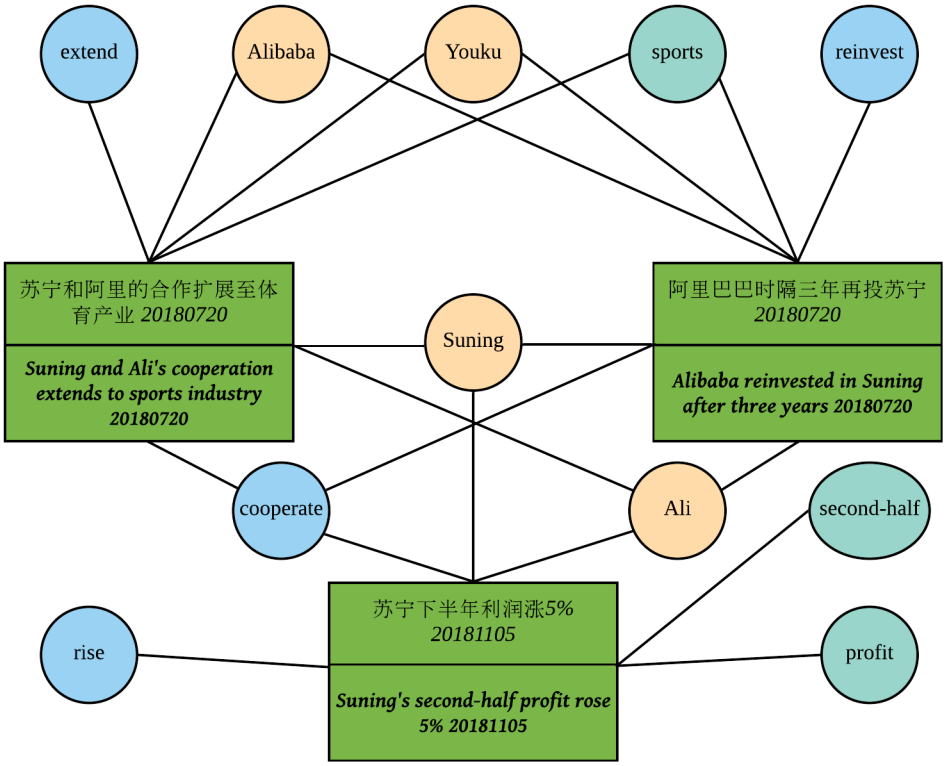


Figure 3.1: Simplified sample of the news network

The advantages of the proposed News2vec model are twofold. First, the News2vec embeddings capture both the semantic features and the potential connections between news by constructing the news network. Second, the Subnode model offers an easy solution to create embeddings for unseen (news) nodes outside the trained network, at the same time enriches the node vector with its node attributes (news features). As the experiments suggest, the News2vec model achieves state-of-the-art performance on the news-related downstream tasks, namely the stock movement prediction and news recommendation.

3.2 News2vec Model

In this section, the formation of the News2vec model is discussed in terms of the news network construction and news embedding based on the Subnode model. According to Figure 3.2, news elements including entities, actions and nouns, are first extracted from the news titles and texts. Based on these elements and their term frequency–inverse document frequency (tf-idf), the news network is built and sequences of nodes are sampled by the biased random walk. News nodes in these sequences are then represented as bags of extracted news features. The associated vector is thus assigned to each feature by the Subnode model.

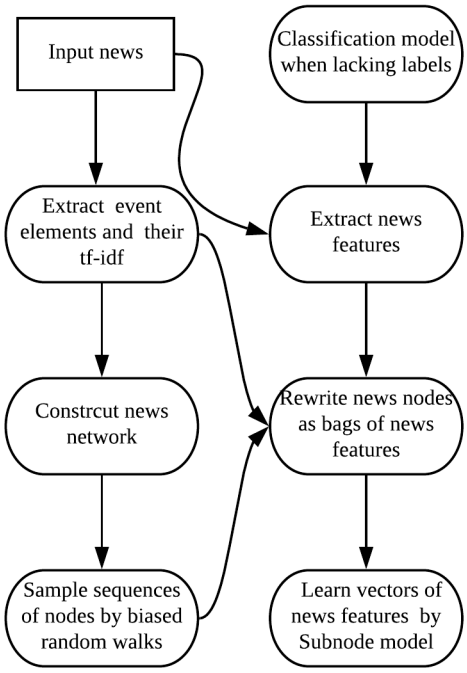


Figure 3.2: Flow chart of creating news embeddings with News2vec

3.2.1 News Network

The News2vec model creates news embeddings based on the news network that connects news with their elements. With the argument that a piece of news often describes one or

several events which could be represented as a collection of elements, this study extracts news elements as entities, actions and nouns from news titles and texts, with respect to their tf-idf scores. Specifically, the tf-idf is a well recognized tool to measure the importance of a word/phrase inside the document and to extract the representative elements at the document level. The tf-idf score is formulated as:

$$tfidf_{k,e} = \frac{n_{k,e}}{\sum_w n_{w,e}} \times \log \frac{|D|}{|e : k \in d_e|}, \tag{3.1}$$

where $n_{k,e}$ denotes the number of the news element k in the news story e, whereas its denominator is the total number of words/phrases in the news document. $|D|$ is the total number of documents, the denominator represents the number of stories containing the element k. In a nutshell, the tf-idf assigns a higher score to the element which appears more frequently in the news story than in others.

After determining the elements of news, the news network is hence constructed by connecting the news nodes $e \in V_e$ of the news title and published time with its element nodes $k \in V_k$. The weights of connective edges (e, k) are computed based on the importance of the element to the news, which is determined by their tf-idf scores. The weight of an edge (e, k) is computed as:

$$W_{e,k} = \frac{tfidf_{k,e}(title)}{Z_1} + \frac{tfidf_{k,e}(content)}{Z_2}, \tag{3.2}$$

where Z is a normalization constant.

3.2.2 Network Embedding with Subnode Information

Network embedding

The New2vec network embedding is based on the Node2vec [29] model, with the objective to learn a latent feature vector $F(v)$ for each node $v \in V$ that maximizes the probability of predicting the node $v's$ network neighborhood $N(v)$. The objective function is written as:

$$\max \sum_{v \in V} \log p(N(v)|F(v)). \tag{3.3}$$

Given the feature vector of node v, it is assumed that the prediction probabilities of its neighborhoods are independent from each other, which leads to:

$$p\left(N(v)|F(v)\right) = \prod_{u \in N(v)} p(u|F(v)). \tag{3.4}$$

To solve the objective function and obtain the optimized node representations, the skip-gram architecture [68] is adopted on the basis of network neighboring sequences sampled by the biased random walk. To reduce the computational cost, negative sampling [45] is used to replace the softmax classifier by multiple logistic binary classifiers. The optimizer is the Stochastic Gradient Descent (SGD).

Different from the uniform random walk of DeepWalk [73], News2vec uses the Breadth-first Search (BFS) and the Depth-first Search (DFS) as its search strategies. Suppose that the walk just transitioned from t to v and is now walking to its next step node x, BFS determines the next step with a greater chance of revisiting t, i.e., $d_{tx} = 0$, whilst DFS drives the walk to go deeper and further away from the node t, i.e., $d_{tx} = 2$. In line with the Node2vec model, News2vec defines a search bias α to control the search preference.

$$\alpha_{vx} = \begin{cases} \frac{1}{p}, & if\ d_{tx} = 0 \\ \frac{1}{q}, & if\ d_{tx} = 2 \end{cases} \tag{3.5}$$

Next, the transition probability of edge (v, x) is defined as the product of the weight of the edge and the bias, i.e., $\frac{\alpha_{vx} \times w_{vx}}{Z}$ where Z is a normalization constant. Explicitly, if $p < q$, the walk is width-first which indicates that there is a greater chance to revisit the original node and sample the nodes around it, that is, local contexts and semantically similar news. If $p > q$, the walk is depth-first and tends to go to nodes that are not passed before, which leads to the exploration of news with latent relations (i.e., global contexts). During the random walk, each node is used as a starting vertex for a fixed-length walk to produce a group of sequences. Based on the skip-gram model, we can train vector representations of nodes on these sequences.

Subnode model

Classic skip-gram model assigns a distinct vector to each word, neglecting the morphology information of words. With the purpose of addressing this problem, the Subword [4] model represents each word using a bag of character $n-$grams, and the word vector is thus the

summation of its character $n-$grams. The major advantage of the Subword model is that it allows the computation of word vectors that are not in the trained corpus by using their character $n-$gram vectors. Inspired by the Subword model, we introduce the Subnode model to solve network embedding problems by offering a solution to create inferential vectors of unseen news nodes outside the trained network, at the same time incorporate node attributes to the node vectors.

The biased random walk produces sequences of news and event element nodes from the news network. The proposed Subnode model expresses each of the news node as a bag of subnode information of its associated features including semantic features such as event elements with high tf-idf scores, text structure features such as words count and paragraphs count, and side information such as published date, news type and emotion. As a result, each news vertex $e \in V_e$ is represented as:

$$G =< entity_A, entity_B, action, \cdots ,$$

$$words\ count, paragraph\ count,$$

$$month, day, week, type, sentiment, \cdots >$$

where word count, paragraph count and sentiment are ordinal data rather than real values. Further, the Subnode model represents a news node as the sum of its features, instead of a distinct vector obtained from the other network embedding models. Therefore, the objective function of news network embedding is expressed as:

$$max \sum_{e \in V_e} \log p(N(e)| \sum_{g \in G_e} f(g)) \tag{3.6}$$

where $f(g)$ denotes the vector representation of a news feature. In addition, we remove nodes with only one edge to reduce the sparsity of the network. As it is mentioned in the previous section, the Subnode vector representations of news features are learned based on the skip-gram architecture by optimizing the updated objective function using SGD with negative sampling [45]. With an adequately large news network that contains a wide range of features, the Subnode mechanism of News2vec allows the vector representation of an unseen news node directly computed by extracting news features and summing up their corresponding vectors according to the $feature\ to\ vector$ dictionary. See Github[2] for the

[2] `https://github.com/yema2018/News2vec`

codes and examples of News2vec.

Algorithm 1: Subnode

Input: S, the node sequence $S = [k^1, e^1, k^2, e^2, \cdots]$ generated by random walks
Output: F, the embedding matrix with respect to news feature g

Rewrite the sequence as $S^{'} = [(g_1, g_2, \cdots), e^1, (g_3, g_4, \cdots), e^2, \cdots]$;
Initialize F;
for s in S', $s \neq e$ **do**
  for u in $Context(s)$ **do**
    $NegativeSampling$ u' from $S'/\{s\}$;
    $J(F) = -\log p(u|\sum_{g \in G_s} F(g))$;
    $F := F - \eta * \frac{tialJ}{tialF}$
  end
end
Return F

3.3 Experiments

Four experiments are implemented and explored in this section. In particular, we first visualize the News2vec embeddings in terms of the news vectors and feature vectors with the dimension reduction plot and the correlation heat maps. Two downstream experiments, i.e. the stock movement prediction and news recommendation, are then conducted to examine the News2vec model in comparison with other established text/sentence embedding models. Overall, News2vec achieves state-of-the-art performances in these news-related tasks which demonstrates its validity of addressing potential relevance of news via the network, as well as the integrated consideration of the news features.

3.3.1 Visualization of News Vectors

In this section, the THUCTC[3] (THU Chinese Text Classification) news data set is used to train the vectors of news features, including the semantic features and four additional features, namely *type*, *words count*, *paragraph count* and *sentiment*. We determine the sentiment by counting positive words over negative words. There are 14 types of news in the news corpus and we randomly take 6,000 pieces of news from each type as the training

[3] `http://thuctc.thunlp.org`

set. In the test set, we additionally extract 1,000 pieces of news from each type. Based on these 84,000 pieces of news in the training set, we learn a vector representation of 128 dimensions for each news feature. The news vectors in the test set are then computed, whereas their dimension reduction plots are produced using the t-Distributed Stochastic Neighbor Embedding (t-SNE).

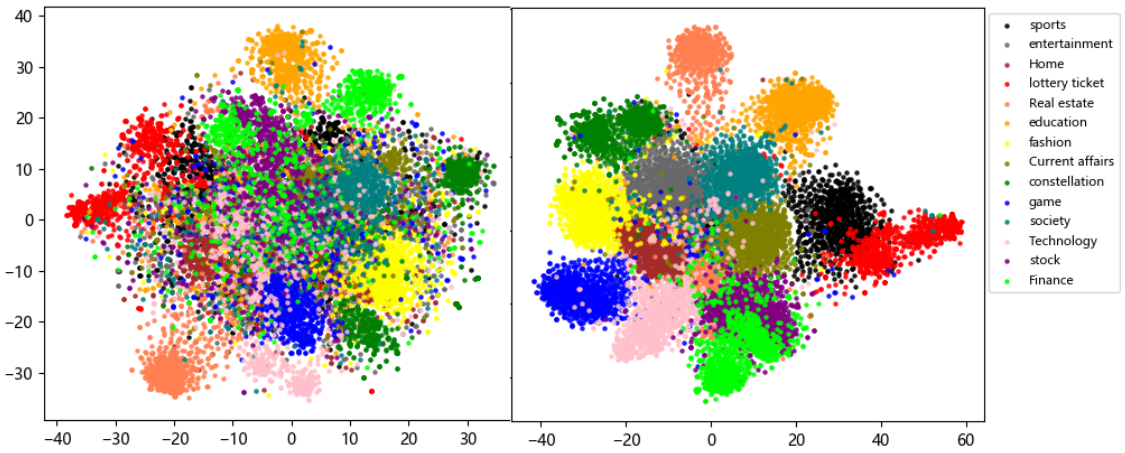


Figure 3.3: Dimension reduction plots of test set news without and with the type feature

Figure 3.3 shows the dimension reduction results of news vectors in the test set. According to the right panel, a high level of clustering is observed, indicating that News2vec allows the embedding of type information into news vectors after incorporating the type feature. Moreover, the distance between two clusters is in-line with the relevancy between the clustered topics in reality. In particular, neighboring relationships are exhibited between similar topics such as sports & lottery ticket, current affairs & society, stock & finance, and fashion & entertainment, whilst the distance between less relevant clusters, such as constellation & finance, game & education, is relatively larger. Nevertheless, the left panel of Figure 3.3 shows that news embeddings significantly lose classification information once vectors of the type feature are removed from the news vector, which shows that News2vec enriches the news vector representation by the news network and the Subnode information.

3.3.2 Correlations of News Features

In this section, we further investigate the vector correlation between three news features, namely the *type*, the *words count* and the *sentiment*. The correlation coefficients (i.e.,

normalized cosine similarity) heat maps for each feature are displayed in Figure 3.4.

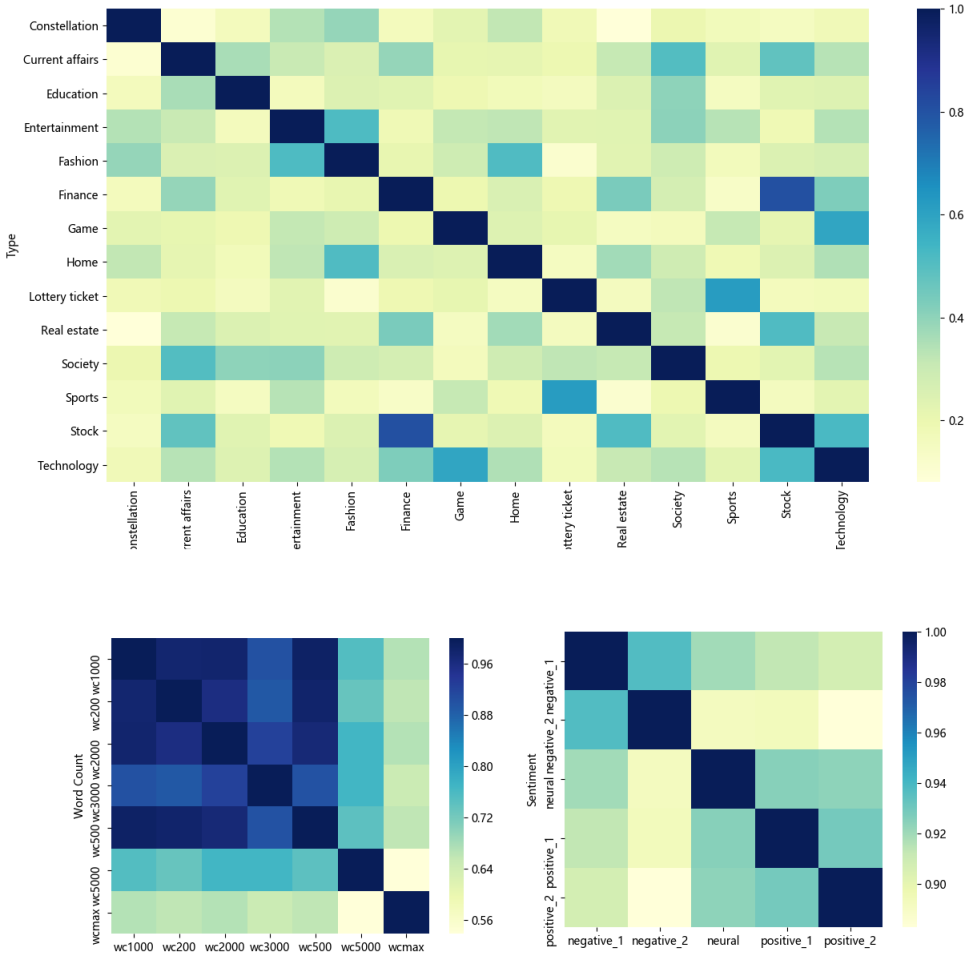


Figure 3.4: Correlations of news features (Upper panel: news type. Lower left: word count. Lower right: sentiment.

The correlation heat map of news type features (upper panel) supports the conclusion in Section 3.1. It is observed that vector pairs of close topics such as *type* : *Stock* and *type* : *Finance* are highly correlated, indicating stock news and financial news share similar contextual elements. Other highly correlated pairs include sports and lottery ticket news, game and technology news, fashion and entertainment news, etc., whilst less relevant topics show low correlations, such as real estate and constellation, sports and real estate. According to the bottom left panel of Figure 3.4, a significant number of dark blocks are

observed in the area from $wc200$ to $wc3000$ ($wc200$ means the news is less than 200 words and $wc2000$ means the news is between 1000 and 2000 words), and there is a clear reduction of correlation once the news exceeds 3,000 words. As it is indicated by the word count correlation heat map, news that has similar numbers of words is more likely to have similar content. The correlation reduction from $wc3000$ to $wc5000$ suggests that there is a group of news with specific content, that is often discussed in long articles. Meanwhile, the right bottom of Figure 3.4 shows that news with close sentiment levels tends to have higher correlations. As the sentiment moves from more positive to more negative, the correlation decreases. Overall, the correlation heat maps show that the News2vec feature vectors are reliable in terms of expressing Subnode information.

3.3.3 Stock Movement Prediction

Experimental settings

We use financial news (2009.10.19 to 2016.10.31) from Sohu [4] to predict the daily movement of Shanghai securities composite index. The news feature vectors are trained based on news from 2009.10.19 to 2015.12.31 and news vectors are computed by summing up these feature vectors. There are five additional news features, namely *month*, *week*, *sentiment*, *words count* and *day*.

The length of the walk is fixed at 100, the context size is 10, return hyper-parameter (p in Equation (5)) and input hyper-parameter (q in Equation (5)) are both set to 1.

Predictive model

The predictive model is the long short term memory (LSTM) network [35] with self-attention mechanism [106]. As shown in 3.5, the LSTM model has T time steps (i.e., T days) in total. At each time step, the input is the weighted average of news vectors in that day. Weights are initialized at the beginning and updated by the gradient descent. Based on the attention model, news that is closely related to the stock price movement is assigned with higher weights.

In this paper, we use the previous 20 days' ($T = 20$) news to predict the next day's stock index movement. The output layer is a Softmax classfier that indicates whether the index goes UP (rise percent $> 0.33\%$), DOWN (rise percent $< -0.29\%$) or PRESERVE (-0.29%

[4] https://www.jianshu.com/p/370d3e67a18f

$<$ rise percent $<$ 0.33%). We employ a rolling window with $size = 20$ and $strider = 1$ to augment the sample. News before 2016 is used as the training set, whilst news after 2016 is used as the test-validation set.

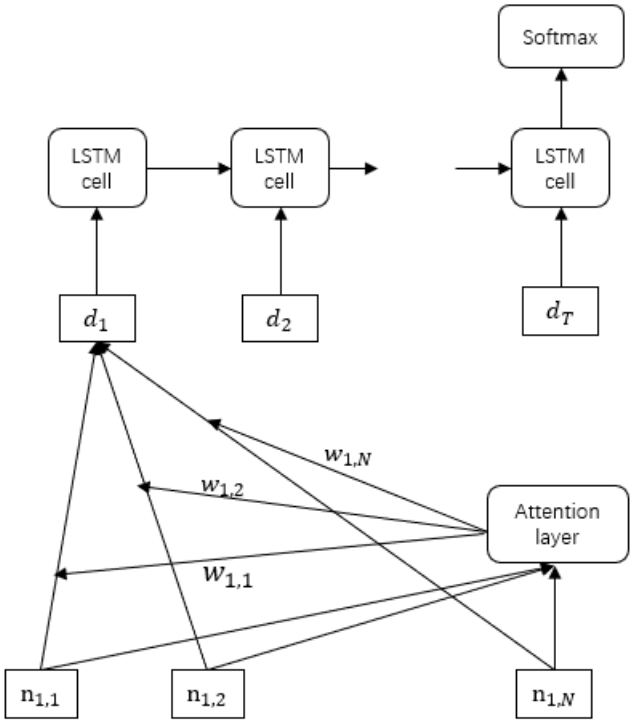


Figure 3.5: LSTM model with attention mechanism

Results

As for the baseline models, we include several established embedding approaches to compute news vectors in the experiment of the stock movement prediction:

Average BOW: News embedding is represented as the average of word vectors in the news titles and bodies [37]. Word embedding is obtained by training the skip-gram Word2vec model [68].

Doc2vec: Paragraph Vector [45]: News texts are represented as dense vectors by the Paragraph Vector model[1].

Event embedding [21, 22]: Event tuples are extracted from headlines and represented as vector representations by scoring the correct tuples higher than the corrupted tuples.

In addition, we apply a sentence-level encoder BERT to encode news headlines as another baseline model:

BERT [20]: Average pooling is used to get a fixed representation of a headline based on the Chinese pre-trained model[5].

The proposed News2vec model is implemented separately with and without the five additional features (i.e. *month*, *week*, *sentiment*, *words count* and *day*):

News2vec-: News vectors are the summation of solely the semantic features (i.e., event elements).

News2vec+: News vectors are the summation of all news features including the semantic and the five additional features.

Table 3.1: Acc and MCC Results of stock movements in the test set

	Acc	**MCC**
Average BOW($d = 128$)	41.91%	0.1137
Doc2vec($d = 128$)	43.90%	0.1484
Event embedding($d = 128$)	43.76%	0.1439
BERT($d = 764$)	45.83%	0.1442
News2vec-($d = 128$)	44.68%	0.1664
News2vec+($d = 128$)	**46.24%**	**0.1936**

In Table 3.1, test results are presented according to two measurements, namely the accuracy (Acc) and the Matthews Correlation Coefficient (MCC). We find the News2vec model outperforms BERT and all the other baseline models both in terms of the Acc and MCC, with a lower dimension of 128 (The dimension of BERT embeddings is 764). As the MCC suggests, Doc2vec, Event embedding and BERT produce stock movement predictions of similar qualities as the news vectors generated by the three approaches uniformly limit to the semantic information of news articles. As for news-driven stock price prediction, it is more reasonable to take global contexts into account, such as having similar background story or leading to the same event, and reflect the connection by embed similar values in certain dimensions of the news vectors. In the case of the New2vec+ model, we find that extra features contribute to the improvement of the MCC, indicating the New2vec representations are able to capture the latent connections between news events. After considering all news features, the results are improved by 0.028, showing their positive

[5] `https://github.com/google-research/bert#pre-trained-models`

effect on stock movement predictions. It is worth-mentioning that the New2vec model is performed in an unsupervised manner as all features are extracted from common news text without any artificial label or trained classifier. We believe that there is a large chance to further improve the result once artificial labels such as the polarity are incorporated.

3.3.4 News Recommendation

Experiment settings

In this section, a news recommendation task is implemented based on the data of news browsing records (2014.2 to 2014.3) using the news embedding models [6]. The obtained data set contains 16214 news browsing sequences in total, which is split into halfs for testing and fine-tuning, respectively. The overall objective is to recommend news based on the content by computing the cosine similarities of the news vectors.

News recommendations are made for each news by selecting the top ten news with the highest vector similarities. The recommendation is considered to be successful as long as the next browse in the sequence belongs to the ten news articles selected. We compute the success rate for each sequence. The average success rate of all sequences is the final evaluation result for the embedding model.

Since the news titles are absent in the data set, event elements are only extracted from news bodies. As a result, we only use the text-level embedding models (i.e., Paragraph Vector and Average BOW) as the baselines to compare with the News2vec model. We use the trained news feature vectors in the previous section as the initial inputs of the new news network (Sohu+rec). Moreover, half of the news sequences are employed to fine-tune these vectors (Sohu+rec+seq). Due to the lack of information, only *sentiment* and *word counts* are included as the extra news features together with the existing semantic features.

Evaluation of news recommendation

From Table 3.2, it is observed that the success rate improves significantly after fine-tuning the news vectors with respect to the news browsing sequences. In comparison with Paragraph Vector, News2vec (Sohu+rec) demonstrates no significant advantage without sequence fine-tuning (Sohu+rec+seq), indicating potential connections between news have limited effect on this task. Nonetheless, News2vec achieves state-of-art performance after

[6] `https://github.com/YLonely/web-data-mining`

Table 3.2: Evaluation results of news recommendation

	Success rate
Paragraph Vector	4.31%
Average BOW	3.09%
Sohu (no extra features)	3.36%
Sohu+rec (no extra features)	3.69%
Sohu+rec+seq (no extra features)	6.73%
Sohu	2.95%
Sohu+rec	3.61%
Sohu+rec+seq	**7.53**%

sequence fine-tuning. On the other hand, the result of Paragraph Vector cannot be further improved by fine-tuning. A possible explanation is that the Paragraph Vector model assigns a distinct vector to each of the news articles, thus inconsistency between news in different news browsing sequences is easily created by the sequence-level fine-tuning. On the other hand, fine-tuned News2vec embeddings carry the information of both the news network and the browsing sequences by treating the browsing sequence as a special sequence generated by the biased random walk. In addition, it is found that with fine-tuning, the additional features tend to have positive effects on the recommendation task (see Sohu to Sohu+rec+seq without extra features and Sohu to Sohu+rec+seq with extra features). As a matter of fact, additional news features, such as the release time and type, are considered to contain important information for news recommendation tasks. Unfortunately, due to the lack of data, only two extra features are included in this experiment.

3.4 Related Works

To represent news articles as vectors, two possible ways are emerged. As news articles are essentially textual data, one could use directly text/sentence embedding models to embed news. Alternatively, news event embedding models are developed with the argument that news stories describe events.

In terms of **Text/Sentence Embedding**, one of the simplest but robust baseline model [18] is to perform numerical operations on the existing contextual word vectors [45]. SDAE [94] uses an auto-encoder to project text into a low-dimensional vector space, whilst Paragraph Vector [45] learns the distributed representation of sentences and documents by

predicting their contexts, i.e., words in the sentence/document. Other unsupervised models include Skip-Thought [50], FastSent [34] and Quick-Though [63] and learn distributed sentence-level representations regarding to the coherence among sentences across the text. BERT [20] is a powerful pre-trained sentence encoder trained on unsupervised data sets based on the masked language model and next-sentence prediction. As for supervised learning models, most studies use a sentence encoder such as LSTM or Transformer [91] to convert word vectors to sentence vectors. The sentence encoder is usually trained on NLP tasks, which could be a single supervised learning task, such as InferSent [17] which uses only the language inference data, or multiple tasks [11] which involve both unsupervised and labeled corpus.

On the other hand, **Event Embedding** models emphasize the mining of event tuples and their projections onto the vector space. Ding et al.'s Event Embedding model [21, 22] extracts event tuples $E = (Actor, Predicate, Object)$ from headlines and learn dense vector representations by scoring correct tuples higher than corrupted tuples. Basically, the event embedding model is built on the knowledge graph embedding after decomposing the knowledge graph into tuples of entities, relations, etc. Following Ding et al.'s idea, news events could then be embedded with other knowledge graph embedding models such as TransE [6], which exploits scoring functions based on the distance between $h + r$ and t. In addition to knowledge graph embedding, Event2vec [87] employs network embedding to learn distributed representations of summarized news events by proposing an event network where each news event node is connected to the associated entities, the event type and the year of the event. The major limitations of Event2vec are threefold. First, it only works for artificially organized database of already extracted events. Second, the heterogeneous network it constructs raises difficulties in determining the weights of element nodes of different types which leads to further imposed assumptions on the importance of the elements. Third, it lacks a solution to represent unseen events based on the trained network.

In addition to converting news into continuous vectors, structured **Event Extraction** from news also allows statistical analysis based on the event type [31, 64]. In the early studies of news data mining, the textual information of news is often regarded as high-dimensional term vectors [83, 98, 103], which are then considered to lead to a great loss of information [53].

3.5 Conclusions

In this paper, we develop the News2vec model that learns the vector representation of news articles by constructing a news network. The Subnode model is further proposed to allow the embedding of unseen (news) nodes outside the existing network, at the same time to enrich the news vector with its associated feature vectors. Compared to other established text embedding models, the News2vec embedding contains not only the information of the contextual relationship between news and its event elements, but also potential connections as the network goes deeper. According to the dimension reduction plots and correlation heat-maps, it is suggested that the news/feature vectors contain adequate information as expected. Two downstream tasks, the news-driven stock movement prediction and news recommendation, show the News2vec embeddings demonstrate the latent connections between news articles, and the integration of news features enhances the model's performance in comparison to the baseline models.

Chapter 4

Parallel Hierarchical Transformer with Local and Global Cross Attention

4.1 Introduction

Since [89] propose the sequence-to-sequence (seq2seq) model for machine translation, the development of NLP applications has been almost inseparable from this framework. In the field of abstractive summarization, the seq2seq model is first applied by [82] to summarize sentences. With respect to the recent bloom of the attention mechanism and pre-trained models, great effort has been made to improve neural machine summarization upon extensions of seq2seq [27, 85, 111]. With the promising results on single documents [85, 27, 48, 75, 58, 57], there are increasing recent attempts to study abstractive multi-document summarization (MDS) in the seq2seq framework [60, 46, 26, 61, 65].

This study makes an exploratory attempt to improve the established abstractive summarization models for multi-document summarization (MDS) utilizing the Transformer [91] architecture. In comparison to single-document summarization, MDS brings challenges on the representation and coverage of its lengthy and linked sources. [60] propose a two-stage model to first extractively select the important paragraphs, then train the concatenated flat sequences using the Transformer-decoder with memory compressed attention (T-DMCA). Although the two-stage approach effectively reduces redundant information of

source documents and retains salient information as inputs, it fails to take into account the cross-document relationship in its summaries. On the other hand, [61] propose a Hierarchical Transformer (HT) with local and global encoder layers to represent cross-token and cross-document information. Summaries are then generated based on a vanilla Transformer [91] by concatenating document-information-enriched token embeddings to a flat sequence. The essentially flat nature of the model leads to restrictions on learning dependencies of input sequences longer than 2000 tokens [60].

As a solution to better process the long-term dependency and cross-document relationship in MDS, this study develops a novel Parallel Hierarchical Transformer (PHT) with the paragraph-level attention alignment. Operationally, PHT first creates the word- and paragraph-level context vectors from a shared encoder, then generates summaries by the the word- and paragraph-level multi-head attentions parallel to each other in the decoder. In this way, PHT allows a pure hierarchical learning structure extended from the vanilla Transformer [91] to learn both cross-token and cross-document relationships. The word- and paragraph-level context vectors are then jointly used to generate target sequences in order to address the long-dependency problem of the flat structure, thus to permit extended length of input document. Our experiments show the sole PHT model has already the capacity to outperform other strong Transformer-based summarization models.

To address the coverage of the multi-document summaries, the decoding inference is further modified according to the proposed attention-alignment algorithm. As the original beam search prefers to generate typical and dull sentences to avoid making mistakes [36], the paragraph-level attention alignment mechanism is designed to regulate generated summaries to attend to source contents with the optimal coverage of salient information. Inspired by Google's Neural Machine Translation (NMT) [102], attention alignment taps into the determination of the optimal attention distribution of source paragraphs on summaries, by predicting the reference attention from the source. The score function of the beam search is then refined in order to select summaries closest to the predicted optimal attention distribution. With significantly elevated ROUGE scores, it is evident that incorporating the attention-alignment mechanism further enhances the quality of generated summaries with minor computational cost-added from a shallow attention-prediction model, of which inputs and labels are both extracted from the PHT model.

With regards to the core target of developing an enhanced paradigm for multi-document summarization based on Transformer, the contribution of this study is twofold. First, the hierarchical architecture with parallel multi-head attentions is designed to represent and

exchange token- and document-level information for the generation of summaries based on the lengthy inputs. The effectiveness of the PHT model is investigated relative to a variety of summarization models, in terms of the ability to capture cross-document relationship, computational efficiency and improvements on the summarization quality. Second, the paragraph-level attention-alignment mechanism is proposed to guide the generated summaries in the decoding stage to calibrate the original beam search according to the learned attention distribution. The merits of attention alignment are not only reflected by promoting the optimal coverage of the generated summary to the source, but also its practical value of low computational cost and potential to be adopted by other attention-based summarization models.

The remaining of the paper is organized as follows. Section § 4.8 discusses the related work. Section § 4.2 and § 4.3 introduce the methodology associated with the Parallel Hierarchical Transformer and the attention-alignment mechanism. Section § 4.5 and § 4.6 describe the experimental setups and analyze the results. Section § 4.9 concludes.

4.2 Parallel Hierarchical Transformer

This section discusses the design of the Parallel Hierarchical Transformer for MDS. Figure 4.1 graphically presents the architecture of the proposed PHT. The encoder and decoder are displayed the second and third blocks on the left (highlighted in purple), respectively. The process of generating summaries is described as follows.

4.2.1 Encoder

As shown in Figure 4.1, the PHT encoder is shared by all paragraphs and consist of two major units, i.e. the Transformer encoder and the additional Multi-head Attention Pooling layer, to obtain the token- and paragraph-embeddings, respectively. To be specific, context-aware word embeddings $\boldsymbol{C}_p \in \mathbb{R}^{n \times d}$ in the paragraph p of length n are first produced as the output of the Transformer encoder $\text{TransE}(\cdot)$ based on the summation of word embeddings $\boldsymbol{W}_p \in \mathbb{R}^{n \times d}$ in the paragraph and their fixed positional encodings $\boldsymbol{E} \in \mathbb{R}^{n \times d}$ [91].

$$\boldsymbol{C}_p = \text{TransE}(\boldsymbol{W}_p + \boldsymbol{E}) \tag{4.1}$$

The context-aware word embedding is then used to compute paragraph embeddings as well as being a part of inputs to the PHT decoder to calculate word-level cross attention.

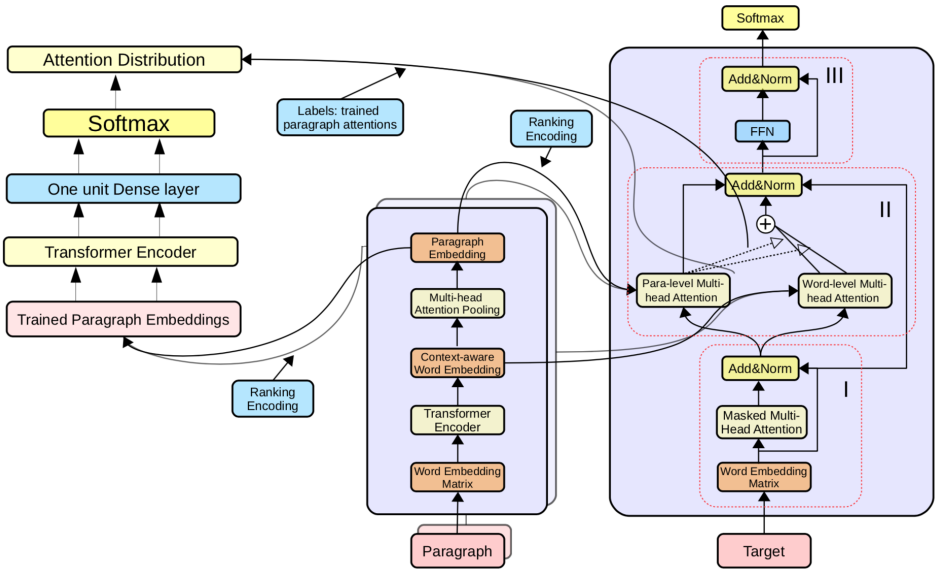


Figure 4.1: Model flowchart with two input paragraphs. Middle and right: PHT encoder and decoder (See Section § 4.2). Left: prediction model of the optimal attention distribution (See Section § 4.3).

At the second step, PHT achieves paragraph embeddings based on $\boldsymbol{C}_p$ using Multi-head Attention Pooling:

$$\boldsymbol{head}_p^h = \text{HeadSplit}\left(\boldsymbol{C}_p \boldsymbol{W}_1\right) \tag{4.2}$$

$$\boldsymbol{\phi}_p^h = \left(\text{Softmax}(\boldsymbol{head}_p^h \boldsymbol{W}_2)\right)^\top \boldsymbol{head}_p^h \tag{4.3}$$

$$\boldsymbol{\phi}_p = \boldsymbol{W}_3 \left(\Big\Vert_{h=1}^{heads} \boldsymbol{\phi}_p^h \right) \tag{4.4}$$

$$\boldsymbol{\phi}_p := \text{LayerNorm}\left(\boldsymbol{\phi}_p + \text{FFN}(\boldsymbol{\phi}_p)\right) \tag{4.5}$$

where $\Vert$ represents concatenation, and $\boldsymbol{W}_1 \in \mathbb{R}^{d \times d}$, $\boldsymbol{W}_2 \in \mathbb{R}^{d_{head} \times 1}$, $\boldsymbol{W}_3 \in \mathbb{R}^{d \times d}$ are linear transformation parameters. Besides, $\boldsymbol{head}_p^h \in \mathbb{R}^{n \times d_{head}}$ denotes the h_{th} attention head, and $\boldsymbol{\phi}_p^h \in \mathbb{R}^{d_{head}}$ is paragraph embedding of the head. These head embeddings are concatenated and fed to a two-layer feed forward network (FFN) with Relu activation function after linear transformation. The paragraph embedding is another input to the

decoder for obtaining paragraph-level cross attention, together with the context-aware word embedding.

4.2.2 Decoder

The PHT decoder accepts three classes of inputs, namely the target summary, context-aware word embeddings in the p_{th} paragraph $\boldsymbol{C}_p \in \mathbb{R}^{n \times d}$ where n is the length of the paragraph, and paragraph embeddings $\boldsymbol{\Phi} \in \mathbb{R}^{m \times d}$ where m is the number of paragraphs. Let $\boldsymbol{X}^{(\mathrm{I})} \in \mathbb{R}^{k \times d}$ denote the output of decoder part I, where k is the length of target sequence or the number of time steps. Note that both the word embedding and vocabulary in the decoder part are shared with the encoder.

Different from the token-level ranking encoding [61], we intend to incorporate the information of paragraph importance to their embeddings. Specifically, ranking encoding[1] $\boldsymbol{R} \in \mathbb{R}^{m \times d}$ created by the positional encoding function [91] are added to the original paragraph embeddings:

$$\boldsymbol{\Phi} := \boldsymbol{\Phi} + \boldsymbol{R} \tag{4.6}$$

PHT decoder consists of three parts. Similar to a vanilla Transformer [91], the first and last parts of the PHT decoder are the masked multi-head attention and the feed forward network, whereas the second part includes two parallel-computing cross attention models to respectively capture the mutual information the target summary shares with source paragraphs and source words.

Paragraph-level cross attention (global cross attention). This cross attention model is to calculate the attention distribution that the decoder assigns to the paragraphs at each step, and at the same time represents the cross-paragraph relationships as paragraph-level context vectors. The query is the output of part I: $\boldsymbol{X}^{(\mathrm{I})} \in \mathbb{R}^{k \times d}$ where k is the length of the target sequence. The key and value are context-aware paragraph embeddings $\boldsymbol{\Phi}$.

$$\boldsymbol{X}^{\langle \mathrm{para} \rangle}, \boldsymbol{A}^{\langle \mathrm{para} \rangle} = \mathrm{MultiHead}\left(\boldsymbol{X}^{(\mathrm{I})}, \boldsymbol{\Phi}, \boldsymbol{\Phi}\right) \tag{4.7}$$

where $\boldsymbol{X}^{\langle \mathrm{para} \rangle} \in \mathbb{R}^{k \times d}$ is the weighted summation of paragraph embeddings, and $\boldsymbol{A}^{\langle \mathrm{para} \rangle} \in \mathbb{R}^{k \times m}$ is the paragraphs attention weights [2].

Word-level cross attention (local cross attention). This cross attention mech-

[1] We directly use the ranked paragraphs provided by [61].

[2] In this paper, average pooling is adopted to obtain the final attention from multi-head attention.

anism aims at modeling how the decoder attends to source tokens of the paragraph. It could be considered as the local cross attention of each paragraph since their calculations of different paragraphs are independent. By comparison, paragraph-level cross attention refers to the global cross attention which captures the dependencies among paragraphs. Since the calculations of the two cross attention are based on different encoder outputs, they are non-interfering and parallel. Finally, the mechanism produces the word-level context vectors for each paragraphs. The query of the self attention is $\boldsymbol{X}^{(\mathrm{I})}$, whilst the key and value are context-aware word embeddings $\boldsymbol{C}_p$.

$$\boldsymbol{X}_p^{\langle \mathrm{word} \rangle} = \mathrm{MultiHead}\left(\boldsymbol{X}^{(\mathrm{I})}, \boldsymbol{C}_p, \boldsymbol{C}_p\right) \tag{4.8}$$

where $\boldsymbol{X}_p^{\langle \mathrm{word} \rangle} \in \mathbb{R}^{k \times d}$ denotes the word-level context vectors of all time steps in the p_{th} paragraph.

Multi-level attention fusion. Since the word-level cross attention model need to be implemented for each paragraph independently, there are totally m word-level context vectors $\boldsymbol{X}_p^{\langle \mathrm{word} \rangle}$, equivalently denoted as $\boldsymbol{\mathcal{X}}^{\langle \mathrm{word} \rangle} \in \mathbb{R}^{k \times d \times m}$. To fuse it with the paragraph-level context vectors $\boldsymbol{X}^{\langle \mathrm{para} \rangle} \in \mathbb{R}^{k \times d}$ and part I hidden states $\boldsymbol{X}^{(\mathrm{I})} \in \mathbb{R}^{k \times d}$, we need to integrate m groups of context vectors $\boldsymbol{\mathcal{X}}^{\langle \mathrm{word} \rangle}$ to one group. The straightforward way is to use mean pooling or max pooling, but both may cause loss of context information. An alternative approach is the adaptive attention pooling but not conductive to the computational efficiency. To handle the two problems, we directly integrate $\boldsymbol{\mathcal{X}}^{\langle \mathrm{word} \rangle}$ with knowledge learned by the paragraph-level cross attention model, i.e., using paragraph attention $\boldsymbol{A}^{\langle \mathrm{para} \rangle}$ to weight the context vectors $\boldsymbol{X}_p^{\langle \mathrm{word} \rangle}$ of the corresponding paragraph. The related matrix calculation process is as follows:

$$\boldsymbol{X}^{\langle \mathrm{int} \rangle} = \boldsymbol{\mathcal{X}}^{\langle \mathrm{word} \rangle} \boldsymbol{\mathcal{A}}^{\langle \mathrm{para} \rangle}, \tag{4.9}$$

where $\boldsymbol{\mathcal{X}}^{\langle \mathrm{word} \rangle} \in \mathbb{R}^{k \times d \times m}$, $\boldsymbol{\mathcal{A}}^{\langle \mathrm{para} \rangle} \in \mathbb{R}^{k \times m \times 1}$, and matrices are multiplied in the last two dimensions. The output of part II $\boldsymbol{X}^{(\mathrm{II})}$ is expressed as:

$$\boldsymbol{X}^{(\mathrm{II})} = \mathrm{LayerNorm}\left(\boldsymbol{X}^{(\mathrm{I})} + \boldsymbol{X}^{\langle \mathrm{para} \rangle} + \boldsymbol{X}^{\langle \mathrm{int} \rangle}\right). \tag{4.10}$$

With the outputs of part II, we are able to proceed to part III and compute the final probability distributions.

4.3 Attention-Alignment Mechanism

To further enhance the coverage of multi-document summarization, this section introduces the attention-alignment mechanism to guide the text decoding. The algorithm first predicts the optimal attention distribution of source paragraphs, then regulates the beam search according to the scoring function derived from the predicted attention distribution. Note that the attention-alignment mechanism is implemented after the training of PHT, in order to allow the extraction of the attention distribution from the trained parameters.

4.3.1 Learn from Neural Machine Translation (NMT)

The idea of the Attention-Alignment is inspired by Google's NMT [102], where candidates in the beam search are re-ranked according to a refined score function with the length normalization and coverage penalty. The penalty function is based on the assumption of one-to-one alignment in the translation so that $\sum_{t=1}^{T} \alpha_{t,i} = 1$, where $\alpha_{t,i}$ indicates the attention weight of the t_{th} translated word on the i_{th} source word. To penalize the situation that source words are not fully covered, i.e. the sum of attention weights is less than one, the coverage penalty is defined as:

$$cp = \sum_{i=1}^{n} \log \left(\min \left(\sum_{t=1}^{T} \alpha_{t,i}, 1 \right) \right) \tag{4.11}$$

This assumption is not tenable for summarization as uniform coverage is no longer required. Pointer-generator [85] re-defines the coverage loss for summarization as:

$$cp_t = \sum_{i} \min \left(\alpha_{t,i}, \sum_{t'<t} \alpha_{t',i} \right) \tag{4.12}$$

where $\alpha_{t,i}$ is the word-level attention distribution and $\sum_{t'<t} \alpha_{t',i}$ is the coverage vector. In this way, repeated attention is penalized according to the overlap between the attention distribution and the coverage til time step t.

[55] further corporate this concept to their structural-coverage regularization, forcing the generation to focus on different source sentences to raise the diversity of the summary.

In detail, the structural-coverage is defined as:

$$strCov(\alpha_t) = 1 - \sum_i \min\left(\alpha_{t,i}, \sum_{t'<t} \alpha_{t',i}\right) \tag{4.13}$$

which is rather similar to the coverage function of Pointer-generator [85] except that [55] consider the sentence-level attention $\alpha_{t,i}$.

In summary, both the Pointer-generator [85] and structural-coverage regularization [55] build their models based on the principle of searching for words/sentences that have previously attracted less attention to avoid repetition, thus to increase coverage. Comparing NMT's coverage penalty with the coverage functions in the aforementioned models [85, 55], the restriction of summarization is rooted in the absence of the optimal attention distribution of contents, that maps a holistic layout of the summary with comprehensive coverage. This motivates us to develop the attention-alignment inference to address this matter.

4.4 Paragraph-level Attention Alignment

4.4.1 Optimal Attention Distribution

To explicitly express the coverage of source content, the first step of attention alignment is to use the encoded paragraph embeddings to predict the optimal attention distribution of input paragraphs. Specifically, the attention prediction model is trained from the label of paragraph-level attention distribution $\boldsymbol{\eta} \in \mathbb{R}^m$ (m is the number of paragraphs) calculated from paragraph attention weights $\boldsymbol{A}^{\langle \text{para} \rangle}$ in Eq. 4.7 and

$$\boldsymbol{A}^{\langle \text{para} \rangle} = \begin{bmatrix} \alpha_{1,1} & \dots & \alpha_{1,m} \\ & \ddots & \vdots \\ \alpha_{k,1} & & \alpha_{k,m} \end{bmatrix} \tag{4.14}$$

where $\alpha_{t,p} \in \boldsymbol{A}^{\langle \text{para} \rangle}$ denotes the attention weight of the t_{th} summary word on the p_{th} source paragraph[3].

$$\eta_p = \frac{\sum_{t=1}^{k} \alpha_{t,p}}{\sum_{p=1}^{m} \sum_{t=1}^{k} \alpha_{t,p}} \tag{4.15}$$

[3]In the case of multiple decoder layers, the final paragraph attention are the summation of paragraph attentions in each layer.

$$\boldsymbol{\eta} = [\eta_1, \cdots, \eta_p, \cdots, \eta_m] \tag{4.16}$$

Since the reference summary is known for training data, $\boldsymbol{\eta}$ is regarded as the optimal attention distribution and serves as the training label of the attention-prediction model. In other words, the labelling process only utilizes paragraph attention weights from the already-trained PHT parameters. Besides, the inputs of the attention-prediction model are extracted from the PHT, which are paragraph embeddings $\boldsymbol{\Phi}$ in Eq. 4.6. The training process is displayed in Figure 4.1.

As for the construction of the attention prediction model, paragraph embeddings are first input to a Transformer-encoder to obtain the context-aware paragraph embeddings in order to make full usage of the context information between paragraphs. The context-aware paragraph embeddings are then linearly transformed and converted to m (i.e. the number of paragraphs) units before normalized by softmax. Given the nature of the prediction is regression, mean square error (MSE) is used as the loss function.

During inference, the source paragraphs are first fed to the PHT encoder to obtain the paragraph embeddings $\boldsymbol{\Phi}$, based on which the trained attention-prediction model predicts the optimal attention distribution $\widehat{\boldsymbol{\eta}}$.

4.4.2 Attention Alignment Score

In line with NMT, the score function of the pure beam-search is modified taking into account the predicted optimal attention distribution. With length normalization and the attention-alignment score, the score of each candidate hypothesis is given by:

$$score(\boldsymbol{y}) = \frac{\log\left(p(\boldsymbol{y}|\boldsymbol{x})\right)}{|\boldsymbol{y}|} + \beta * attAlign(\boldsymbol{y}) \tag{4.17}$$

$$attAlign(\boldsymbol{y}) = \sum_{p=1}^{m} \log\left(\min\left(\eta_p^{(\boldsymbol{y})}, \widehat{\eta}_p\right)\right) \tag{4.18}$$

where $\boldsymbol{x}$ denotes the source, $\boldsymbol{y}$ refers to a candidate hypothesis, and $\widehat{\eta}_p \in \widehat{\boldsymbol{\eta}}$. Notably, different from the η_p of reference, $\eta_p^{(\boldsymbol{y})}$ is obtained from the generated candidate summary.

This predicted optimal paragraph attention $\widehat{\eta}_p$ is compared with the paragraph attention $\eta_p^{(\boldsymbol{y})}$ in the real-time generation. Given any deviation between the real and optimal attention distributions, paragraphs that are assigned with underestimated attention place negative impacts on the overall scoring, whereas those with overestimated attention re-

ceive a constant score of $\widehat{\eta}_p$. Regarding the length normalization, we direct use the length $|\boldsymbol{y}|$, rather than $length^a$ [102], as it is empirically proven to be more suitable for longer summaries.

4.4.3 Why Paragraph Attention?

An alternative way to obtain the optimal attention distribution is to use the paragraph ranking generated by an extractive model that predicts the probability each source sentence/paragraph appears in the final summary. However, the prediction of extractive probabilities is a separate unit from the summarization model which results in problematic inconsistency with the paragraph attention during the decoding process.

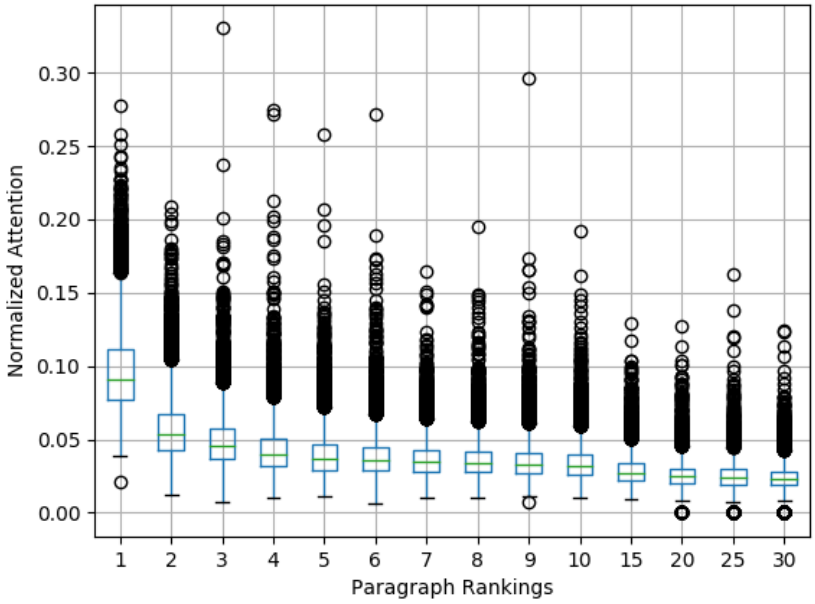


Figure 4.2: Box plot of paragraph attention with different initial rankings.

To support this argument with empirical evidence, Figure 4.2 randomly selects 10,000 training samples to compare the paragraph rankings [61] with the corresponding normalized paragraph attention by the trained HT decoder. In general, the decoder assigns higher attention to paragraphs with higher rankings. However, the outliers suggest that there are several cases of different judgements by the two approaches, which lead to potential

conflicts during the inference given the inconsistent measures between the optimal and real attention. Therefore, we make our prediction model to learn paragraph attention from the trained decoder directly. These attentions are considered optimal as the training targets are gold summaries. The prediction model maps the connection between source documents and optimal attention distributions, to allow the predicted attention distribution to maximally approach the optimal attention distribution if the target is unknown.

In addition to the inconsistency problem, it is easier to acquire the attention weight since attention exists in almost all neural abstractive summarization models, among which many (especially single-document summarization) do not require extractive probabilities. Besides, the attention-prediction model directly extracts input vectors and labels from the summarization model, whereas the extractive method requires extra effort on representing inputs and making labels.

4.5 Experiment Setup

4.5.1 WikiSum Dataset

Data sparsity has been the bottleneck of neural MDS models til WikiSum [60] came along. In this study, we use the ranked version of WikiSum provided by [61]. Each sample contains a short title, 40 ranked paragraphs with a maximum length of 100 tokens as source inputs, and a target summary with an average length of 140 tokens. Consistent with [61], the dataset is split with $1,579,360$ samples for training, $38,144$ for validation and $38,205$ for test. Subword tokenization [5] is adopted to tokenize our vocabulary to $32,000$ subword units to better represent unseen words.

4.5.2 Configuration

The proposed PHT is trained on a single *2080ti* with 0.3 dropout rate and an Adam optimizer of 16,000 warm-up steps. We stack 3-layer encoder-decoder of the PHT with 256 hidden units, 1024 FFN units and 4 headers, top 3000 tokens (30 paragraphs, 100 tokens) are used to train the PHT for approximately $600,000$ steps. Checkpoints are saved every $20,000$ steps and the best result on the validation set is used to generate the final summary. All parameters are randomly initialized including token embeddings. In the decoding process, we take 3000 tokens as inputs of PHT to generate summaries. Since the fixed positional encodings are used, so the attention-prediction model can accept inputs

of dynamic length. We set the beam size to 5 and terminate the inference til the length exceeds 200. In addition, we disallow the repetition of trigrams, at the same time block two tokens (except commas) before the current step to prevent degeneration situations. For the attention prediction model, we construct a two-layer Transformer encoder with dropout rate 0.5. The complete set of the training data is used to train the attention prediction for approximately $100,000$ steps.

4.5.3 Baselines

Summarization model

- **Lead** [69] is an extractive model that extracts the top K tokens from the concatenated sequence. In MDS, we combine paragraphs in order and place the title at the beginning of the concatenated sequence.
- **LexRank** [25] is a widely-used graph-based extractive summarizer.
- **Flat Transformer (FT)** is the vanilla Transformer encoder-decoder model [91]. We adopt a 3-layer Transformer in this study.
- **T-DMCA** [60] is a Transformer-decoder model that splits a concatenated sequence into segments, and uses a Memory Compressed Attention to exchange information among them.
- **Transformer-XL** [19] is a language model that excels in handling excessively long sequences. This model improves the vanilla Transformer-decoder with the recurrent mechanism and relative positional encoding.
- **Liu's Hierarchical Transformer (Liu's HT)** [61] uses a hierarchical structure to enrich tokens with information from other paragraphs before inputting to the Flat Transformer.
- **GraphSum** [54] is an graph-based hierarchical transformer, where graph neural network is used to capture cross-document relationships.
- **Parallel Hierarchical Transformer (PHT)** is the model proposed in this paper. Different from Liu's HT, our hierarchical structure could compute token-level and paragraph-level dependencies, thus not requiring FT.

Decoding strategy

- **PHT with vanilla beam search.** Beam search is a well-established baseline that is popularly-used in text generation tasks of all sorts.

- **PHT with the structural-coverage (strCov).** As the original study [55] summarizes single document using a hierarchical decoding algorithm to first decode sentence by sentence then realize the sentence word by word, we need to modify the regularization to adjust to the word-by-word inference. Therefore, we re-define $\alpha_{t,i}$ (in Eq. 4.13) from the attention of the t_{th} generated sentence on the i_{th} source sentence to the t_{th} generated word on the i_{th} source paragraph. To obtain an independent observation on the effect of the structural coverage, we skip the structural-compression regularization and other modifications on the loss function as discussed in [55].

- **PHT with extractive probability (extProb)** also adopts attention alignment mechanism but replaces the learned optimal attention distribution $\widehat{\boldsymbol{\eta}}$ by the normalized extractive probabilities of paragraphs. We use the extractive method in Liu's HT to calculate these probabilities.

- **PHT with the attention alignment mechanism (attAlign)** is PHT combined with the proposed attention-alignment mechanism. We probe the optimal value of the attention-alignment coefficient β in Eq. 4.17 by a numerical comparison for $\beta \in [0.2 : 0.2 : 1]$ on ROUGE. The result of development set suggests the optimal value of β is approximately 0.8 for the Wikisum dataset.

4.6 Results

4.6.1 Automatic Evaluation

In this section, we adopt a group of widely-used evaluation metrics ROUGE [59] to evaluate the MDS models. ROUGE-1 & -2 and ROUGE-L F_1 scores are reported in Table 4.1 assessing the informativeness and fluency of the summaries, respectively.

As shown in Table 4.1, the extractive model Lead exhibits overall inferior performance in comparison to the abstractive models, except that it produces a 0.11-higher ROUGE-L than the Flat Transformer. Although Liu's HT improves FT with a hierarchical structure, it fails to outperform the two extended flat models, i.e. T-DMCA and Transformer-XL,

Table 4.1: Average ROUGE F_1 scores of different summarization models.

Model	ROUGE-1	ROUGE-2	ROUGE-L
Lead	36.40	16.66	32.95
FT	40.30	18.67	32.84
T-DMCA	41.09	19.78	33.31
Transformer-XL	41.11	19.81	33.72
Liu's HT	40.83	19.41	33.26
1-layer PHT	41.02	19.82	33.28
PHT	41.99	20.44	34.50
PHT + attAlign	**42.58**	**20.84**	**35.66**

that are developed to learn lengthier inputs. Moreover, T-DMCA and Transformer-XL report comparable results in terms of the informativeness (ROUGE-1 & -2), whilst the latter outperforms the former by 0.41 in terms of the fluency (ROUGE-L).

Further, the proposed PHT model shows promising ROUGE results. Benefited from the pure hierarchical structure that allows prolonged token inputs, PHT outperforms Liu's HT in all domains of the ROUGE test. Moreover, the models' potential to be deepened is suggested by enhanced results of the 3-layer architecture over the 1-layer architecture. The ultimate 3-layer PHT stably surpasses T-DMCA and Transformer-XL, that are also tailored to handle long input sequences of 3,000 tokens, due to its hierarchical processing of token and document-level information.

Table 4.2: Average ROUGE F_1 scores of different decoding strategies.

Parallel HT	ROUGE-1	ROUGE-2	ROUGE-L
+ vanilla beam search	41.99	20.44	34.50
+ strCov [55]	41.74	20.25	33.88
+ extProb	42.17	20.46	34.79
+ attAlign	**42.58**	**20.84**	**35.66**

Table 4.2 shows the average ROUGE F_1 scores of all model combinations investigated. The attention-alignment mechanism promotes the quality of summaries by raising ROUGE-1 by 0.59, ROUGE-2 by 0.4, ROUGE-L by 1.16 for PHT with beam search. Technically, the attention alignment mechanism could be applied to all hierarchical models with an attention mechanism. Further, Table 4.2 provides empirical evidence to Section § 4.4.3,

suggesting that extractive probabilities (extProb) are not as good protocols as the paragraph attention for the optimal attention distribution, given the marked decline in the ROUGE scores in comparison to the proposed attention-alignment mechanism.

Besides, the structural-coverage mechanism hinders the performance of MDS with reduced ROUGE scores. We print its beam-search scores and find consecutive zeros as the generated sequences get longer, resulted from the t_{th} word attention on the i_{th} paragraph ($\alpha_{t,i}$) remains lower than the its cumulative attention ($\sum_{t'<t} \alpha_{t',i}$). Therefore, it is concluded that the structural coverage regularization is not particularly suitable for word-by-word summarization with lengthy inputs, that come along with multiple documents.

4.6.2 Human Evaluation

Table 4.3: Human evaluation results.

Model	Informativeness	Fluency	Conciseness	Factual consistency
T-DMCA	3.69	3.66	3.82	3.04
Transformer-XL	3.57	3.71	3.77	2.88
Liu's HT	3.34	3.76	3.75	2.82
PHT	4.11	3.97	3.81	3.11
PHT with attAlign	**4.39**	**4.22**	**3.95**	**3.35**
Average	3.77	3.89	3.84	3.05

To provide a better comparison between the MDS models, we select 4 representative summarization models with the best ROUGE performances in the human evaluation, namely T-DMCA & Transformer-XL (flat structure), and Liu's HT & PHT (hierarchical structure), and two decoding strategies including the original Beam search and attention alignment.

In the survey, multi-document summaries are scored from four perspectives, including (A) **Informativeness** (Does the summary include important information in the gold summary), (B) **Fluency** (Is the summary fluent and grammatically-correct), (C) **Conciseness** (Does the summary avoid repetition and redundancy), (D) **Factual consistency** (Does the summary avoid common sense mistakes such as wrong date, wrong location, or anything else against facts). We specify five ratings from *Very poor* (1) to *Very good* (5) to assess criteria (A)-(C), and three ratings of *Much better* (2), *Better* (1), and *Hard to*

score (0) to assess criterion (D). Twenty examples are randomly selected from generated summaries. Fifteen human evaluators participated in the experiment.

The results are displayed in Table 4.3. The general observation is twofold. A) Discrepancy exists in the ROUGE and human evaluations. For instance, T-DMCA tends to yield higher human scores relative to Transformer-XL although ROUGE suggests the opposite. Given the merits and weaknesses of the different metrics, we focus on discussing results that exhibit consistency in different parts of evaluation. B) Given the lowest average mark, factual consistency appears to be the bottleneck of abstractive summarization models that hinders human experience on the machine generated summaries.

As far as the summarization models are concerned, PHT achieves the highest human evaluation scores in all four areas investigated. On the other hand, the other hierarchical baseline, Liu's HT, turns to be less competitive than the flat structures in terms of informativeness, conciseness and factual consistency, possibly due to its length limit of input. With regards to the optimal attention distribution of summaries, attention-alignment is proven effective in improving the hierarchical model.

4.7 Analysis

Through the preliminary analysis on PHT, we intend to obtain an initial understanding of the hierarchical model on its capacity to better express the cross-document relationship, as well as the associated computational cost.

4.7.1 Cross-document Relationship

Cross-document relationships could be reflected by the distribution of paragraph attentions. If a model assigns higher attention weights to more important paragraphs and vice versa, the model is believed to have greater capacity of capturing cross-document relationships. To analytically assess the models' performance in this aspect, we use paragraph attentions of reference summaries as the gold attention distribution, and its cosine similarity to the attention distribution of generated summaries as the evaluation metric. To model the paragraph attention of the reference, we compute the normalized tf-idf similarities between the gold summary and each input paragraph as the gold attention distribution. For the baseline models, the summation of token weights in each paragraph are computed to indicate each paragraph's attention, whilst PHT returns the paragraph attention distribu-

tion directly from its paragraph-level multi-head attention.

Table 4.4: Average cosine similarities between attention distributions of generated summaries and the reference.

Model	Cosine similarity
Lead	0.8098
Flat Transformer	0.8143
T-DMCA	0.8654
Transformer-XL	0.8447
Liu's HT	0.8769
PHT	**0.8936**

As suggested by Table 4.4, hierarchical structures place significant improvements on the flat models in learning cross-document dependencies by assigning paragraph attentions in a way that is closer to the gold summaries. Moreover, PHT generates summaries of the greatest similarity 89.36% with the gold summaries, most likely due to its paragraph-level multi-head attention in addition to the token-level one, allowing the exchanging of cross-document information.

4.7.2 Computational Efficiency

This section assesses the computational efficiency of PHT comparing to other neural abstractive models in three aspects, i.e. the memory usage, parameter size and validation speed. We uniformly hire the 3-layer architecture and 1600 input tokens in this part to ensure fairness. During the experiment, we increase the batch size until out of memory in a 2080ti GPU, and the model with the maximum batch size occupies the lowest memory space. To measure the parameter size, we count the number of parameters in the neural network. Finally, we run each trained model in the validation set (38,144 samples), and the average time consumed in each checkpoint is used to evaluate the speed of forward-propagating in the model.

As indicated by higher batch sizes in Table 4.5, models in the hierarchical structure (second panel) appears to be overall more memory-saving than those in the flat structure (first panel), with higher requirements on the parameters. In particular, models based on the Transformer-decoder, i.e. T-DMCA and Transformer-XL, demonstrate absolute superiority in reducing the parameter size. As for the speed of forward-propagating,

Table 4.5: Computational efficiency.

Model	**Max Batch Size**	**Parameters** (MB)	**Validation Speed** (s)
Flat Transformer	11	165.0	634
T-DMCA	10	131.1	656
Transformer-XL	8	**130.4**	**489**
Liu's HT	11	190.8	639
PHT	**17**	182.4	601

Transformer-XL dominates due to its recurrent mechanism, whereas others share close performance in the inference speed. Between the two hierarchical models, PHT is proven to outperform Liu's HT in all three aspects, due to its parallel, rather than sequential, computation of the word & paragraph-level attention mechanisms.

4.8 Related Works

In general, hierarchical models are designed with strengthened capacity to handle lengthy inputs, which have been widely used in document classification [106] or large-document summariztion [55] tasks. In the filed of MDS, hierarchical structures allow not only to represent massive source inputs, but also to capture cross-document relationships. [26] use a hierarchical RNN structure with Maximal Marginal Relevance [10] to better select salient paragraphs and reduce repetitions in the summary. [111] pre-train a hierarchical BERT [20] by masking a sentence and using other sentences to generate the masked one. [61] propose a Hierarchical Transformer for multi-document summarization to enrich token embeddings with cross-document relationships. A vanilla Transformer [91] is then used to conduct summarization after combining the enriched token embeddings in a flat sequence [60].

With the aim to improve the quality of seq2seq summaries, existing studies tend to focus on the coverage of salient contents. [60] use a text-ranking mechanism to extract important paragraphs, which are later input to the neural abstractive model. [27] train a selector to predict the phrases ought to appear in the final summaries and use them as summarization inputs. [12] select and compress salient sentences that are later re-organized in the summaries. In addition to the two-stage extraction-abstraction approaches, attempts are made to build hybrid summarization models by incorporating the sentence-level atten-

tion [66, 15, 108], graph neural networks [90, 57, 54], Maximal Marginal Relevance [46, 26] or reinforcement learning [107]. Additionally, [49] add representations of key words to the summarizaiton model. [38] use an abstractive summarization model to concatenate extracted key phrases. Moreover, some studies suggest to improve the summarizaiton quality by modifying the objective function to encourage salient words [71] or to penalize repetitive generations [85, 99]. More details on representative methods of this sort and their differences with attention alignment have been discussed in Section § 4.3.1.

4.9 Conclusions

This study develops a Parallel Hierarchical Transformer with attention alignment inference for multi-document summarization. Using the Wikisum dataset, we empirically show that the proposed hierarchical architecture with token- and paragraph-level multi-head attentions excels in capturing the cross-document relationship of lengthy sources, and generates summaries of greater quality than other existing Transformer-based models. Further, the paragraph-level attention-alignment algorithm is designed to address the coverage issue by predicting the optimal attention distribution according to the multi-document sources. In theory, the decoding strategy has the potential to accommodate all seq2seq summarization models in the presence of the attention mechanism. Our experiment shows that attention alignment places significant improvements on the summaries generated by the original beam search.

Given the fact that the attention mechanism is nowadays almost a necessity in the seq2seq architecture, the authors target at investigating the capacity of word-level attention-alignment in the future study. The application of word-level attention alignment is no longer confined to the hierarchical architecture and can be adopted to all attention-based seq2seq models including the pre-trained model BART [48]. Different from paragraph-level attention alignment, word-level attention alignment though requires the processing of numerous attention units, bringing challenges in obtaining the optimal attention distribution where a dynamic scoring function might be developed for text decoding.

Chapter 5

Decoding with Awareness of Global Attention Distribution

5.1 Introduction

As the barriers exist from the auto-regressive design of neural probabilistic text generators to predicting the global optimum directly [89], the heuristic algorithm beam search that factorizes global optimization to multiple local optimizations, has been popularly used for text decoding [67]. In the original beam search setting, the global optimum is a hypothesis $\mathbf{g}$ of the highest probability among all possible sentences, and consists of words in vocabulary $\mathcal{V}$. Given the global optimum at step t denoted as $\mathbf{g}_{\leq t}$, the local optimums $\mathbf{l}_{\leq t}$ refer to $\mathcal{K}$ candidate sequences with the highest probabilities at each step. While it is necessary to compromise on the beam size $\mathcal{K} \ll \mathcal{V}$ to ensure text quality [16, 70, 67] and search efficiency, beam search suffers from a major limitation due to its local property. Concretely, assuming that the global optimal hypothesis is within the $\mathcal{K}$ local optimal hypotheses of the highest probabilities, i.e. $p(\mathbf{g}_{\leq t}) \geq p(\mathbf{l}_{\leq t})$, for all t until the termination T, it operates solely with the local information available at each step. In practice, such assumption may however fail in the case that the probability of the global optimum at step $\tau < T$ is less than those of the local optimums, i.e. $p(\mathbf{g}_{\leq \tau}) < p(\mathbf{l}_{\leq \tau})$, but is adjusted to a higher level in the later steps, $p(\mathbf{g}_{>\tau}|\mathbf{g}_{\leq \tau}) > p(\mathbf{l}_{>\tau}|\mathbf{l}_{\leq \tau})$ and $p(\mathbf{g}_{\leq \tau})p(\mathbf{g}_{>\tau}|\mathbf{g}_{\leq \tau}) > p(\mathbf{l}_{\leq \tau})p(\mathbf{l}_{>\tau}|\mathbf{l}_{\leq \tau})$. This often leads beam search to get stuck in the local optimum from step τ onward in generating texts.

To cope with this limitation, this study proposes a calibrated beam-based algorithm with global awareness at all searching steps. Generally, our novel algorithm is imple-

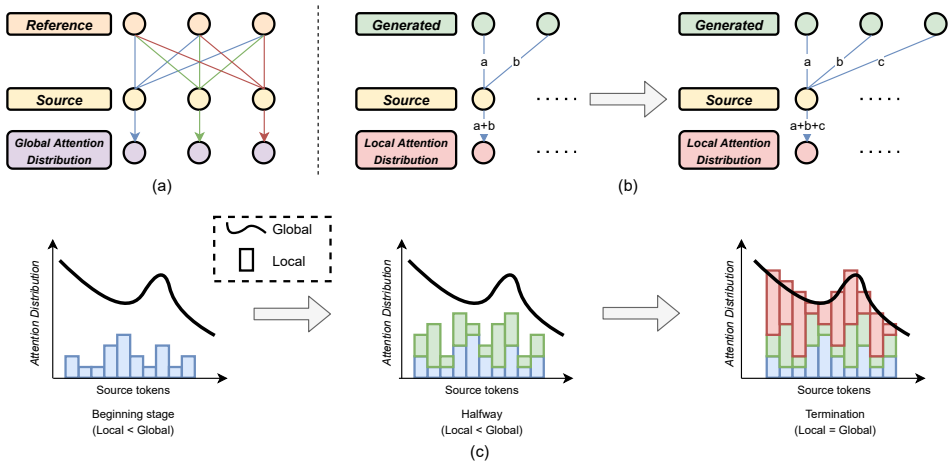


Figure 5.1: (a) Attention distribution is composed of the summation of cross attention on the same-colored lines, distinguished from that of different-colored lines which always equals 1 due to softmax. (b) Local attention gradually increases as the decoding proceeds. (c) Desired situation: growing local attention has been lower than global attention during decoding and exactly reaches it at the end.

mented in two phases. Before the beam search (Phase I), the *global attention distribution* is predicted in order to be included as a protocol to calibrate beam search at each step, encouraging the generated hypotheses to attend to the source in a more near-global optimal way. Specifically, the global attention distribution describes how all reference tokens should assign the attention to each source token (illustrated in Figure 5.1.a), which could be predicted from the source by training an attention-prediction model. The training is fairly straightforward and resembles a sequence tagging task [39], except that the predicted attention distribution from the source is a regression result. There are several advantages of using the attention distribution as the global protocol. 1) Attention distributions are sensitive to the decoder input, suggesting that any input to the decoder leads to a unique attention distribution with fixed model parameters; 2) attention distributions are accessible for almost all text generation tasks thanks to the recent advances in attention models [78, 20, 91]; 3) relying on the source only, the global attention distribution can be predicted before beam search, thus offering a rigorous mechanism to calibrate a global-aware beam search.

During beam search (Phase II), we develop a novel *global scoring mechanism* composed of attention scores and length rewards to guide beam search based on the predicted global

attention distribution. As one main theoretical result, we show that the attention score can be considered as the probability that generated texts attend to sources in line with the predicted global attention distribution. Specifically, the generated tokens in each step update the local attention distribution to source tokens dynamically, where the attention values grow monotonically as the generation proceeds (see Figure 5.1.b). Since the desired situation is that the local distribution reaches exactly the global distribution at the terminal step, we regulate the inference by discouraging local attention from exceeding their corresponding predicted global attention at all steps.

With regards to the core target to investigate the possible paradigm that improves beam search with global awareness during decoding, contributions of this study are summarized as follows:

- We argue that the limitation of beam search roots from its defect in finding the global optimal hypothesis. We improve the algorithm by proposing a global protocol to regulate beam search step-by-step. This paper is the first to predict and deploy the global attention distribution to calibrate the inference in a rigorous way, thus returning a hypothesis that attends to source tokens in a more near-global optimal manner. In contrast, previous works [102, 27, 55, 85, 56] try to use attention distributions to improve beam search, but ignore that the global attention distribution is predictable.

- A novel global scoring mechanism is designed to evaluate the generated sequences at each step based on the desired situation described in Figure 5.1.c. As theoretically justified, its major component can be elegantly integrated into beam search in the form of a probability so that merely $\mathcal{O}(\mathcal{K})$ of the time complexity is increased in each step (see Section § 5.3.1 for more details).

- The proposed algorithm with global awareness manifests a robust and plug-and-play property in enhancing beam search for neural abstractive summarization. Without requiring any model or parameter modification, the global-aware inference shows excellent performance in generating meaningful texts, even if the attention distribution is corrupted or not of its own. Further, it is identified that summaries generated by global-aware inference are both higher-quality and different from beam search hypotheses (see *Global-aware* in Table 5.1). More interestingly, we find that the generation style of a dataset could be transferred by the designated global attention

distribution. For instance, summaries of higher abstractness for CNN/DM could be generated by only replacing its global attention distribution with a highly abstractive distribution during inference, as presented in *Global-aware†* of Table 5.1.

- On the empirical side, we show that the proposed global-aware inference can stably and significantly boost two state-of-the-art summarization models BART [48] and PEGASUS [109] to produce higher quality summaries on nine datasets, even if only the empirical hyper-parameters are used.

Table 5.1: Use BART [48] fine-tuned in CNN/DM to generate summaries. *Global-aware* uses the attention distribution learned from CNN/DM, while *Global-aware†* takes the attention distribution learned from XSUM.

Beam search	President Obama says climate change is a public health issue that affects all of us . Obama: "No challenge poses more of a public threat than climate change" Obama: "Millions of people would lose their health insurance" if Affordable Care Act is not upheld . Obama: "I am not anticipating the Supreme Court would make such a bad decision"
Global-aware	President Obama says climate change is a public health issue that affects all of us . He says the average American can do their part to reduce their own carbon footprint . Obama did not appear particularly concerned about the Supreme Court challenge to the Affordable Care Act .
Global-aware†	President Barack Obama says climate change is a public health issue . He says the average American can do their part to reduce their carbon footprint .

5.2 Preliminaries

The proposed decoding strategy is applied in BART [48] and PEGASUS [109] to perform summarization. BART is a pre-trained seq-to-seq model whose structure essentially follows a vanilla Transformer encoder-decoder [91]. PEGASUS has a similar structure but is pre-trained on a larger dataset differently. Notably, the fine-tuned parameters of both models are downloaded from *HuggingFace Models*[1] and are fixed in all subsequent operations, where "fine-tuned" means the pre-trained model has been fine-tuned on a specific dataset.

5.2.1 Beam Search

Since the decoder of the seq-to-seq model is an auto-regressive model, the probability of a target sequence $\boldsymbol{y} = (y_0, \cdots, y_t, \cdots, y_T)$ can be factorized to the probabilities conditional

[1] https://huggingface.co/models

on the source $\boldsymbol{x} = (x_1, \cdots, x_i, \cdots, x_n)$ and $\boldsymbol{y}_{<t} = (y_0, \cdots, y_{t-1})$, i.e.,

$$p(\boldsymbol{y}|\boldsymbol{x}) = \prod_{t=1}^{T} p(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}),\ T \geq 1 \tag{5.1}$$

When $T = 0$, $\boldsymbol{y} = (y_0)$ and $p(y_0|\boldsymbol{x}) = 1$ because y_0 is a fixed start token. Beam search [28] is a decoding strategy to predict a target sequence by maximizing this factorization. Given a vocabulary set $\mathcal{V}$, at each inference step t, beam search selects a candidate beam set $\mathcal{B}_t^{\mathcal{K}} = \{\boldsymbol{b}_t^k\}_{k=1}^{\mathcal{K}}$ (where each beam $\boldsymbol{b}_t^k = (b_0^k, \cdots, b_t^k)$ is a candidate sequence) from an all-possible beam set $\mathcal{B}_t$ of size $\mathcal{K} \times |\mathcal{V}|$, namely,

$$\mathcal{B}_t = \left\{ \boldsymbol{b}_{t-1}^k \circ v \mid \boldsymbol{b}_{t-1}^k \in \mathcal{B}_{t-1}^{\mathcal{K}},\ v \in \mathcal{V} \right\} \tag{5.2}$$

$$\mathcal{B}_t^{\mathcal{K}} = \left\{ \boldsymbol{b}_t^k \mid \boldsymbol{b}_t^k = \text{argtopk}\left(\log p(\boldsymbol{b}_t|\boldsymbol{x})\right),\ \boldsymbol{b}_t \in \mathcal{B}_t \right\},\ t \geq 1 \tag{5.3}$$

where argtopk$(\cdot)$ outputs $\mathcal{K}$ beams with the highest conditional probability, and $\circ$ is the concatenation operation. Besides, $\mathcal{B}_0^{\mathcal{K}} = \{\boldsymbol{b}_0^k\}_{k=1}^{\mathcal{K}}$ where b_0^k is the start token. By Eq. 5.1, $\log p(\boldsymbol{b}_t|\boldsymbol{x})$ is an accumulated value. Its calculation can be simplified as:

$$\log p(\boldsymbol{b}_t|\boldsymbol{x}) = \begin{cases} \log p(\boldsymbol{b}_{t-1}^k|\boldsymbol{x}) + \log p(v|\boldsymbol{x}, \boldsymbol{b}_{t-1}^k), & t \geq 2 \\ \log p(v|\boldsymbol{x}, b_0^k), & t = 1 \end{cases} \tag{5.4}$$

where the value of $\log p(\boldsymbol{b}_{t-1}^k|\boldsymbol{x})$ is computed from the previous step. Therefore, at each step, we only need calculate the condition probability of each token in the vocabulary set.

A beam is terminated after it generates an ending token, and the beam set of $\mathcal{K}$ terminated beams is defined as $\mathcal{Y}$. The final hypothesis $\boldsymbol{y}^*$ is chosen from $\mathcal{Y}$ based on the beam probability normalized by $length^a$ where a is a hyper-parameter of length [102]:

$$\boldsymbol{y}^* = \underset{\boldsymbol{y}^k \in \mathcal{Y}}{\text{argmax}}\ \frac{\log p(\boldsymbol{y}^k \mid \boldsymbol{x})}{(|\boldsymbol{y}^k| - 1)^a} \tag{5.5}$$

where $\boldsymbol{y}^k = (y_0^k, \cdots, y_T^k)$. $|\boldsymbol{y}^k| - 1$ is used since the start token is not considered in calculating the length.

5.2.2 Attention Distribution

Attention distribution is a continuous vector whose element indicates the degree of attention paid to a source token. The element is formed by the accumulation of cross attention, i.e., $\sum_t \alpha_{t,i}$, where $\alpha_{t,i}$ refers to the cross attention that the t_{th} token in the target sequence gives to the i_{th} source token.[2] Specially, cross attention is a scaled dot-product [91] of hidden states of the source $\boldsymbol{x}$ and the target sequence $\boldsymbol{y}$. Notably, since Transformer-decoder is an auto-regressive model, the cross attention assigned by t_{th} target token is actually calculated by $\boldsymbol{y}_{<t} = (y_0, \cdots, y_{t-1})$.

Global Attention Distribution. The global attention distribution $\boldsymbol{g} = [g_1, \cdots, g_i, \cdots, g_n] \in \mathbb{R}^n$ is the attention distribution given by the reference, where global attention g_i refers to the total attention that the reference attends to the i_{th} source token, and n is the source length.

Optimal Length. The summation of $\boldsymbol{g}$, namely $\sum_{i=1}^{n} g_i$, is equal to $\sum_{i=1}^{n}\sum_{t=1}^{T} \alpha_{t,i} = T$ due to $\sum_{i=1}^{n} \alpha_{t,i} = 1$ in softmax operation, where T is the reference length, or equivalently the optimal length Z.

Local Attention Distribution. The local attention distribution $\boldsymbol{l}_t^k = [l_{t,1}^k, \cdots, l_{t,i}^k, \cdots, l_{t,n}^k] \in \mathbb{R}^n$ is the attention distribution of the k_{th} generated sequence and updated at each decoding step t. Thereinto, the local attention $l_{t,i}^k$ denotes the total attention paid to the i_{th} source token by the k_{th} beam sequence $(b_1^k, \cdots, b_t^k)$ and is dependent on the sequence generated before t, i.e., $\boldsymbol{b}_{t-1}^k = (b_0^k, \cdots, b_{t-1}^k)$.

5.3 Proposed Global-aware Inference

5.3.1 Global Scoring Mechanism

The global scoring mechanism consists of an attention scoring function and a length reward function. Given the global attention distribution $\boldsymbol{g}$, the attention scoring function $\mathcal{A}(\cdot)$ at the step t depends on $\boldsymbol{b}_{t-1}^k$,

$$\mathcal{A}(\boldsymbol{b}_{t-1}^k) = \frac{\sum_{i=1}^{n} \min(l_{t,i}^k, g_i)}{\zeta_t^k},\ \zeta_t^k = \sum_{i=1}^{n} l_{t,i}^k,\ t \geq 1 \tag{5.6}$$

[2]Mean pooling is used for multi-layers and multi-heads

where ζ_t^k indicates the total attention that the generated sequence $(b_1^k, \cdots, b_t^k)$ gives to the source, and $\zeta_t^k = |\boldsymbol{b}_{t-1}^k| = t$ because the assignable attention for each generated token is 1. Notably, Eq. 5.6 attains the maximum score provided that each $l_{t,i} \leq g_i$. As mentioned in Section § 5.1, the reason for this design is that we desire the local attention $l_{T,i}$ at the termination is exactly g_i, since the final hypothesis is expected to attend to source tokens in the global-optimal manner. Meanwhile, we have $l_{T,i} > l_{t,i}$ for $t < T$ because $l_{t,i}$ monotonically increases on t with $\alpha_{m,i}^l > 0$. Therefore, at any step, the local attention $l_{t,i}$ should not surpass g_i. Otherwise, the attention score will decline, and the penalty depends on the total amount by which these $l_{t,i}$ exceed g_i (see Theorem 1). Further, the attention score could be considered as the proportion of correctly assigned attention to the total attention given by the generated sequence, where correct assignment indicates that all parts of $l_{t,i}$ do not exceed g_i. Also, it could be interpreted as the correct allocation probability of local attention against the global attention distribution (see Corollary 1.1). In this case, the total attention score can be expressed as the same multiplicative form as Eq. 5.1 to be elegantly integrated into beam search.

Theorem 1. *Let* $\mathcal{M} = \left\{s \,:\, l_{t,s}^k > g_s\right\}$ *and* $\Delta_t^k = \left\{\delta \mid \forall s \in \mathcal{M},\ \delta = l_{t,s}^k - g_s\right\}$. *Given* $\Delta = \sum_{\delta \in \Delta_t^k} \delta$ *where* $\Delta \geq 0$, *then* $\mathcal{A}(\boldsymbol{b}_{t-1}^k)$ *decreases as* Δ *increases.*

Proof. By the formula derivation, Eq. 5.6 can be converted to:

$$\begin{aligned}
\mathcal{A}(\boldsymbol{b}_{t-1}^k) &= \frac{\sum_{i=1}^n \min(l_{t,i}^k, g_i)}{\zeta_t^k} \\
&= \sum_{i=1}^n \min\left(\frac{l_{t,i}^k}{\zeta_t^k}, \frac{g_i}{\zeta_t^k}\right) \\
&= \sum_{i=1}^n \min\left(0, \frac{g_i - l_{t,i}^k}{\zeta_t^k}\right) + \sum_{i=1}^n \frac{l_{t,i}^k}{\zeta_t^k}
\end{aligned} \tag{5.7}$$

Since $\zeta_t^k = \sum_{i=1}^n l_{t,i}^k$, we have $\sum_{i=1}^n \frac{l_{t,i}^k}{\zeta_t^k} = 1$. Besides, with $\zeta_t^k > 0$,

$$\forall s \in \mathcal{M},\ \min\left(0, \frac{g_s - l_{t,s}^k}{\zeta_t^k}\right) = \frac{g_s - l_{t,s}^k}{\zeta_t^k} = \frac{-\delta}{\zeta_t^k} \tag{5.8}$$

Thus Eq. 5.7 is equal to:

$$\begin{aligned}\mathcal{A}(\boldsymbol{b}_{t-1}^k) &= 0 + \sum_{\delta \in \Delta_t^k} \frac{-\delta}{\zeta_t^k} + 1 \\ &= 1 - \frac{\Delta}{\zeta_t^k}\end{aligned} \tag{5.9}$$

Then it is easy to prove that $\mathcal{A}(\boldsymbol{b}_{t-1}^k)$ goes down as Δ goes up. □

Corollary 1.1. *The bound of* $\mathcal{A}(\boldsymbol{b}_{t-1}^k)$ *is between* 0 *and* 1.

Proof. Since we have achieved $\mathcal{A}(\boldsymbol{b}_{t-1}^k) = 1 - \frac{\Delta}{\zeta_t^k}$ where $\mathcal{A}(\boldsymbol{b}_{t-1}^k)$ decreases monotonically on Δ and $\Delta \geq 0$, $\mathcal{A}(\boldsymbol{b}_{t-1}^k)$ reaches the maximum 1 when $\Delta = 0$. Next, we assume a beam can be generated indefinitely. In this case, there must be a situation that each $l_{t,i}^k > g_i$. Therefore we have

$$\begin{aligned}\mathcal{A}(\boldsymbol{b}_{t-1}^k) &= 1 - \frac{\sum_{i=1}^n (l_{t,i}^k - g_i)}{\zeta_t^k} \\ &= 1 - \frac{\sum_{i=1}^n l_{t,i}^k - \sum_{i=1}^n g_i}{\zeta_t^k} \\ &= 1 - \frac{\zeta_t^k - \sum_{i=1}^n g_i}{\zeta_t^k} \\ &= \frac{\sum_{i=1}^n g_i}{\zeta_t^k}\end{aligned} \tag{5.10}$$

where $\lim_{\zeta_t^k \to \infty} \mathcal{A}(\boldsymbol{b}_{t-1}^k) = 0$. Therefore, the bound of $\mathcal{A}(\boldsymbol{b}_{t-1}^k)$ is proven to be between 0 and 1. □

In addition to the constraint for $l_{t,i}$, we still desire $l_{T,i} = g_i$ for each token. An ideal hypothesis should have two characteristics simultaneously. Namely, its attention score at the termination is the maximum 1, and its length equals the optimal length. Therefore, we introduce a length reward function to cooperate with the attention score to penalize the situation $l_{T,i} \neq g_i$, which will be discussed at the end of this subsection.

As mentioned before, the total attention score at the decoding step t is defined as:

$$A(\boldsymbol{b}_{t-1}^k) = \prod_{m=1}^{t} \mathcal{A}(\boldsymbol{b}_{m-1}^k) \tag{5.11}$$

Thus, the joint scoring function $J(\cdot)$ is modified from Eq. 5.4:

$$\begin{aligned} J(\boldsymbol{b}_t, \boldsymbol{x}) &= \log p(\boldsymbol{b}_{t-1}^k|\boldsymbol{x}) + \beta \log A(\boldsymbol{b}_{t-1}^k) + \log p(v|\boldsymbol{x}, \boldsymbol{b}_{t-1}^k) \\ &= \sum_{m=1}^{t-1} \Big(\log p(b_m|\boldsymbol{x}, (\boldsymbol{b}_{t-1}^k)_{<m}) + \beta \log \mathcal{A}(\boldsymbol{b}_{m-1}^k) \Big) + \log p(v|\boldsymbol{x}, \boldsymbol{b}_{t-1}^k) + \beta \log \mathcal{A}(\boldsymbol{b}_{t-1}^k) \\ &= J(\boldsymbol{b}_{t-1}^k, \boldsymbol{x}) + \log p(v|\boldsymbol{x}, \boldsymbol{b}_{t-1}^k) + \beta \log \mathcal{A}(\boldsymbol{b}_{t-1}^k), \ t \geq 2 \end{aligned} \tag{5.12}$$

and $J(\boldsymbol{b}_1, \boldsymbol{x}) = \log p(v|\boldsymbol{x}, b_0^k) + \beta \log \mathcal{A}(b_0^k)$, where β is a hyper-parameter to trade-off between the probability and attention score. Similar to $\log p(\boldsymbol{b}_t|\boldsymbol{x})$ in Eq. 5.4, $J(\boldsymbol{b}_t, \boldsymbol{x})$ is also an accumulative score. Consequently, at each step t, we only need compute $p(v|\boldsymbol{x}, \boldsymbol{b}_{t-1}^k)$ and $\mathcal{A}(\boldsymbol{b}_{t-1}^k)$. Compared with Eq. 5.4, the time complexity of each step is only increased by $\mathcal{O}(\mathcal{K})$ as there are $\mathcal{K}$ attention scores. Replacing $\log p(\boldsymbol{b}_t|\boldsymbol{x})$ in Eq. 5.3 by $J(\boldsymbol{b}_t, \boldsymbol{x})$, we can select the top $\mathcal{K}$ beams of each decoding step according to not only the probability distribution conditional on local information $\boldsymbol{b}_{t-1}^k$ but also the score conditional on global information $\boldsymbol{g}$.

Considering the length reward function, the final hypothesis is thus defined as:

$$\boldsymbol{y}^* = \underset{\boldsymbol{y}^k \in \mathcal{Y}}{\operatorname{argmax}} \left(\frac{J(\boldsymbol{y}^k, \boldsymbol{x})}{|\boldsymbol{y}^k| - 1} + \beta\gamma R(\zeta_T^k, Z) \right) \tag{5.13}$$

where $R(\cdot)$ is the length reward function dependent on the optimal length Z and the candidate hypothesis length ζ_T^k. Exactly, ζ_T^k is the total attention that a candidate hypothesis $(y_1^k, \cdots, y_T^k)$ pays to the source and equals $|\boldsymbol{y}^k| - 1$. Besides, the attention score and length reward are weighted by a hyper-parameter γ, and the role of β is to ensure that the two are at the same level relative to the probability. We remove a in Eq. 5.5 as it only adjusts the length preference without really controlling the length.

The design of $R(\cdot)$ could be straightforward – one only need ensure that it increases as ζ_T^k approaches Z, and reaches the maximum only at $\zeta_T^k = Z$. In this paper, we design a step-wise length reward function $\mathcal{R}(\zeta_t^k, Z)$ to better balance the relationship between the attention score and the length reward and make the whole searching process as succinct as beam search. We put the design details of the step-wise length reward in § 5.4, and we regard Eq. 5.13 as the general scoring formulation of global-aware inference.

5.3.2 Predict the Global Attention Distribution

Since the reference is unknown practically, the global attention distribution could only be predicted from the source. We construct an attention-prediction model to learn the relationship between the source tokens and the global attention distribution.

The input of the attention-prediction model is the fixed encoder output $\boldsymbol{E} \in \mathbb{R}^{n \times d}$ of BART or PEGASUS plus learnable positional encodings $\boldsymbol{P} \in \mathbb{R}^{n \times d}$, where d is the dimension of hidden states. The input is fed to a learnable Transformer-encoder to obtain $\widetilde{\boldsymbol{E}} \in \mathbb{R}^{n \times d}$ that is encoded with additional context information, followed by a linear transformation with an exponential function:

$$\widehat{\boldsymbol{g}} = \exp\left(\widetilde{\boldsymbol{E}}\boldsymbol{W}_g + \boldsymbol{b}_g\right) \tag{5.14}$$

where $\widehat{\boldsymbol{g}} \in \mathbb{R}^n$ refers to the prediction of $\boldsymbol{g}$, $\boldsymbol{W}_g \in \mathbb{R}^{d \times 1}$ and $\boldsymbol{b}_g \in \mathbb{R}^n$ are the learnable weights and biases. The exponential function is imposed to ensure $\widehat{\boldsymbol{g}} > 0$. We choose the exponential function for this operation because it is shown stable in the training and testing stage. Given the objective of minimizing the distance between $\widehat{\boldsymbol{g}}$ and $\boldsymbol{g}$, the loss is defined as their Euclidean distance:

$$\mathcal{L} = \|\widehat{\boldsymbol{g}} - \boldsymbol{g}\|_2 \tag{5.15}$$

The predicted optimal length $\widehat{Z}$ is the sum of elements in $\widehat{\boldsymbol{g}}$. Note that the length reward function is not affected no matter whether $\widehat{Z}$ is an integer or not.

5.4 Length Reward

In the first subsection, we will introduce our step-wise length reward function and how it works. In the second part, we will discuss in details how it is designed.

5.4.1 Our Step-wise Length Reward

Our length reward function $\mathcal{R}(\zeta_t^k, Z)$ is to give a length score to a beam at each decoding step, so that its cumulative score $R(\zeta_T^k, Z)$ for a terminated beam could approach the maximum as its length ζ_T^k gets closer to the optimal length Z. Our designed length reward

function is as follows:

$$\mathcal{R}(\zeta_t^k, Z) = -\frac{\left|\zeta_t^k - \frac{Z}{\sqrt{2}} - 0.5\right|}{Z} \tag{5.16}$$

Eq. 5.12 is thus expanded to:

$$\begin{aligned} J(\boldsymbol{b}_t, \boldsymbol{x}) &= \log p(\boldsymbol{b}_{t-1}^k|\boldsymbol{x}) + \beta\left(\log A(\boldsymbol{b}_{t-1}^k) + \gamma\sum_{m=1}^{t}\mathcal{R}(\zeta_m^k, Z)\right) + \log p(v|\boldsymbol{x}, \boldsymbol{b}_{t-1}^k) \\ &= J(\boldsymbol{b}_{t-1}^k, \boldsymbol{x}) + \log p(v|\boldsymbol{x}, \boldsymbol{b}_{t-1}^k) + \beta\left(\log \mathcal{A}(\boldsymbol{b}_{t-1}^k) + \gamma\mathcal{R}(\zeta_t^k, Z)\right),\ t \geq 2 \end{aligned} \tag{5.17}$$

In this case, the final score of $\boldsymbol{y}^k$ is re-defined as

$$\boldsymbol{y}^* = \underset{\boldsymbol{y}^k \in \mathcal{Y}}{\operatorname{argmax}}\ \frac{J(\boldsymbol{y}^k, \boldsymbol{x})}{|\boldsymbol{y}^k| - 1} \tag{5.18}$$

The original $R(\zeta_T^k, Z)$ in Eq. 5.13 has been integrated to $J(\boldsymbol{y}^k, \boldsymbol{x})$, and exactly equals $\frac{\sum_{t=1}^{T}\mathcal{R}(\zeta_t^k, Z)}{|\boldsymbol{y}^k|-1} = \frac{\sum_{t=1}^{\zeta_T^k}\mathcal{R}(t, Z)}{\zeta_T^k}$ which gets the maximum at $\zeta_T^k = Z$. Related theorem and proof are presented in the next subsection.

5.4.2 How It Is Designed

Intuitively, if we want to give a score to the beam length at each step, the score should reach the maximum when the beam length gets the optimality. Accordingly it can be defined as:

$$\mathcal{R}(\zeta_t^k, Z) = -\frac{\left|\zeta_t^k - Z\right|}{Z},\ Z = \sum_{i=1}^{n} g_i \tag{5.19}$$

Since $\zeta_t^k = t$, it is represented as:

$$\mathcal{R}_t = -\frac{|t - Z|}{Z} \tag{5.20}$$

Then, we test whether its cumulative value at the terminated step $R(\zeta_T^k, Z)$ reaches the maximum when $\zeta_T^k = Z$. The cumulative length reward is defined as

$$R(\zeta_T^k, Z) = \frac{\sum_{t=1}^{\zeta_T^k}\mathcal{R}_t}{\zeta_T^k} \tag{5.21}$$

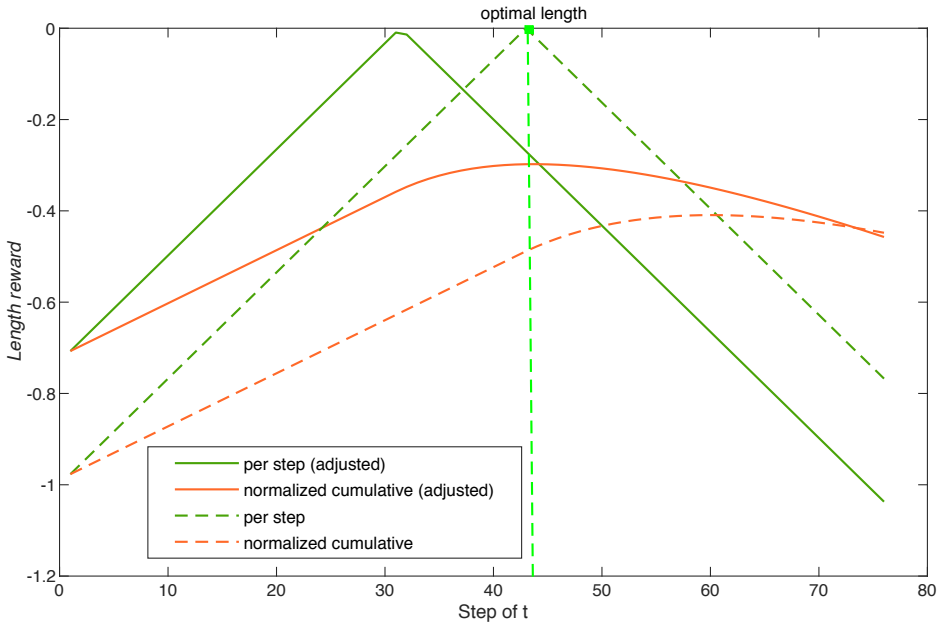


Figure 5.2: Difference between length reward and adjusted length reward.

If we replace ζ_T^k by j, it can be represented as:

$$R_j = \frac{\sum_{t=1}^{j} \mathcal{R}_t}{j} \tag{5.22}$$

Unfortunately, when we plot the curve of R_j as the orange dummy line in Figure 5.2, we find that it reaches the maximum at a length longer than the optimal length. This prompts us to translate $\mathcal{R}_t$ from the green dummy line to the green full line, and the adjusted length reward is designed as:

$$\mathcal{R}_t = -\frac{\left| t - \frac{Z}{\sqrt{2}} - 0.5 \right|}{Z} \tag{5.23}$$

which is the same as Eq. 5.16. After the adjustment, the normalized cumulative length reward R_j (the orange solid line in Figure 5.2) peaks right at the optimal length.

Theorem 2. *Given* $\mathcal{R}_t = -\frac{|t+D-Z|}{Z}$ $(t \geq 1)$ *and* $R_j = \frac{\sum_{t=1}^{j} \mathcal{R}_t}{j}$ *reaches the maximum when* $j = Z$*, then* D *is approximate to* $-\frac{Z}{\sqrt{2}} - 0.5 + Z$*.*

Proof. Since $t \geq 1$ and $t \in \mathbb{Z}^+$, $\mathcal{R}_t$ can be regarded as two arithmetic sequences with difference of $\frac{1}{Z}$ and $-\frac{1}{Z}$, so that it is easy to calculate $\sum_{t=1}^{j} \mathcal{R}_t$. Then R_j could be computed

by:

$$R_j = -\frac{1}{j}\left[\frac{(Z-D-1)(Z-D)}{2Z} + \frac{(j+D-Z)(j+D-Z+1)}{2Z}\right] \tag{5.24}$$

Hence the derivative of R_j^a is:

$$\frac{\partial R_j}{\partial j} = \frac{\frac{2(Z-D)^2-2(Z-D)}{j^2} - 1}{2Z} \tag{5.25}$$

In addition, when $j = Z$, letting $\frac{\partial R_j}{\partial j} = 0$, we have:

$$2D^2 - (4Z-2)D + Z(Z-2) = 0 \tag{5.26}$$

If we solve this equation, we can get:

$$D = \frac{4Z - 2 - \sqrt{8Z^2+4}}{4} \tag{5.27}$$

Omitting the constant item in the radical due to $8Z^2 \gg 4$ under normal conditions, we can get:

$$D = Z - 0.5 - \frac{Z}{\sqrt{2}} \tag{5.28}$$

If we further take D into $\mathcal{R}_t$, we can get Eq. 5.23. □

5.5 Experiments

5.5.1 Setup

Datasets. We evaluate the performance on totally 9 summarization datasets, where 2 datasets (CNN/DM [33], XSUM [23]) with BART [48] and 8 datasets (XSUM [23], BillSum [43], Multi-News [26], NewsRoom [30], WikiHow [44], Reddit TIFU [97], arXiv and PubMed [15]) with PEGASUS [109]. Thereinto, XSUM [23] is a highly abstractive dataset whose summaries are all expressed in a short sentence.

Implementation Details. We adopt a randomly initialized 2-layer transformer-encoder in the attention-prediction model wherethe structure of each layer is the same as the BART-encoder layer. The optimizer is the Adabelief-optimizer [113] with eps $1e-16$, betas $(0.9, 0.999)$, weight decay $1e-4$ and learning rate $2e-5$. The attention-prediction model is trained on the training set for about $50,000$ steps, and checkpoints are saved per

$10,000$ steps to select the best checkpoints on the development set. Since the attention prediction is slightly different from common sequence tagging tasks, we have summarized two notable points after several attempts – the dropout rate should be 0, and a small learning rate is preferred. All experiments are conducted on 3 *RTX 6000*. We include the global-aware inference in the generation code of *HuggingFace transformers* [101]. At the time of evaluation, ROUGE-1, ROUGE-2 & ROUGE-L (R-1, R-2 & R-L) scores [59] are computed from the ROUGE code[3] used by BART [48].

Hyper-parameter Selection. Although the global-aware inference requires two new hyper-parameters γ and β, some original hyper-parameters of beam search, namely length penalty, minimum and maximum length, are omitted. The searching scopes of β and γ are in $\{2, 4, 6, 10, 12, 15, 18, 20\}$ and $\{0, 0.5, 1, 1.5, 2\}$, respectively. According to the numerical tests on the development set, we finally choose $\beta = 12,\ \gamma = 1.5$ for CNN/DM and $\beta = 4,\ \gamma = 0$ for XSUM. As limited improvement could be observed from larger γ's, we recommend $\gamma = 1$ for normal or longer targets. When testing the global-aware inference with PEGASUS [109], we directly use **empirical hyper-parameters** for each dataset, namely $\beta = 4,\ \gamma = 0$ for XSUM and $\beta = 12,\ \gamma = 1$ for other 7 datasets. The beam size $\mathcal{K}$ follows the setups in BART [48] and PEGASUS [109].

5.5.2 Results

Comparison with Beam Search. Beam search is a hard-to-beat baseline which has stood the test of time and proven its superiority in practice for long [67]. In Table 5.2, we compare our global-aware inference to beam search with length regularizations (i.e., α in Eq. 5.5, accompanied with two hard constraints, namely minimum length and maximum length). We strictly follow the hyper-parameter setups of PEGASUS [109] in terms of beam search, while we only adopt empirical hyper-parameters for our method. Even so, significant improvements can be observed on all the data sets, especially when the summary is of normal or longer length.

Comparison with Other Attention Approaches. In this part, we focus on comparing other approaches which also exploit certain attention distributions to improve beam search. The first is the coverage penalty [102, 56]. To enhance its performance in summarization, we replace its preset attention distribution with our predicted global attention distribution. Note that the coverage function can only evaluate the generated sentences at the

[3] https://github.com/pltrdy/files2rouge

Table 5.2: ROUGE F_1 scores of summaries generated by global-aware, in comparison to beam search with length regularizations. Notably, global-aware uses empirical hyper-parameters.

	XSUM			**BillSum**			**Multi-News**			**WikiHow**		
$\mathcal{K} = 8$	**R-1**	**R-2**	**R-L**	**R-1**	**R-2**	**R-L**	**R-1**	**R-2**	**R-L**	**R-1**	**R-2**	**R-L**
Beam search [109]	47.05	24.53	39.33	57.00	39.65	52.70	47.29	18.91	43.31	41.86	19.04	40.40
Global-aware	**47.33**	**24.66**	**39.50**	**58.66**	**40.12**	**53.96**	**47.95**	**19.08**	**43.93**	**42.82**	**19.68**	**41.43**
	Reddit TIFU			**NewsRoom**			**PubMed**			**arXiv**		
	R-1	**R-2**	**R-L**	**R-1**	**R-2**	**R-L**	**R-1**	**R-2**	**R-L**	**R-1**	**R-2**	**R-L**
Beam search [109]	27.55	8.67	22.12	42.05	29.88	38.70	44.25	19.19	41.11	43.82	16.75	39.28
Global-aware	**28.31**	**9.13**	**23.30**	**44.68**	**31.71**	**41.28**	**45.78**	**20.16**	**42.62**	**44.92**	**17.41**	**40.31**

Table 5.3: Comparison with other methods.

	CNN/DM		
$\mathcal{K} = 4$	**R-1**	**R-2**	**R-L**
Beam search [48]	44.12	21.21	40.89
+ Our coverage	44.74	21.69	41.48
+ Repetition penalty [40]	44.11	21.14	40.87
+ Attention masking [9]	**45.54**	**22.24**	**42.44**
Global-aware	45.13	21.77	42.04
	XSUM		
$\mathcal{K} = 6$	**R-1**	**R-2**	**R-L**
Beam search [48]	45.38	22.32	37.15
+ Our coverage	44.54	21.82	36.97
+ Repetition penalty [40]	45.40	22.31	37.13
+ Attention masking [9]	45.35	22.31	37.15
Global-aware	**45.57**	**22.60**	**37.61**

Table 5.4: ORACLE and ablation results.

	CNN/DM		
	R-1	**R-2**	**R-L**
ORACLE global-aware	**51.85**	**28.13**	**48.68**
-w/o length reward	50.46	27.53	47.43
Global-aware	**45.13**	**21.77**	**42.04**
-w/o length reward	44.39	21.58	41.41
-w/o attention score	44.12	21.29	40.91
	XSUM		
	R-1	**R-2**	**R-L**
ORACLE global-aware	**49.50**	26.24	41.13
-w/o length reward	48.92	**26.48**	**41.45**
Global-aware ($\gamma = 1$)	45.44	22.15	37.11
-w/o length reward	**45.57**	**22.60**	**37.61**
-w/o attention score	45.23	21.88	36.73

Table 5.5: Improvements of attention head masking and global-aware on beam search [109] in terms of ROUGE-L F_1 score. Both use empirical setups.

	XSUM	**BillSum**	**Multi-News**	**WikiH**
Attention head masking [9]	-0.31	0.23	0.34	0.10
Global-aware	**0.17**	**1.26**	**0.62**	**1.03**
	Reddit	**NewsRoom**	**PubMed**	**arXiv**
Attention head masking [9]	-0.16	1.24	0.35	0.21
Global-aware	**1.18**	**2.58**	**1.51**	**1.03**

terminal step. Instead of comparing the global-aware inference to the methods [85, 55, 27] that aim to reduce repetition using the dynamic attention distribution, we compare our algorithm with the CTRL repetition penalty [40] which has similar motivation but is more systematical and independent of training. Table 5.3 lists the comparison results against different algorithms. It can be observed that our global-aware approach can improve the performance of beam search stably and fundamentally. We also observe that the attention head masking [9] appears to outperform the global-aware approach on CNN/DM, but it fails to gain any improvement on XSUM. To further show the advantage of the proposed approach, we will take a closer examination on the attention head masking [9] and our proposed approach in the next part.

Further Comparison with Attention Head Masking. First, one should bear in mind that the attention head masking [9] acts on the model instead of beam search, which is opposite to us. Specifically, it selects contents during decoding by disabling partial attention heads for unimportant tokens to decrease the attention to these tokens. According to the reported results presented in Table 5.3, we can see that although attention masking achieves amazing results on CNN/DM, it does fail completely on XSUM. Since hiding unimportant tokens from some heads results in the loss of context information of salient tokens, this would lead to its instability. Thus, it could be ineffective for tasks that require contextual information of the whole source such as XSUM. Taking a further comparison, we deploy the attention head combinations selected for CNN/DM and XSUM to examine its effect on PEGASUS [109]. These comparison results are shown in Table 5.5. Evidently, our method enjoys a robust feature that is able to boost summary inference on various datasets and models even with the same set of hyper-parameters. In contrast, attention masking [9] behaves much sensitive to the changes of models and datasets. Besides, at-

Table 5.6: Generate CNN/DM summaries with XSUM's style.

R1/R2/RL	**Shorter beam search**	**Global-aware†**
F_1 **score**	**43.6**/**20.9**/40.4	**43.6**/20.4/**40.6**
Recall	**48.9/23.4/45.2**	40.4/18.8/37.6
Precision	41.4/19.9/38.3	**50.5/23.8/47.0**

Table 5.7: Generate summaries with corrupted attention distributions.

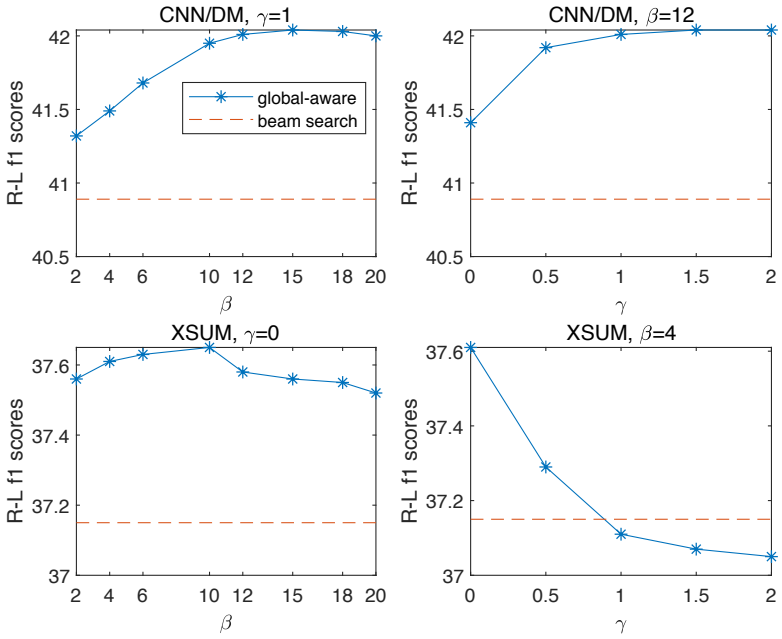
	R-1	**R-2**	**R-L**
Beam search	27.00	12.21	23.88
Global-aware (10K)	28.78	12.82	25.51
Global-aware (100K)	**29.59**	**13.72**	**26.30**

tention masking has to construct its training saliency labels based on the longest common subsequences between a reference summary and the source. This may be hardly achieved in some text generation tasks (e.g. translation) where no common subsequence exists at all. Such drawback presents one main limitation for attention masking.

ORACLE Results and Ablation Study. ORACLE refers to the global-aware inference combined with (true) global attention distribution instead of predicted global attention distribution. The related results have been presented in Table 5.4, and the significant boosting certifies that the proposed method could improve beam search with the global attention distribution. On the other hand, we conduct ablation experiments on ORACLE and global-aware. Both results indicate that length reward plays an important role in generating normal-length text but causes adverse effect on generating text of very short length. Besides, the performance declines significantly when only length reward is applied, due to the fact that sole length reward cannot calibrate beam search step-wise.

Robustness. We intend to examine the robustness of our proposed global-aware algorithm in this and next part. To do so, we substitute the parameters in CNN/DM's attention prediction model to XSUM's to create more abstractive summaries for CNN/DM. For comparison, we set the minimum length of beam search as 0 to allow it to generate shorter summaries. Table 5.6 shows the F_1 score, recall and precision of the shorter beam search and global-aware†. It is surprising that even by using the attention distribution from a different dataset with distinct properties, the proposed global-aware mechanism still manages to generate meaningful summaries with competitive F_1 scores, proving the robustness of

Figure 5.3: Sensitive analysis in the test set.



this algorithm. Moreover, the higher Precision and lower Recall of the global-aware suggest that although information is partially lost, the algorithm still summarizes core information in a concise format, compared to the standard beam search. On the other hand, we exploit a BART model fine-tuned on CNN/DM to generate summaries of NewsRoom directly, and the ROUGE scores of beam search are shown in Table 5.7. Next, we randomly select 10K and 100K samples from the training set and use them to fine-tune the attention-prediction model, where the global-aware improves beam search substantially. The experiment once again validates the robustness of the proposed inference algorithm, as it maintains a reasonably good performance even from learning a corrupted attention distribution from a BART without fine-tuning.

Sensitive Analysis. We further examine the performance robustness with respect to the hyper-parameters. From Figure 5.3, we could see the global-aware inference is always better than beam search in CNN/DM, no matter how its hyper-parameters change. Besides, the performance is less sensitive to the hyper-parameters when $\beta \geq 10$ or $\gamma \geq 1$. While in XSUM, the global-aware could improve beam search stably with $\gamma = 0$, but there is a significant decline when applying length reward. In fact, the attention score

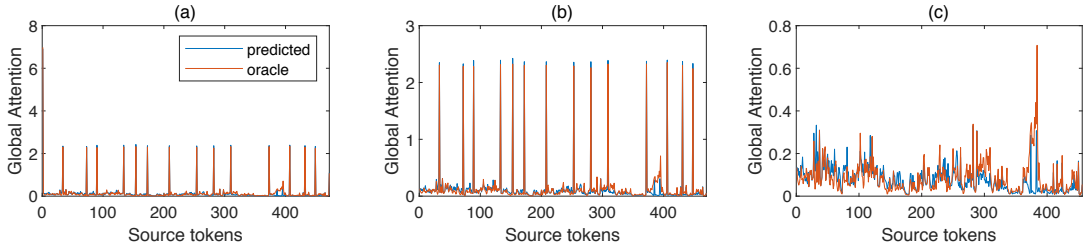


Figure 5.4: Predicted and ORACLE global attention in BART. There are attention distributions of (a) the whole source, (b) the source without the start & end tokens, (c) the source without the start & end tokens and full stops.

favors shorter hypotheses, and the length reward could alleviate the bias. However, if the references of a dataset are already very short such as XSUM, the length reward may lead to a counterproductive effect. Since the setup of CNN/DM is applicable to most datasets, we argue that the global-aware inference is robust to both hyper-parameters in most cases.

5.6 Analysis on Global Attention Distribution

5.6.1 Distribution Law

The distribution law of global attention in BART [48] is shown in Figure 5.4. It is observed that most attention is assigned to the start token and full stops which are not semantically salient, and most of the remaining attention is unevenly allocated to (semantically) overlapped tokens between the source and the reference (i.e., important words). It is worth mentioning that the importance here is no longer a simple binary classification like in [27, 9], but a continuous numerical value decided by the model knowledge learned from data. In general, one should not simply equate the global attention with the word importance, but should be clear that it essentially reflects knowledge learned by the attention model such as the syntactic and semantic structure of sentences. Meanwhile, the distribution law indicates that attention distributions in pre-trained models may not be relevant with the uniform distribution at all. That is to say, it is not reasonable to still use an uniform attention threshold (like the threshold 1 preset in [102, 27]) to regulate the decoding of pre-trained models. Last but not least, our general motivation is to alleviate locality bias by aligning the attention distribution of reference and hypothesis, which does not really care how the global attention is distributed only if it is predictable. However, the proposed

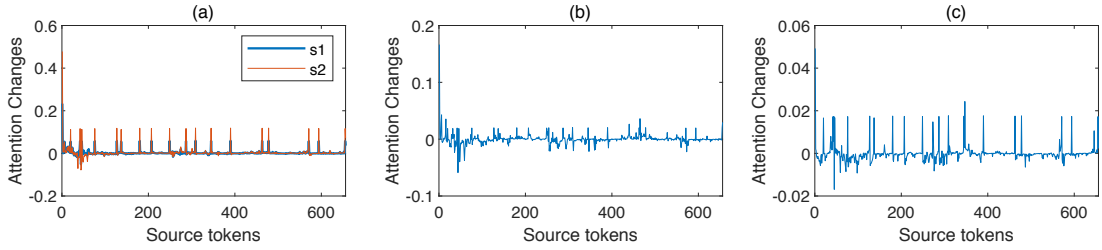


Figure 5.5: Changes of the attention distribution when (a) one word in the reference is replaced by a similar word (s1) and a random word (s2), (b) the sentence order of the reference is shuffled, (c) a factual knowledge in the reference is distorted.

penalty mechanism is indeed insensitive to some distributions, and we will provide more thinking about this in § 5.7 by applying the global-aware inference to translation.

5.6.2 Why It can be Predicted from Source?

Since the ORACLE experiments indicated that the global attention is helpful for decoding, the concern remains if it is predictable using only source. In App. 5.8.1 we will show the predictability empirically, while here we just provide an interesting explanation. In our opinion, the global attention distribution is an interpretable representation of reference which not only has the characteristics of hidden representation but also can be explained by the source tokens. First of all, given the source, the global attention is calculated by the input reference and trained neural network parameters; this is similar to achieving hidden representation. Moreover, like hidden representation, the global attention distribution could also capture the semantic similarity, e.g., replacing a reference word with a semantically similar word leads to slighter attention changes than that with a random word (see Figure 5.5.a). Besides, it is observed from Figure 5.5.b and Figure 5.5.c that global attention is able to represent the sentence order and factual knowledge. On the other hand, the global attention distribution enjoys a distinct characteristic that each of its features can be explained as the degree of attention paid to a source token, which means changes of such representation are regular to some extent. For example, in Figure 5.5.c, we distort a numerical number in the reference to violate the fact stated in the source, and then find that the attention assigned to the actual number originally is most transferred to the special tokens and punctuation. Overall, similar sources should have similar global attention distributions, since similar sources often have similar references and global at-

tention distribution is a representation of reference. Moreover, the global attention and source tokens are in an one-to-one correspondence. Thereby, we argue that it is convenient to use the source to predict the global attention distribution.

5.7 Global-aware Inference in Neural Machine Translation

Though the focus of this paper is Neural Abstractive Summarization, we will share how to enable global-aware inference to improve beam search significantly in NMT using a simple variation.

The gap between NMT and NAS is that NAS is a more-to-less generation task where the global attention values of many words are very low, which makes it relatively easy to trigger punishment at the beginning. As one can see in App. 5.8.4 and the examples provided, many global-aware summaries are different from that of beam search from scratch, implying that the global scoring mechanism is activated in the very early stage. Since translation is a nearly one-to-one task, it is more likely to ignore some locality biases during the beginning stage when the local attention is generally low. Notably, although there is no very low global attention in translation tasks, their global attention distributions are still irrelevant with uniform distributions (here we only consider pre-trained translation models).

Table 5.8: BELU results of WMT16. We perform the blocked global-aware inference with the block length of 10.

$\mathcal{K} = 5$	**BELU**
beam search	37.6
ORACLE global-aware	39.2
+ Blocking	**41.6**
Global-aware	37.8
+ Blocking	**38.1**

To alleviate the gap, a straightforward-but-effective approach is to transform the one-to-one generation task to several more-to-less tasks. We divide the reference into several blocks of equal length and predict the global attention distribution of each block. Taking a reference of 25 tokens as an example, if we set the block length as 10, then there will be three blocks, namely the block of $1-10$ tokens, $11-20$ tokens, and the last 5 tokens. We use the special token b_1 to denote the first block, b_2 to denote the segment composed of the

first and second block, and b_0 to denote the whole reference. When we train the attention-prediction model, these special tokens are concatenated before the source document to tell the predictor which segment it should predict. During inference, we first predict the global attention distribution of b_0 and calculate the optimal length, e.g. 32, then we can figure out there are three more global attention distributions that need be predicted, i.e., b_1, b_2 and b_3. We apply the predicted attention distribution of b_1 to guide the inference at the beginning, and the attention distribution is transformed to that of b_2 when the length of generated sequence is larger than 10 and so on. When the decoding step reaches 30, the attention distribution of b_0 is deployed until all candidate sequences are terminated. Although this blocking operation would increase the difficulty of training and inference, these negative effects are controllable by adjusting the block length.

We use mBART [62] as the neural translation model to examine the blocking operation on WMT16 en-ro sentence-level translation dataset, and related results are presented in Table 5.8. Compared with beam search, our proposed global-aware elevates the BELU score from 37.6 to 37.8, and ORACLE global-aware improves the result to 39.2. The blocking operation further improves global-aware to 38.1, and ORACLE global-aware to 41.6. We also experiment this operation on summarization tasks where minor improvement is observed.

Finally, since the blocking operation would inevitably increase the difficulty of training and inference and is not conducive to the robustness, it may not be the most reasonable direction for improving the global-aware inference in one-to-one tasks. We think there are three following issues that might be worth exploring in the future. 1) Whether the proposed global-aware inference has already been able to improve document-level translation significantly; 2) intuitively, we averaged the global attention distributions of all layers, but it is possible that the global attention in a single layer or a combination of several layers can better guide decoding; 3) whether a more comprehensive global protocol can be found to regulate beam search in a more detailed way.

5.8 Discussions

5.8.1 Predictability of Global Attention

We plot the decline trend of training loss and validation loss on CNN/DM and XSUM to Figure 5.6. It is observed that training loss has been falling significantly, while validation

loss has an upward trend at later epochs.

On the other hand, we randomly select $1,000$ test samples from each dataset and plot their average R^2 scores of the predicted optimal attention distributions in Figure 5.7. For each sample, its R^2 score is equal to $1 - \frac{\sum_{i=1}^{n}(o_i - \hat{o}_i)^2}{\sum_{i=1}^{n}(o_i - \bar{o})^2}$, where $R^2 \in [-\infty, 1]$. A higher R^2 signifies better model fitting, while $R^2 < 0$ means the model is entirely invalid.

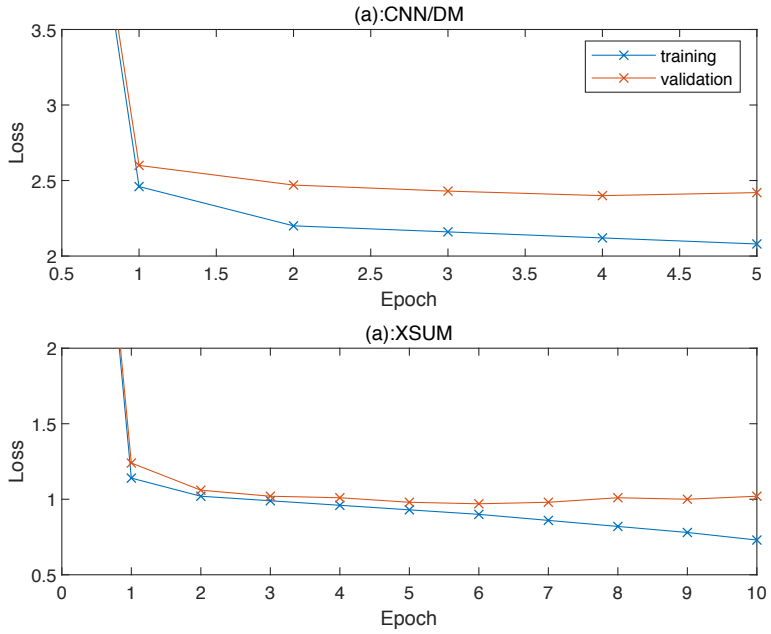


Figure 5.6: Loss trend.

5.8.2 Degradation of Beam Search

It is widely known that text quality generally degrades with increased beam size [16, 70, 67]. The degradation originates from the gap between training and inference, where teacher forcing training only need ensure the local optimality, but beam search pursues the global optimality [100, 80, 110]. The proposed algorithm reduces the gap by informing beam search how the global optimal hypothesis attends to the source so that beam search can be calibrated to pursue a more reasonable global optimum instead of that obtained by teacher forcing training. We enlarge the beam size and find that the global attention distribution can guide beam search to overcome degradation, though the practical improvement might not be significant due to the prediction bias. According to Figure 5.8, the average ROUGE

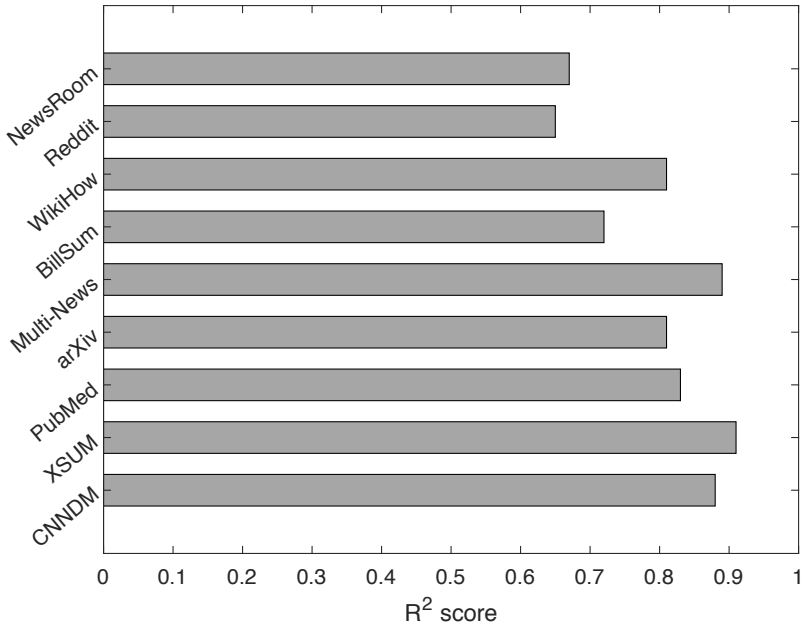


Figure 5.7: Fitting of att-prediction model.

F_1 score of beam search summaries declines sharply as beam size increases. Even if there is a prediction bias, the global-aware inference manages to resist degradation effectively with increased beam sizes. Moreover, a significant boosting in ROUGE scores is observed as beam size increases in the ORACLE global-aware inference, where the true potential of the algorithm is reflected.

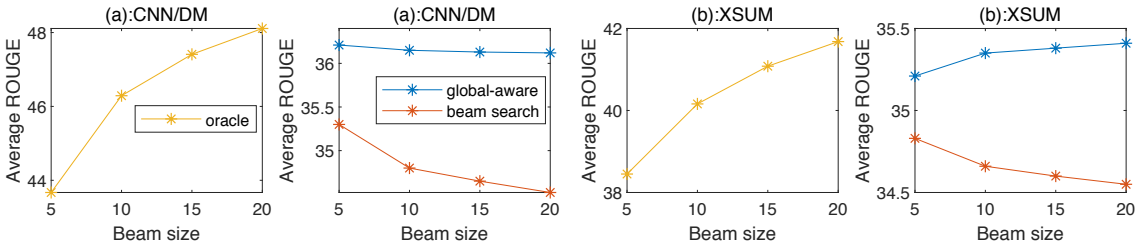


Figure 5.8: Degradation of beam search.

5.8.3 Length Comparison

As shown in Table 5.9, even with empirical hyper-parameters, our method often leads to more concise summary with its length closer to the reference length on many datasets. By contrast, beam search, which conducts searching for length penalty and length constraints, generates many lengthy summaries on some datasets. For example, In CNN/DM, Reddit, and NewsRoom datasets, beam search must extend summaries to obtain higher ROUGE scores, making the generated text more bloated. Overall, in most cases, the global-aware inference could produce summaries of higher quality in a more concise way.

5.8.4 Are the Outputs Different from Beam Search from the Beginning?

One may question whether these hypotheses are just the subsets or extensions of beam search hypotheses. The answer lies in the selection of β. Specifically, the global-aware with a greater β (such as $\beta = 12$ for CNN/DM) generates quite different summaries. We plot the count distribution of discrepancy positions between beam search and global-aware inference in Figure 5.9. It is interesting to find that about one-third of CNN/DM test samples (total count: $11,490$) start to differ at the first word, and most start to differ at the first 5 words. In other words, the global-aware inference generates texts in a way different from beam search that leads to higher scores.

5.8.5 Newly Generated Words

Table 5.10 shows that summaries generated by our method appear more creative. Generally, a greater proportion of novel words implies that the generated text resembles more human-writing, at the same time increases the risk of deviating from reference.

5.8.6 Inference Speed

As presented in Table 5.11, the speed of the global-aware inference is comparable with beam search.

5.8.7 Our Advantages over Predicting Future Rewards

A previous study [51] guides inference by predicting the future automatic matrix score at each decoding step. There are two main disadvantages of training such predictor, (a) there are too many training samples especially when the target sequence is long, making it

Table 5.9: The average summary length. †: consisting of shorter beam search and global-aware†. *: without length reward. In addition to the two datasets, the numbers closer to the reference length are bolded while shorter lengths are underlined. All reference summaries are truncated to 256 tokens.

Token Length	**CNNDM**	**Bill**	**Multi**	**Wiki**	**Reddit**	**NRoom**	**Pub**	**arXiv**	**CNNDM†**	**XSUM***
reference	67.5	176.5	232.9	70.5	26.2	34.4	214.7	191.3	67.5	26.1
beam search	82.3	162.5	219.4	**49.8**	36.5	46.7	**205.8**	**176.1**	78.2	21.8
global-aware	**64.9**	**165.6**	**223.7**	46.2	**20.8**	**31.2**	200.4	166.4	53.2	22.2

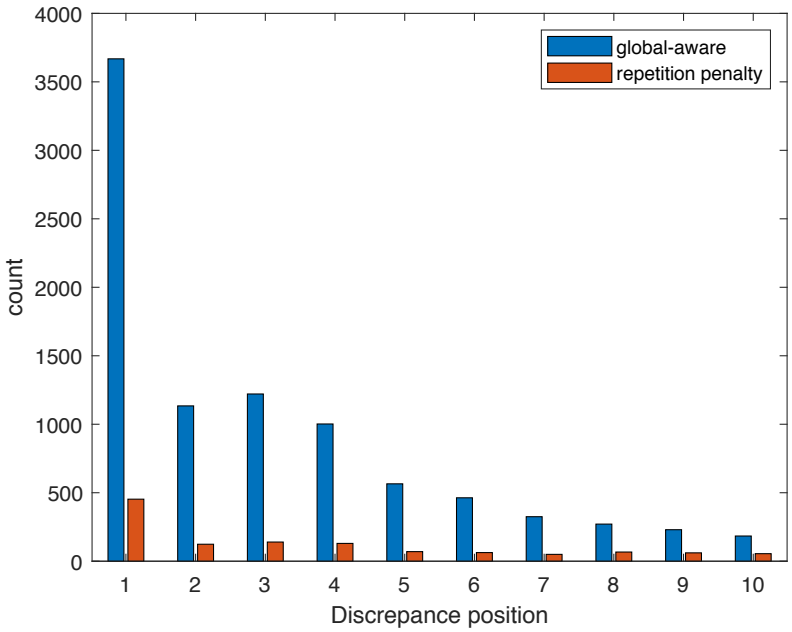


Figure 5.9: The position that the summary starts to differ from the beam search summary.

Table 5.10: The percentage of words in the summary but not in the source.

CNN/DM	**%Novel**
reference	14.8
beam search	4.5
global-aware	**6.7**

Table 5.11: Inference speed

CNN/DM	**Tokens/s**
beam search	**25.6**
global-aware	23.2
repetition penalty	19.1

hard to train; (b) this method forces us to predict every decoding step, greatly slowing the inference speed. By contrast, one appealing benefit of our approach is its high efficiency, since our method only need predict once but can calibrate beam search at each step, which improves beam search from scratch with very little burden of training and inference.

5.8.8 Can Our Method Work Together with Attention Head Masking?

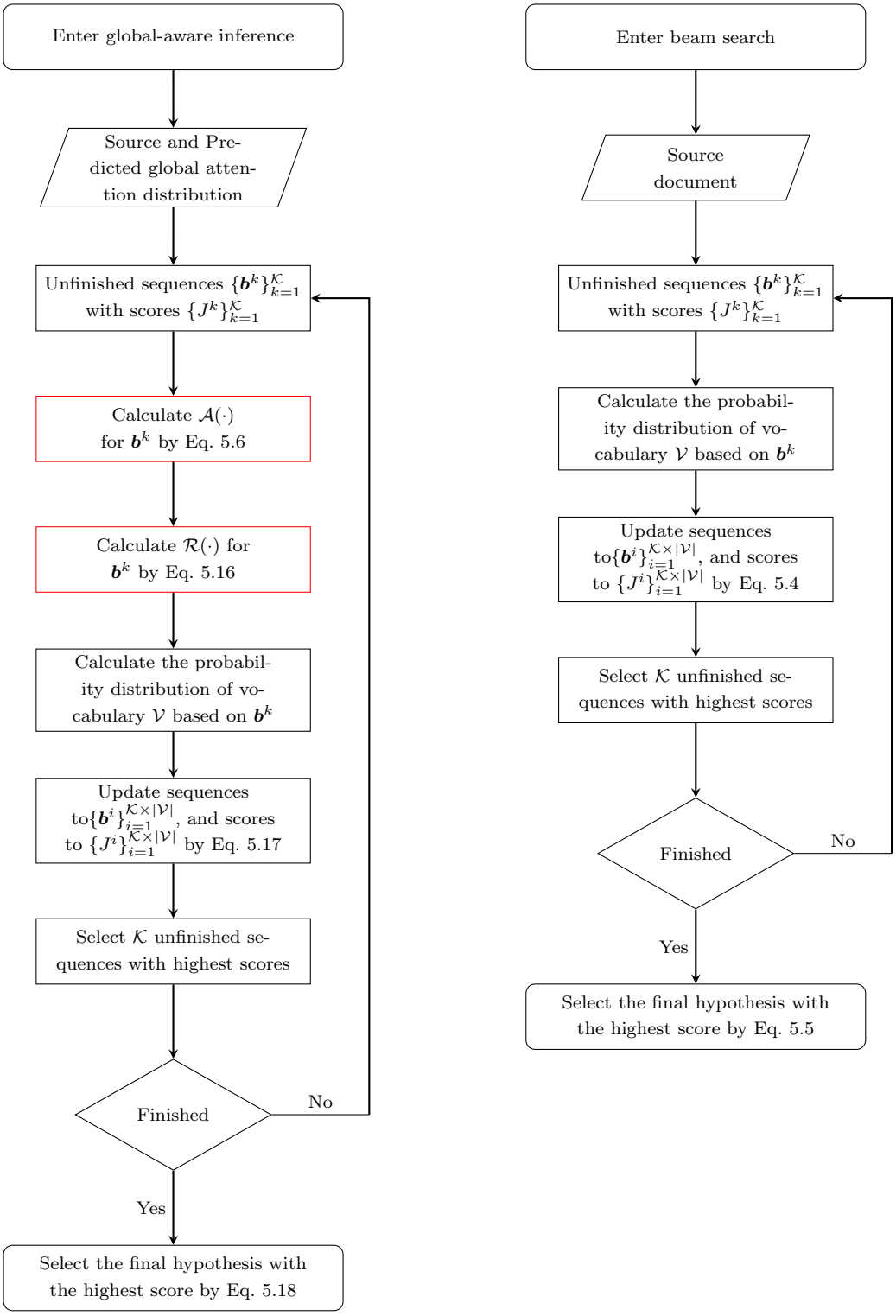
There is no doubt that the answer is yes because the attention head masking [9] still need beam search to generate summaries, and our approach is an alternative to beam search. However, we do not recommend the joint usage of the two due to the following three reasons. First, the roles of the two overlap; second, the loss of context information due to masking will negatively affect the global-aware inference; third, adding attention head masking will impose additional burden on training and hyper-parameter searching.

5.8.9 Can Predicted Attention Distribution still Lead to the Same Theoretical Result?

Through explaining that the global attention and the predicted attention have the same value range, we will simply prove that the predicted attention distribution is also applicable to the theory in the paper. Since each attention value in the global attention distribution is a cumulative value of the attention probability $\in [0, 1]$ given by the reference tokens, the lower bound tends to be 0 and the upper bound is $1 *$ the number of reference tokens. When the reference is infinite, the upper bound is ∞. Therefore, we only need ensure that the predicted attention value is larger than 0 which is constrained by exp.

5.9 Flow Chart: Global-aware vs. Beam Search

-


Enter global-aware inference
Source and Predicted global attention distribution
Unfinished sequences $\{\boldsymbol{b}^k\}_{k=1}^{\mathcal{K}}$ with scores $\{J^k\}_{k=1}^{\mathcal{K}}$
Calculate $\mathcal{A}(\cdot)$ for $\boldsymbol{b}^k$ by Eq. 5.6
Calculate $\mathcal{R}(\cdot)$ for $\boldsymbol{b}^k$ by Eq. 5.16
Calculate the probability distribution of vocabulary $\mathcal{V}$ based on $\boldsymbol{b}^k$
Update sequences to$\{\boldsymbol{b}^i\}_{i=1}^{\mathcal{K}\times|\mathcal{V}|}$, and scores to $\{J^i\}_{i=1}^{\mathcal{K}\times|\mathcal{V}|}$ by Eq. 5.17
Select $\mathcal{K}$ unfinished sequences with highest scores
Finished
No
Yes
Select the final hypothesis with the highest score by Eq. 5.18
Enter beam search
Source document
Unfinished sequences $\{\boldsymbol{b}^k\}_{k=1}^{\mathcal{K}}$ with scores $\{J^k\}_{k=1}^{\mathcal{K}}$
Calculate the probability distribution of vocabulary $\mathcal{V}$ based on $\boldsymbol{b}^k$
Update sequences to$\{\boldsymbol{b}^i\}_{i=1}^{\mathcal{K}\times|\mathcal{V}|}$, and scores to $\{J^i\}_{i=1}^{\mathcal{K}\times|\mathcal{V}|}$ by Eq. 5.4
Select $\mathcal{K}$ unfinished sequences with highest scores
Finished
No
Yes
Select the final hypothesis with the highest score by Eq. 5.5

5.10 Related Works

In the field of text generation, efforts have been made to boost beam search with regards to the attention distribution. For instance, some studies engage the attention distribution to penalize the situation where sources fail to be fully covered in translation tasks [102, 56], while others [27, 85, 55] incorporate dynamic attention distributions to evade tokens that have been highly regarded to reduce repetition. However, none of the aforementioned studies attempts to apply the global attention distribution to acquire the knowledge that the level of attention should be allocated to each token from the reference. Further, the existing score functions used by those studies are rather different from the proposed global scoring mechanism.

We summarize the characteristics of each score function in Table 5.12, where $\alpha_{t,i}$ refers to the cross attention that the t_{th} generated token pays to the i_{th} source token. Previous works [102, 56, 85, 55, 27] design the score function based on some assumptions which may be invalid. Specifically, [102], [56] and [27] use the same constant (1 or β) to constrain all source tokens, based on the assumption that the minimum/maximum attention allocated to each token is the same. [85] and [55] assume the attention assigned before t should be lower than the attention of t. Obviously, both assumptions may often not hold in practice. By contrast, our global scoring mechanism does not depend on any assumption but follows the nature of generation, i.e., the growing local attention should not surpass the global attention during inference and exactly reach it at the termination.

In addition to the attention distribution, other techniques are developed to improve beam search in terms of length bias [105], diversity-less [93], vapidity [36] and degradation [70, 16]. These methods are not included for comparison because they are not suitable for summarization [105, 36], or do not aim to enhance beam search as the main purpose [93, 70, 16]. Besides, we argue that these patches to beam search are supposed to be hard for improving the performance stably and fundamentally (as shown in Table 5.3 given by our global-aware method) because they fail to specify what a final hypothesis should look like and are easy to trap into local optimums.

5.11 Conclusions

Beam search tends to fall into local optimums due to the lack of global information during inference. To calibrate beam search on the premise of global awareness, this study proposes

Table 5.12: Scoring Mechanism Comparison. Training: whether the score function is a part of loss function of the seq-to-seq model. Inference: whether the score function is a part of the hypothesis score. Desired Situation: the score function reaches the optimum in this situation. †: the function described in the paper [27] is only activated at the termination, but their code added a step-wise modification on it.

Article	**Score Function**	**Training**	**Inference**	**Step-wise**	**Desired Situation**
[102]	$\sum_{i=1}^{n} \log \min\left(\sum_{t=1}^{T} \alpha_{t,i}, 1\right)$	✗	✓	✗	$\sum_{t=1}^{T} \alpha_{t,i} \geq 1$
[56]	$\sum_{i=1}^{n} \log \min\left(\sum_{t=1}^{T} \alpha_{t,i}, \beta\right)$	✗	✓	✗	$\sum_{t=1}^{T} \alpha_{t,i} \geq \beta$
[85]	$-\sum_{i=1}^{n} \min\left(\alpha_{t,i}, \sum_{m=1}^{t-1} \alpha_{m,i}\right)$	✓	✗	✓	$\sum_{m=1}^{t-1} \alpha_{m,i} < \alpha_{t,i}$
[55]	$1 - \sum_{i=1}^{n} \min\left(\alpha_{t,i}, \sum_{m=1}^{t-1} \alpha_{m,i}\right)$	✓	✓	✓	$\sum_{m=1}^{t-1} \alpha_{m,i} < \alpha_{t,i}$
[27]	$n - \sum_{i=1}^{n} \max\left(\sum_{m=1}^{t} \alpha_{t,i}, 1\right)$	✗	✓	✓†	$\sum_{m=1}^{t} \alpha_{m,i} \leq 1$
Ours	$\frac{\sum_{i=1}^{n} \min\left(\sum_{m=1}^{t} \alpha_{m,i}, g_i\right)}{\sum_{i=1}^{n} \sum_{m=1}^{t} \alpha_{m,i}} + \gamma \mathcal{R}(\zeta_t^k, Z)$	✗	✓	✓	$\sum_{m=1}^{t} \alpha_{m,i} < g_i, t < T$ **and** $\sum_{m=1}^{T} \alpha_{m,i} = g_i$

a predictable protocol to stipulate how a global optimal hypothesis should attend to source tokens. By training a simple prediction model of the global attention distribution, a novel global scoring mechanism is then developed to regulate the inference on a step-wise basis. Our experiment shows that the proposed global-aware beam search generates higher-quality summaries, relative to beam search and other counterparts of the similar nature. Besides, the proposed algorithm is proven robust in generating meaningful texts even with corrupted attention distributions, implying its potential to cooperate with user-defined global attention distributions. We plan to focus our future study on generalizing the global-aware inference to a broader range of text generation tasks, including not only text2text but also image caption [96], multimodal summarization [52], graph2text [41] and data2text [76].

5.12 Generation Examples

Examples are presented in the following pages. We sample some good cases and some bad cases of global-aware inference by the ROUGE-1 F_1 score.

Table 5.13: Generated summaries on CNN/DM sampled by ROUGE-1 F_1 score.

Reference	The ramp agent fell asleep in the plane's cargo hold . He can no longer work on Alaska Airlines flights .
Beam search (R-1: 25.05)	"Iḿ inside a plane and I feel like itś up moving in the air," the caller says . "There could be a person in there so weŕe going to come back around," the pilot tells air traffic control . The ramp agent is an employee of Menzies Aviation, a contractor for Alaska Airlines . The airline says the man has been permanently banned from working on planes .
Global-aware (R-1: 43.29)	A ramp agent fell asleep in the cargo hold of an Alaska Airlines flight . The crew and passengers reported unusual banging from the belly of the Boeing 737 . The pilot radioed air traffic control and said he would make an emergency landing .
Global-aware† (R-1: 50.81)	A ramp agent falls asleep in the cargo hold of an Alaska Airlines plane . The plane makes an emergency landing in Seattle . The man has been banned from working on Alaska Airlines planes .
Reference	Three people killed; five wounded in attack on attorney general's office in Balkh province . Staff and civilians have been rescued as gunmen engaged Afghan security forces .
Beam search (R-1: 38.78)	A group of armed assailants stormed into the attorney general's office in Balkh province, northern Afghanistan . Two police officers and a security guard of the provincial attorney general's office were among the dead . Most staff members and civilians have been rescued, but an exchange of fire between Afghan security forces and the assailants is ongoing .
Global-aware (R-1: 54.00)	Three people were killed and five others were wounded in an attack on the attorney general's office . An exchange of fire between security forces and the assailants is ongoing ."
Global-aware† (R-1: 44.00)	Gunmen storm attorney general's office in northern Afghanistan . Three people killed, five others wounded .
Reference	Six young Minnesotans conspired to sneak into Syria and join ISIS "by any means necessary," prosecutors say . The men, ages 19 to 21, were arrested Sunday . They plotted for 10 months, the U.S. attorney for the District of Minnesota says .
Beam search (R-1: 47.68)	The group of friends, ages 19 to 21, were arrested Sunday . They plotted for 10 months, U .S . Attorney Andrew M . Luger says . Recruiting for the ISIS terrorist network is a problem in Minnesota's Somali community . Four of the men who were arrested appeared in federal court on Monday but did not enter a plea .
Global-aware (R-1: 61.97)	Six men, ages 19 to 21, were arrested Sunday . They plotted for 10 months, U .S . attorney says . Recruiting for the ISIS terrorist network is a problem in Minnesota's Somali community .
Global-aware† (R-1: 38.92)	The six men were arrested Sunday . They are accused of plotting to join ISIS in Syria . Recruiting for ISIS is a problem in Minnesota's Somali community .

Reference	Raul Reyes: In seeking Latino vote, Marco Rubio his own worst enemy on two key issues: immigration reform, Cuba relations . He says on health care, climate change and other issues, he breaks from Latinos' positions. Polls show they don't favor him .
Beam search (R-1: 35.71)	Ruben Navarrette: Marco Rubio has been his own worst enemy on immigration and Cuba . Navarrette: Rubio has embraced a typical conservative approach to immigration . He says Rubio's political philosophy will be a tough sell to Hispanics . Navarrette: Unless he can offer new ideas, his climb to the Republinomination will be steep .
Global-aware (R-1: 36.50)	Ruben Navarrette: Marco Rubio is running for the GOP presidential nomination . Navarrette: Rubio has been his own worst enemy on immigration reform and Cuba relations . He says Rubio has embraced a typical conservative approach to immigration . Navarrette: Rubio's political philosophy will be a tough sell to Hispanics .
Global-aware† (R-1: 40.11)	Ruben Navarrette: Marco Rubio is running for the GOP presidential nomination . Navarrette: Rubio has been his own worst enemy on immigration reform and Cuba relations . He says Rubio has embraced a typical conservative approach to immigration .
Reference	Police: Yuhei Takashima, 64, says he had sex with girls as young as 14 in Philippines . Officers seize nearly 150,000 photos that the former principal kept of his activities .
Beam search (R-1: 52.48)	Yuhei Takashima, 64, says he paid for sex with more than 12,000 women in the Philippines . Police seized 147,600 photos that Takashima took of his activities over the years . The ages of the prostitutes he hired ranged from 14 to over 70, he says .
Global-aware (R-1: 37.33)	Police say Yuhei Takashima, 64, paid for sex with more than 12,000 women . He took 147,600 photos of his activities over more than a quarter of a century, police say .
Global-aware† (R-1: 49.28)	Police: Yuhei Takashima, 64, paid for sex with more than 12,000 women in the Philippines . He took 147,600 photos of his activities, police say .
Reference	Kabul faces uncertain future as NATO presence – and the money that came with it – fades away . Interpreters are out of work, NATO trucks sit idle on roads, restaurants are empty .
Beam search (R-1: 21.95)	Kabul is a city swollen in size but shrunken in scope, anxiously awaiting what comes next . Vast supply chains once kept 120,000 troops fed and watered . Now the bases are gone, and the trucks that once supplied millions sit still . One road, forever pot-holed in the past decade, is now being covered over by Afghans .
Global-aware (R-1: 16.12)	Afghanistan's capital, Kabul, has been transformed since NATO arrived in 2001 . The city's population has swelled to five times what it was when the U .S . arrived . But the city has also shrunk in scope, with many waiting to see what comes next .
Global-aware† (R-1: 20.39)	"Kabul, the capital of Afghanistan, has been transformed by the U .S . troop withdrawal . Roads are lined with the detritus of America's war here . Vast supply chains once kept 120,000 troops fed and watered .

Table 5.14: Generated summaries on BillSum sampled by ROUGE-1 F_1 score.

Reference	Directs the Secretary of Health and Human Services to establish a process under which a physician may request, in writing from a carrier, assistance in addressing questionable codes and procedures under the medicare program. Sets forth provisions concerning: (1) policy development regarding evaluation and management guidelines; and (2) medicare overpayments.
Beam search (R-1: 27.61)	Prohibits the Health Care Financing Administration (HCFA) from implementing any new evaluation and management guidelines under the Medicare program unless HCFA: (1) has provided for an assessment of the proposed guidelines by physicians; (2) has established a plan for improving participation of physicians; (3) has carried out a minimum of four pilot projects in at least four different HCFA regions to test such guidelines; and (4) finds that the objectives will be met in the implementation of such guidelines . Requires each pilot project to: (1) be of sufficient length to allow for preparatory physician and carrier education, analysis, and use and assessment of potential E&M guidelines; and (2) be conducted, throughout the planning and operational stages of the project, in consultation with national and State medical societies .
Global-aware (R-1: 40.00)	Directs the Secretary of Health and Human Services to establish a process under which a physician may request, in writing from a carrier, assistance in addressing questionable codes and procedures under the Medicare program . Requires the carrier to respond in writing within 30 business days and allows a written statement to be used as proof against a future audit or overpayment under the Medicare program . Requires the Administrator of the Health Care Financing Administration to restore the toll-free telephone hotline so that physicians may call for information and questions about the Medicare program .Prohibits the Secretary from implementing any new evaluation and management guidelines under the Medicare program, unless the Health Care Financing Administration: (1) has provided for an assessment of the proposed guidelines by physicians; (2) has established a plan for improving participation of physicians; (3) . . .
Reference	Medicare Part D Drug Class Protection Act of 2007 - Amends part D (Voluntary Prescription Drug Benefit Program) of title XVIII (Medicare) of the Social Security Act to require that Medicare prescription drug plans using formularies cover all drugs included in six specified therapeutic categories. Sets forth special requirements for reconsideration of coverage determinations, and appeals for drugs included in such categories. Establishes reporting requirements for drugs in these categories.
Beam search (R-1: 73.59)	Amends part D (Voluntary Prescription Drug Benefit Program) of title XVIII (Medicare) of the Social Security Act to require prescription drug formularies to cover all drugs in six specified therapeutic categories and classes .
Global-aware (R-1: 68.45)	Amends part D (Voluntary Prescription Drug Benefit Program) of title XVIII (Medicare) of the Social Security Act to require that the prescription drug formulary include, subject to specified requirements, all or substantially all drugs in each of six specified therapeutic categories of covered Medicare part D drugs: (1) Immunosupessants; (2) Antidepressants; (3) Anticonvulsants; and (4) Antiretrovias . Provides for special coverage of drugs included in specified therapeutic categories during determinations, reconsiderations, and appeals .

Reference	Reduce Expenditures in Nuclear Infrastructure Now Act or the REIN-IN Act - Prohibits the obligation or expenditure of funds authorized to be appropriated to the Department of Defense (DOD) for FY2014-FY2023: (1) for the research, development, test, and evaluation (RDT&E) or procurement of a long-range penetrating bomber aircraft; (2) to procure an SSBN-X submarine (and prohibits the use of such funds for FY2024 and thereafter to procure more than eight such submarines); or (3) for the RDT&E or procurement of a new intercontinental ballistic missile (ICBM). Prohibits the obligation or expenditure of funds authorized to be appropriated for FY2014 or thereafter for DOD or the Department of Energy (DOE): (1) to make the F-35 Joint Strike Fighter aircraft capable of carrying nuclear weapons; (2) until the Secretary of Defense and the Secretary of Energy jointly certify that the total cost of the B61 life extension program has been reduced to not more than $5 billion; (3) for the W78 life extension program; (4) for the mixed oxide fuel fabrication facility project; (5) to replace the chemistry and metallurgy research building at Los Alamos National Laboratory, Los Alamos, New Mexico; or (6) for the uranium processing facility at the Y-12 National Security Complex, Oak Ridge, Tennessee. Prohibits Navy forces, beginning in FY2020, from including more than eight operational ballistic-missile submarines available for deployment . Prohibits the . . .
Beam search (R-1: 56.48)	Reduce Expenditures in Nuclear Infrastructure Now Act or the REIN-IN Act Prohibits using funds appropriated to the Department of Defense (DOD) for FY2014-FY2023: (1) for the research, development, test, and evaluation or procurement of a long-range penetrating bomber aircraft; (2) to make the F-35 Joint Strike Fighter aircraft capable of carrying nuclear weapons; (3) until the Secretary of Defense and the Secretary of Energy jointly certify that the total cost of the B61 life extension program has been reduced to not more than $5 billion; (4) for the W78 life extension program; (5) for the reduction of nuclear-armed submarines, beginning in FY2020; or (6) for the National Nuclear Security Administration for FY2024 .
Global-aware (R-1: 67.03)	Reduce Expenditures in Nuclear Infrastructure Now Act or the REIN-IN Act Prohibits the obligation or expenditure of funds authorized to be appropriated to the Department of Defense (DOD) for FY2014-FY2023: (1) for the research, development, test, and evaluation or procurement of a long-range penetrating bomber aircraft; (2) to make the F-35 Joint Strike Fighter aircraft capable of carrying nuclear weapons; (3) until the Secretary of Defense and the Secretary of Energy jointly certify that the total cost of the B61 life extension program has been reduced to not more than $5 billion; (4) for the W78 life extension program; (5) for the reduction of Nuclear-Armed Submarines, beginning in FY2020; (6) for the SSBN-X submarines; or (7) for the mixed oxide fuel fabrication facility project . Prohibits the obligation or expenditure of funds authorized to be appropriated for FY2014-FY2023 for DOD: (1) to maintain more than 200 intercontinental ballistic missiles (ICBMs), (2) to maintain more than 250 submarine-launched ballistic missiles, (3) for the research, development, test, and evaluation or procurement of a new ICBM, or (4) for the uranium processing facility at the Y-12 National Security Complex, Oak Ridge, Tennessee . Prohibits the . . .

Table 5.15: Generated summaries on arXiv sampled by ROUGE-1 F_1 score.

Reference	in this paper , we develop a structure - preserving discretization of the lagrangian framework for electromagnetism , combining techniques from variational integrators and discrete differential forms . this leads to a general family of variational , multisymplectic numerical methods for solving maxwell s equations that automatically preserve key symmetries and invariants . in doing so , we demonstrate several new results , which apply both to some well - established numerical methods and to new methods introduced here . first , we show that yee s finite - difference time - domain (fdtd) scheme , along with a number of related methods , are multisymplectic and derive from a discrete lagrangian variational principle . second , we generalize the yee scheme to unstructured meshes , not just in space but in @xmath0-dimensional spacetime . this relaxes the need to take uniform time steps , or even to have a preferred time coordinate at all . finally , as an example of the type of methods that can be developed within this general framework , we introduce a new asynchronous variational integrator (avi) for solving maxwell s equations . these results are illustrated with some prototype simulations that show excellent energy and conservation behavior and lack of spurious modes , even for an irregular mesh with asynchronous time stepping .
Beam search (R-1: 37.82)	the yee scheme is one of the most successful numerical methods used in the field of computational electromagnetics , particularly in the area of microwave problems . although it is not a high - order " method , it is still preferred for many applications because it preserves important structural features of maxwell s equations that other methods fail to capture . in this paper , we show that these desirable properties are direct consequences of the variational and discrete differential structure of the yee scheme , which mirrors the geometry of maxwell s equations . moreover , we show how to construct other variational methods that , as a result , share these same numerical properties , while at the same time applying to more general domains .
Global-aware (R-1: 45.96)	the yee (finite - difference time - domain) scheme is one of the most successful numerical methods used in the field of computational electromagnetics , particularly in the area of microwave problems . in this paper , we show that these desirable properties are direct consequences of the variational and discrete differential structure of the yee scheme , which mirrors the geometry of maxwell s equations . moreover , we show how to construct other variational methods that , as a result , share these same numerical properties , while at the same time applying to more general domains . the key symmetry of maxwell s equations is not rotational or translational symmetry , as in mechanics , but a differential gauge symmetry . without taking additional care to preserve this gauge structure , even variational integrators can not be expected to capture the geometry of maxwell s equations . this differential / gauge structure also turns out to be important for the numerical performance of the method , and is one of the hallmarks of the yee scheme . this differential / gauge structure also turns out to be important for the numerical performance of the method , and is one of the hallmarks of the ye . . .

Reference	we present a new general mechanism to generate curvature perturbations after the end of the slow - roll phase of inflation . our model is based on the simple assumption that the potential driving inflation is characterized by an underlying global symmetry which is slightly broken .
Beam search (R-1: 36.88)	most inflationary models assume that there is only one scalar field involved in the dynamics of inflation . in this work we point out a qualitatively new effect that might arise if one relaxes the assumption of a single dynamical field . in a multi - field scenario in which the inflationary potential is characterized by a broken symmetry , the quantum fluctuations generated during the inflationary stage represent fluctuations in the initial conditions for the dynamics of the inflaton in the subsequent stage , thus implying that the background dynamics after the slow - roll phase has ended will differ in different regions of the universe . since the background fields are coupled to the other fields into which they decay , the fluctuations generated during the slow - roll phase will affect the subsequent decay process . the present work , assuming that the inflaton decay into other fields through the non - perturbative process of preheating , is then aimed to understand whether isocurvature inflaton fluctuations , generated during the slow - roll stage , can lead to perturbations of the background metric through variations of the preheating efficiency . while the generation of curvature perturbations during the stages following the slow - roll phase has already been considered in some works , the ...
Global-aware (R-1: 53.46)	we show that in a multi - field scenario in which the inflationary potential is characterized by a broken symmetry , quantum fluctuations generated during the slow - roll stage of inflation may affect the efficiency of the subsequent preheating process . we obtain a general formula for the curvature perturbations generated from an inhomogeneous preheating efficiency related to the quantum fluctuations produced during inflation . the present work is then applied to the case of a broken @xmath0 symmetry .
Reference	in this work we investigate an influence of decoherence effects on quantum states generated as a result of the cross - kerr nonlinear interaction between two modes . for markovian losses (both photon loss and dephasing) , a region of parameters when losses still do not lead to destruction of non - classicality is identified . we emphasize the difference in impact of losses in the process of state generation as opposed to those occurring in propagation channel . we show moreover , that correlated losses in modern realizations of schemes of large cross - kerr nonlinearity might lead to enhancement of non - classicality .
Beam search (R-1: 47.48)	decoherence due to losses is a main obstacle to implementations of schemes using kerr and cross - kerr nonlinearities . in our work we discuss an aspect of the decoherence which has been seldom discussed when considering an influence of losses on states generated via kerr nonlinearity . namely , we address losses arising in the process of generation and not due to propagation of the generated state via lossy channels . we concentrate our attention on a feature that might be quite significantly pronounced in modern schemes of generating large kerr nonlinearity : the modal loss can be strongly correlated . indeed , the modes occupy the same volume and interact with the same physical systems which form the reservoirs . also , if the kerr - nonlinearity scheme implies a sufficiently strong dispersive coupling of light modes to emitters , then coupling of these emitters to dissipative reservoirs might also appear to be quite strong . as a result , this would mean strongly correlated modal losses . on a number of examples we show how the correlated loss enhances and creates intermodal correlations and even entanglement , and might lead to generation of entangled states quite ...
Global-aware (R-1: 41.50)	decoherence of quantum states generated via kerr and cross - kerr nonlinearities is discussed . it is shown that modal loss can be strongly correlated , i .e . , it can be decoupled from uncorrelated reservoirs . this strongly correlated loss enhances and creates intermodal correlations and even entanglement , and might lead to generation of entangled states quite different from those generated in the same scheme without loss . the strongly correlated loss can result in the significantly enhanced robustness of the generation scheme .

Table 5.16: Generated summaries on Multi-News sampled by ROUGE-1 F_1 score.

Reference	– Joy Milne noticed a small difference in her husband just before he turned 40 . "His smell changed," the Scottish woman tells the BBC . "It wasńt all of a sudden . It was very subtle—a musky smell ." He was diagnosed with Parkinsoń́s disease six years later and died in June at age 65 . Milne found herself supporting the Parkinsoń́s UK charity, where she noticed others had the same distinct smell . She happened to mention her observation to scientists at a talk, and they later put her to the test, presenting her with T-shirts worn by six people with Parkinsoń́s and six without . "She got the six Parkinsoń́s, but then she was adamant one of the ćontrolśubjects had Parkinsoń́s," says a scientist . Researchers were impressed and dubbed her a "super-smeller," per AFP . Then eight months later, that one control subject returned to tell scientists he had just been diagnosed with Parkinsoń́s . Because of Milneś sniffer, scientists now theorize . . .
Beam search (R-1: 48.93)	– Joy Milneś husband, Les, was diagnosed with Parkinsoń́s disease at age 45—six years before he died in June . "His smell changed and it seemed difficult to describe . It wasńt all of a sudden . It was very subtle—a musky smell," the Australian woman, who lives in Perth, tells the BBC . "I got an occasional smell ." After joining the charity Parkinsoń́s UK, she met people with the same odor, and researchers at Edinburgh University decided to test Milneś ability to detect the disease using t-shirts worn by six people with Parkinsoń́s and six without . "Her job was to tell us who had Parkinsoń́s and who didńt," says Dr . Tilo Kunath . "Her accuracy was 11 out of 12 . We were quite impressed ." Six of the Parkinsoń́s patients wore the shirts for a day, while the other six wore them for a week . Milne correctly identified six of the Parkinsoń́s patients and six of the non- Parkinsoń́s patients, but she was "adamant" that one of the non-Parkinsoń́s subjects had the disease, Kunath says . Eight months later, he told Kunath hé been diagnosed with Parkinsoń́s . . .
Global-aware (R-1: 55.18)	– When Joy Milneś husband was diagnosed with Parkinsoń́s disease six years before his death in June, her sense of smell told her something was amiss . "His smell changed and it seemed difficult to describe . It wasńt all of a sudden . It was very subtle—a musky smell," she tells the BBC . After joining the charity Parkinsoń́s UK, she met people with the same odor . She mentioned it to scientists at the University of Edinburgh, and they decided to test her—and were "quite impressed," says one of the scientists . Milne correctly identified six people with Parkinsoń́s and six without it using T-shirts that had been worn by six people with Parkinsoń́s and six without it . Her accuracy was 11 out of 12 . "She got the . . .

Reference	– President Obama devoted his weekly radio address today to the Connecticut school shooting, asking Americans to pray for the victims and their families . "Most of those who died were just young children with their whole lives ahead of them," he said, echoing yesterdayś news conference . "And every parent in America has a heart heavy with hurt ." Obama reiterated that itś time to "come together to take meaningful action" to prevent future shootings, thought he stopped short of specifics, reports Reuters . (USA Today has the full transcript here .) Advocates of stricter gun control already have begun to step up pressure, including New York City Mayor Michael Bloomberg . "We have heard all the rhetoric before," he said yesterday . "What we have not seen is leadership—not from the White House and not from Congress . That must end today ."
Beam search (R-1: 59.23)	– "Most of those who died were just young children with their whole lives ahead of them . And every parent in America has a heart heavy with hurt," President Obama said today in his weekly radio address, reports USA Today . "There are families in Connecticut who cant́ do that today . And they need all of us now . All of us can extend a hand to those in need—to remind them that we are there for them, that we are praying for them ." Obama again stopped short of calling for tougher gun laws, reports Reuters . "We grieve for the families of those we lost," he said . "And we keep in our prayers the parents of those who survived ."
Global-aware (R-1: 48.81)	– President Obama today repeated his call for "meaningful action" to reduce gun violence in the wake of the Connecticut school shooting, reports Reuters . "We grieve for the families of those we lost," he said in his weekly radio address . "And we keep in our prayers the parents of those who survived ." Obama didnt́ call for stricter gun laws, notes USA Today, but he did urge a national conversation . "This weekend, Michelle and I are doing what I know every parent is doing—holding our children as close as we can and reminding them how much we love them," he said . "There are families in Connecticut who cant́ do that today . And they need all of us now ."
Reference	– Jesse Jackson Jr . wept in court today, apologized for using $750,000 in campaign funds as his personal piggy bank, and then took his lumps: 30 months in federal prison and three yearsṕrobation, reports the Chicago Tribune . "I am the example for the whole Congress," he said . "I understand that . I didn't separate my personal life from my political activities, and I couldn't have been more wrong ." The former Illinois congressman added an unusual request: He wants to serve his time in Alabama because "I want to make it a little inconvenient for everybody to get to me ." His wife, Sandi, then got sentenced to 12 months in prison for filing false tax returns about their income, reports AP . She wept, too . "My heart breaks every day with the pain this has caused my babies," she said, referring to their two children, ages 9 and 13 . Because of the kids, the judge will . . .
Beam search (R-1: 52.74)	– Former congressman Jesse Jackson Jr . and his wife, former Chicago alderman Sandi Jackson, were sentenced to prison today, reports the Chicago Sun-Times . Jesse Jackson Jr . got 2 1/2 years for using campaign funds for personal expenses, while his wife got one year for filing false tax returns . The Jacksons had pleaded for mercy for each other in court . Jesse Jackson Jr . "I didn't separate my personal life from my political life, and I couldn't be more wrong," he said . "I take responsibility for my actions ." Sandi Jackson said she needed to be with her children . "I ask to continue to provide for my children," she said . "To take the mother away . . . would be an unbearable burden on these two children ."
Global-aware (R-1: 47.29)	– Former congressman Jesse Jackson Jr . and his wife, former Chicago alderman Sandi Jackson, were sentenced to prison today, reports the Chicago Sun-Times . Jackson was sentenced to 2 1/2 years for using campaign funds for personal expenses, while his wife got a year for filing false tax returns . The Jacksons had pleaded for mercy for each other in court, but the judge didnt́ seem swayed, reports AP . "There may be blurred lines for congressmen to follow when their lives are political . This case did not come near those areas," said Judge Amy Berman Jackson . "I cannot do it and I will not do it ." She rejected Jackson Jr .ś defense that his bipolar disorder played a role, saying his string of accomplishments—" propped up by a political family dynasty"—"points to only one conclusion, and that is that you knew better ." She also rejected . . .

Chapter 6

Conclusions

6.1 Research Summary

This thesis studies two primary fields in NLP. Among them, text encoding (or text representation) is very general and is almost a necessary step for deep learning NLP. By contrast, text decoding can be regarded as one of the downstream tasks of the former, and its objective is to recover these encoded hidden states to readable text sequences. In general, our contributions can be divided into two parts, namely, to improve generalized text representation and sequence-to-sequence generation. In Chapter 3, we focus on promoting the distributional representations of documents with potential connections to others in the cluster. And in Chapters 4 and 5, we improve sequence-to-sequence generation, which is a widespread application composed of both encoding and decoding. We improve it from both the encoder (Chapter 4) and the decoding strategy (Chapter 5). Besides, these improvement methods are all designed based on the global perspective to explore higher-level knowledge to complete or guide the original operations of encoding and decoding.

The specific definitions of global information are different, summarized as follows:

- In Chapter 3, global information is the contextual relationships between news documents. This global context is often ignored because it is implicit and required to be constructed first. Instead, local contexts, such as inner words in the document, are explicit and exist naturally. Thus, they become the natural choices to extend the distributional assumption from the words to documents that documents with similar words inside should have similar representations. However, this assumption fails to capture the dependencies across documents, so it is incomplete for news embeddings.

In our proposed news network embedding model, the distributional assumptions are amended so that documents with similar inner components and potentially connected news events should have similar vectors. We define that two news events have latent relationships when they are close in position in the news network. In other words, news nodes with similar topological structures will have similar embeddings. Its first-order neighbors are equal to the local contexts, consisting of inner components. In contrast, its second-order neighbors refer to global contexts, i.e., the potentially linked news documents. Compared with traditional document embedding methods using local contexts only, our representations are enriched with additional knowledge, thus benefiting some downstream tasks where document dependencies work.

- In Chapter 4, the global information indicates the interaction relationships across documents or paragraphs, and the local information, namely the interaction relationships across words in the same document. We leverage the multi-head attention module to capture. When generating one word at inference time, the global cross attention model will decide which document the new word is from, and the local cross attention will choose the possible word from the document. The proposed model transforms the semantic information of multi-documents into two different levels of knowledge and learns them simultaneously in a parallel way. It enjoys the training efficiency and takes up lower memory because it uses the shared encoder and does not need to calculate the dependencies between words of different documents directly, which suffers from high complexity. By comparison, the traditional methods ignore the hierarchical information and concatenate all documents as a flat sequence. However, as the concatenated sequence becomes longer and longer, there is a burden on training speed and the difficulties of finding a reasonable model to capture such a long dependency. Overall, our text generator does not introduce additional knowledge and is still based on source documents completely. Our contribution is to divide the existing knowledge into two levels to learn (namely the global and local levels), thus capturing the cross-document relationships explicitly. We designed a new experiment to verify that the proposed hierarchical structure is easier to learn such relationships and thus explain why the generation quality is improved. Besides, the model uses the source content to predict how the reference attends to source documents and then uses the predicted attention distribution to guide the decoding. This is actually the embryonic form of the model proposed in Chapter 5 and inspires

our future work.

- In Chapter 5, the global information is defined as the global (optimal) attention distribution, i.e., how the optimal target sequence (namely the reference) attends to the source sentence. This study develops a calibrated beam-based algorithm with awareness of the global attention distribution for neural abstractive summarization, aiming to improve the local optimality problem of the original beam search in a rigorous way. Specifically, a novel global protocol is proposed based on the attention distribution to stipulate how a global optimal hypothesis should attend to the source. A global scoring mechanism is then developed to regulate beam search to generate summaries in a near-global optimal fashion. This novel design enjoys a distinctive property, i.e., the global attention distribution could be predicted before inference, enabling step-wise improvements on the beam search through the global scoring mechanism. Extensive experiments on nine datasets show that the global (attention)-aware inference significantly improves state-of-the-art summarization models even using empirical hyper-parameters. The algorithm is also proven robust as it remains to generate meaningful texts with corrupted attention distributions.

6.2 Future Research

My future work will focus on sequence-to-sequence generations, which is, as mentioned before, the most widely-known task that combines both encoding and decoding strategies. All our current work can be applied to this field. For instance, the proposed models introduced in Chapters 3 and 4 respectively indicate that we can use the graph networks or hierarchical structures to represent cross-document dependencies, thus benefiting the encoding of multi-document sources. On the other hand, our global-aware beam search can be generalized to almost all sequence-to-sequence models as long as they adopt the cross-attention mechanism. It should be mentioned that attention models have been the mainstream representation models in almost all fields, not only in NLP, but also in computer vision, speech, etc. That is to say, our improved decoding strategy can be applied to anything-to-text tasks instead of text-to-text only. Therefore, one of the future works is to adapt the global-aware beam search to improve other source-based text generations, where the source could be the image, video, data table, etc. The other future work is to explore more general source-based sequence generations. Not limited to text generations,

different generations like graph generations and image generations can also be conducted in a similar autoregressive way. Now I am working on the study of text-to-graph generations, which brings more challenges than text generations. Similar to text generations, graphs are also generated components by components conditional on source documents. However, the difference is that components in text generations are isomorphic and all refer to vocabulary words. Still, graphs are composed of heterogeneous components following different distributions, namely nodes and edges. Besides, there exist some difficulties in graph generations, like the diversity of topological orders, the sparsity of edge connections and the high complexity when generating edges autoregressively.

Bibliography

[1] R. Akita, A. Yoshihara, T. Matsubara, and K. Uehara. Deep learning for stock prediction using numerical and textual information. In 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS), pages 1–6, June 2016.

[2] Thomas Bachlechner, Bodhisattwa Prasad Majumder, Huanru Henry Mao, Garrison W. Cottrell, and Julian McAuley. Rezero is all you need: Fast convergence at large depth, 2020.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2016.

[4] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. Enriching Word Vectors with Subword Information. arXiv e-prints, July 2016.

[5] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, 5:135–146, 2017.

[6] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2, NIPS'13, pages 2787–2795, USA, 2013. Curran Associates Inc.

[7] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess,

Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[8] Deng Cai and Wai Lam. AMR parsing via graph-sequence iterative inference. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 1290–1301, Online, July 2020. Association for Computational Linguistics.

[9] Shuyang Cao and Lu Wang. Attention head masking for inference time content selection in abstractive summarization. ArXiv, 2104.02205, 2021.

[10] 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, SIGIR '98, pages 335–336, New York, NY, USA, 1998. ACM.

[11] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. Universal sentence encoder. CoRR, abs/1803.11175, 2018.

[12] Yen-Chun Chen and Mohit Bansal. Fast abstractive summarization with reinforce-selected sentence rewriting. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2018.

[13] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. URL https://openai.com/blog/sparse-transformers, 2019.

[14] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, pages 103–111, Doha, Qatar, October 2014. Association for Computational Linguistics.

[15] Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. A discourse-aware attention model for abstractive summarization of long documents. Proceedings of the 2018 Conference of the

North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), 2018.

[16] Eldan Cohen and Christopher Beck. Empirical analysis of beam search performance degradation in neural sequence models. In International Conference on Machine Learning, pages 1290–1299, 2019.

[17] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. CoRR, abs/1705.02364, 2017.

[18] Alexis Conneau, Germán Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. CoRR, abs/1805.01070, 2018.

[19] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 2019.

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[21] Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. Deep learning for event-driven stock prediction. In Proceedings of the 24th International Conference on Artificial Intelligence, IJCAI'15, pages 2327–2333. AAAI Press, 2015.

[22] Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. Knowledge-driven event embedding for stock prediction. In COLING, 2016.

[23] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. Unified language model pre-training for natural language understanding and generation. In Advances in Neural Information Processing Systems, pages 13063–13075, 2019.

[24] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In International Conference on Learning Representations, 2020.

[25] Günes Erkan and Dragomir R. Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. J. Artif. Int. Res., 22(1):457–479, December 2004.

[26] Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 1074–1084, Florence, Italy, July 2019. Association for Computational Linguistics.

[27] Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. Bottom-up abstractive summarization. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 4098–4109, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[28] Alex Graves. Sequence transduction with recurrent neural networks, 2012.

[29] Aditya Grover and Jure Leskovec. Node2vec: Scalable feature learning for networks. In Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, pages 855–864, New York, NY, USA, 2016. ACM.

[30] Max Grusky, Mor Naaman, and Yoav Artzi. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. 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 708–719, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[31] Songqiao Han, Xiaoling Hao, and Hailiang Huang. An event-extraction approach for business analysis from online chinese news. Electronic Commerce Research and Applications, 28, 02 2018.

[32] Zellig S Harris. Distributional structure. Word, 10(2-3):146–162, 1954.

[33] Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1, NIPS'15, page 1693–1701, Cambridge, MA, USA, 2015. MIT Press.

[34] Felix Hill, Kyunghyun Cho, and Anna Korhonen. Learning distributed representations of sentences from unlabelled data. CoRR, abs/1602.03483, 2016.

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

[36] Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration, 2019.

[37] Ziniu Hu, Weiqing Liu, Jiang Bian, Xuanzhe Liu, and Tie-Yan Liu. Listening to chaotic whispers: A deep learning framework for news-oriented stock trend prediction. In Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM '18, pages 261–269, New York, NY, USA, 2018. ACM.

[38] Xinyu Hua and Lu Wang. Sentence-level content planning and style specification for neural text generation. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), 2019.

[39] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. arXiv preprint arXiv:1508.01991, 2015.

[40] Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. Ctrl: A conditional transformer language model for controllable generation, 2019.

[41] Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. Text generation from knowledge graphs with graph transformers. 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), pages 2284–2293, 2019.

[42] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. Computer, 42(8):30–37, 2009.

[43] Anastassia Kornilova and Vladimir Eidelman. BillSum: A corpus for automatic summarization of US legislation. In Proceedings of the 2nd Workshop on New Frontiers in Summarization, pages 48–56, Hong Kong, China, November 2019. Association for Computational Linguistics.

[44] Mahnaz Koupaee and William Yang Wang. Wikihow: A large scale text summarization dataset. arXiv, 1810.09305, 2018.

[45] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In Eric P. Xing and Tony Jebara, editors, Proceedings of the 31st International Conference on Machine Learning, volume 32 of Proceedings of Machine Learning Research, pages 1188–1196, Bejing, China, 22–24 Jun 2014. PMLR.

[46] Logan Lebanoff, Kaiqiang Song, and Fei Liu. Adapting the neural encoder-decoder framework from single to multi-document summarization. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 4131–4141, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[47] Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. Advances in neural information processing systems, 27:2177–2185, 2014.

[48] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7871–7880, Online, July 2020. Association for Computational Linguistics.

[49] 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, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[50] Jiwei Li and Eduard Hovy. A model of coherence based on distributed sentence representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 2039–2048, Doha, Qatar, October 2014. Association for Computational Linguistics.

[51] Jiwei Li, Will Monroe, and Dan Jurafsky. Learning to decode for future success. CoRR, abs/1701.06549, 2017.

[52] Mingzhe Li, Xiuying Chen, Shen Gao, Zhangming Chan, Dongyan Zhao, and Rui Yan. Vmsmo: Learning to generate multimodal summary for video-based news articles. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 9360–9369, 2020.

[53] Q. Li, Y. Chen, J. Wang, Y. Chen, and H. Chen. Web media and stock markets : A survey and future directions from a big data perspective. IEEE Transactions on Knowledge and Data Engineering, 30(2):381–399, Feb 2018.

[54] Wei Li, Xinyan Xiao, Jiachen Liu, Hua Wu, Haifeng Wang, and Junping Du. Leveraging graph to improve abstractive multi-document summarization. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 6232–6243, Online, July 2020. Association for Computational Linguistics.

[55] Wei Li, Xinyan Xiao, Yajuan Lyu, and Yuanzhuo Wang. Improving neural abstractive document summarization with structural regularization. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 4078–4087, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[56] Yanyang Li, Tong Xiao, Yinqiao Li, Qiang Wang, Changming Xu, and Jingbo Zhu. A simple and effective approach to coverage-aware neural machine translation. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 292–297, 2018.

[57] Zeyu Liang, Junping Du, Yingxia Shao, and Houye Ji. Gated graph neural attention networks for abstractive summarization. Neurocomputing, 431:128–136, 2021.

[58] Weizhi Liao, Yaheng Ma, Yanchao Yin, Guanglei Ye, and Dongzhou Zuo. Improving abstractive summarization based on dynamic residual network with reinforce dependency. Neurocomputing, 448:228–237, 2021.

[59] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In Text Summarization Branches Out, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[60] Peter J. Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. Generating wikipedia by summarizing long sequences, 2018.

[61] Yang Liu and Mirella Lapata. Hierarchical transformers for multi-document summarization. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 2019.

[62] Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. Multilingual denoising pre-training for neural machine translation. arXiv preprint arXiv:2001.08210, 2020.

[63] Lajanugen Logeswaran and Honglak Lee. An efficient framework for learning sentence representations. CoRR, abs/1803.02893, 2018.

[64] Ronny Luss and Alexandre D'Aspremont. Predicting abnormal returns from news using text classification. Quantitative Finance, 15(6):999–1012, 2015.

[65] Congbo Ma, Wei Emma Zhang, Mingyu Guo, Hu Wang, and Quan Z. Sheng. Multi-document summarization via deep learning techniques: A survey, 2020.

[66] Hongyuan Mei, Mohit Bansal, and Matthew R. Walter. What to talk about and how? selective generation using lstms with coarse-to-fine alignment. Computer Science, 2015.

[67] Clara Meister, Ryan Cotterell, and Tim Vieira. If beam search is the answer, what was the question? In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 2173–2185, 2020.

[68] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781, 2013.

[69] Ani Nenkova and Kathleen McKeown. Automatic summarization. Now Publishers Inc, 2011.

[70] Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. Analyzing uncertainty in neural machine translation. In International Conference on Machine Learning, pages 3956–3965, 2018.

[71] Ramakanth Pasunuru and Mohit Bansal. Multi-reward reinforced summarization with saliency and entailment. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), 2018.

[72] 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, Doha, Qatar, October 2014. Association for Computational Linguistics.

[73] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, pages 701–710, New York, NY, USA, 2014. ACM.

[74] Matthew E. Peters, Mark Neumann, Mohit Iyyer, 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, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[75] Tribikram Pradhan, Chaitanya Bhatia, Prashant Kumar, and Sukomal Pal. A deep neural architecture based meta-review generation and final decision prediction of a scholarly article. Neurocomputing, 428:218–238, 2021.

[76] Ratish Puduppully, Li Dong, and Mirella Lapata. Data-to-text generation with content selection and planning. In Proceedings of the AAAI conference on artificial intelligence, pages 6908–6915, 2019.

[77] Alec Radford and Karthik Narasimhan. Improving language understanding by generative pre-training. 2018.

[78] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. OpenAI Blog, 1(8), 2019.

[79] 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. Journal of Machine Learning Research, 21(140):1–67, 2020.

[80] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks, 2016.

[81] David E. Rumelhart and James L. McClelland. Learning Internal Representations by Error Propagation, pages 318–362. 1987.

[82] 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, pages 379–389, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[83] Rob Schumaker and Hsiu-chin Chen. Textual analysis of stock market prediction using breaking financial news: The azfin text system. ACM Trans. Inf. Syst., 27, 02 2009.

[84] M. Schuster and K.K. Paliwal. Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing, 45(11):2673–2681, 1997.

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

[86] 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 (Volume 1: Long Papers), pages 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics.

[87] Vinay Setty and Katja Hose. Event2vec: Neural embeddings for news events. In The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR '18, pages 1013–1016, New York, NY, USA, 2018. ACM.

[88] Nathan Srebro and Tommi Jaakkola. Weighted low-rank approximations. In Proceedings of the Twentieth International Conference on International Conference on Machine Learning, ICML'03, page 720–727. AAAI Press, 2003.

[89] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2, NIPS'14, page 3104–3112, Cambridge, MA, USA, 2014. MIT Press.

[90] Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. Abstractive document summarization with a graph-based attentional neural model. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1171–1181, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[91] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[92] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In International Conference on Learning Representations, 2018.

[93] Ashwin K Vijayakumar, Michael Cogswell, Ramprasaath R Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. Diverse beam search for improved description of complex scenes. In Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[94] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. J. Mach. Learn. Res., 11:3371–3408, December 2010.

[95] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. 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.

[96] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2015.

[97] Michael Völske, Martin Potthast, Shahbaz Syed, and Benno Stein. TL;DR: Mining Reddit to learn automatic summarization. In Proceedings of the Workshop on New Frontiers in Summarization, pages 59–63, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[98] Baohua Wang, Hejiao Huang, and Xiaolong Wang. A novel text mining approach to financial time series forecasting. Neurocomput., 83:136–145, April 2012.

[99] Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. Neural text generation with unlikelihood training, 2019.

[100] Sam Wiseman and Alexander M. Rush. Sequence-to-sequence learning as beam-search optimization. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 1296–1306, Austin, Texas, November 2016. Association for Computational Linguistics.

[101] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 38–45, Online, October 2020. Association for Computational Linguistics.

[102] Y. Wu, M. Schuster, Z. Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, M. Krikun, Yuan Cao, Q. Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, M. Johnson, X. Liu, Lukasz Kaiser, S. Gouws, Y. Kato, Taku Kudo, H. Kazawa, K. Stevens, G. Kurian, Nishant Patil, W. Wang, C. Young, J. Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, G. Corrado, Macduff Hughes, and J. Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. ArXiv, abs/1609.08144, 2016.

[103] B. Wuthrich, V. Cho, S. Leung, D. Permunetilleke, K. Sankaran, and J. Zhang. Daily stock market forecast from textual web data. In SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No.98CH36218), volume 3, pages 2720–2725 vol.3, Oct 1998.

[104] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture, 2020.

[105] Yilin Yang, Liang Huang, and Mingbo Ma. Breaking the beam search curse: A study of (re-)scoring methods and stopping criteria for neural machine translation. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 3054–3059, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[106] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1480–1489, San Diego, California, June 2016. Association for Computational Linguistics.

[107] Kaichun Yao, Libo Zhang, Tiejian Luo, and Yanjun Wu. Deep reinforcement learning for extractive document summarization. Neurocomputing, 284:52–62, 2018.

[108] Yongjian You, Weijia Jia, Tianyi Liu, and Wenmian Yang. Improving abstractive document summarization with salient information modeling. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 2132–2141, 2019.

[109] Jingqing Zhang, Y. Zhao, Mohammad Saleh, and Peter J. Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In ICML, 2020.

[110] Wen Zhang, Yang Feng, Fandong Meng, Di You, and Qun Liu. Bridging the gap between training and inference for neural machine translation. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 4334–4343, Florence, Italy, July 2019. Association for Computational Linguistics.

[111] Xingxing Zhang, Furu Wei, and Ming Zhou. HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 5059–5069, Florence, Italy, July 2019. Association for Computational Linguistics.

[112] J. Zhou, T. Naseem, R. F. Astudillo, and R. Florian. Amr parsing with action-pointer transformer. NAACL, 2021.

[113] Juntang Zhuang, Tommy Tang, Yifan Ding, Sekhar Tatikonda, Nicha Dvornek, Xenophon Papademetris, and James Duncan. Adabelief optimizer: Adapting stepsizes by the belief in observed gradients. Conference on Neural Information Processing Systems, 2020.