applied sciences

MDPI

Article

Incorporating Multi-Hypotheses as Soft-Templates in Neural Headline Generation

Yana A [1], Zhenghao Liu [2,*], Suyalatu Dong [1] and Fanyu Bu [1]

[1] Department of Computer Information Management, Inner Mongolia University of Finance and Economics, Hohhot 010070, China; ayn@imufe.edu.cn (Y.A.); srguleng_1983@163.com (S.D.); bufanyu@imufe.edu.cn (F.B.)
[2] Department of Computer Science and Technology, Northeastern University, Shenyang 110167, China
* Correspondence: liuzhenghao@mail.neu.edu.cn

Abstract: Neural models are widely applied to headline generation. Template-based methods are a promising direction to overcome the shortcomings of the neural headline generation (NHG) model in generating duplicate or extra words. Previous works often retrieve relevant headlines from the training data and adopt them as the soft template to guide the NHG model. However, these works had two drawbacks: reliance on additional retrieval tools, and uncertainty regarding semantic consistency between the retrieved headline and the source article. The NHG model uncertainty can be utilized to generate hypotheses. The hypotheses generated based on a well-trained NHG model not only contain salient information but also exhibit diversity, making them suitable as soft templates. In this study, we use a basic NHG model to generate multiple diverse hypotheses as candidate templates. Then, we propose a novel **M**ultiple-**H**ypotheses-based NHG (MH-NHG) model. Experiments on English headline generation tasks demonstrate that it outperforms several baseline systems and achieves a comparable performance with the state-of-the-art system. This indicates that MH-NHG can generate more accurate headlines guided by multiple hypotheses.

Keywords: soft template; neural headline generation; multiple hypotheses

Citation: A, Y.; Liu, Z.; Dong, S.; Bu, F. Incorporating Multi-Hypotheses as Soft-Templates in Neural Headline Generation. *Appl. Sci.* **2023**, *13*, 8478. https://doi.org/10.3390/app13148478

Academic Editor: Panagiotis G. Asteris

Received: 19 June 2023
Revised: 13 July 2023
Accepted: 20 July 2023
Published: 22 July 2023

1. Introduction

The past several years have witnessed the rapid development of end-to-end neural headline generation (NHG) [1–6]. Through a vast neural network, the end-to-end NHG system maps between the input article and the headline and generates the corresponding headline for a document word by word without additional linguistic knowledge and more manual annotation. Recently, the emergence of large-scale pre-trained language models has further improved the performance of NHG models [7–12].

Nevertheless, the input information of most previous models is simply the source article. Due to the complexity and verbosity of natural language text, and the limited size of training data, these models tend to deteriorate as they generate more words, often producing irrelevant and repeated words [3,13]. This deterioration occurs because the models struggle to distinguish important information from noise. The accumulation of word generation further exacerbates this issue. In the preliminary phases of summarization, the template-based approach [14] demonstrates considerable potential. In this approach, researchers create a heuristic template framework by manually formulating rules and filling in the required information, to generate a headline based on these templates. This approach offers the advantage of producing concise and coherent summaries without the need for training data. However, the manual creation of all templates is impractical due to the substantial domain knowledge and labor-intensive nature of the task. Cao et al. [15] and Wang et al. [4] extended this idea in the deep learning context. They proposed to make better use of the available training data and retrieve existing headlines as soft templates to guide the summarization process.

Although previous template-based methods avoid the problem of manually designing the headline templates, they had two drawbacks. First, their retrieval modules still need additional retrieval tools and carefully designed retrieving rules. Second, the retrieved headline is uncertain to be semantically consistent with the source article. In Figure 1, we provide a news fragment with a reference headline and different templates. The concise, human-written template to conclude the given article is designated as the "Hard Template". The existing headline retrieved from the training data is denoted as the"Retrieved Headline" (This example is cited from Cao et al. [15]). Despite being written by humans and being grammatically correct, the ground truth "Retrieved Headline" is semantically inconsistent with the original article.

Article:	European stock markets advanced strongly Thursday on bargain-hunting and gains by wall street and Japanese shares a head of an expected hike in us interest rates , dealers said .
Reference:	European stocks bounce back UNK UNK with closing levels
Hard Template:	[REGION] stocks [bounce back/fall] with closing levels
Retrieved Headline:	European shares sharply **lower** on us interest rate **fears**
Hypothesis1:	European shares **higher ahead of us rate decision**
Hypothesis2:	European stocks advance on bargain-hunting **wall street gains**
Hypothesis3:	European stocks advance on bargain-hunting
Hypothesis4:	European stocks rise ahead of us **rate hike**

Figure 1. A news article paired with reference headline and different templates. We use Bold font to indicate the difference between templates. The red texts denote the semantically different words in the retrieved headline.

To address these issues, we propose to utilize the multiple hypotheses generated based on a well-trained NHG model as soft templates to improve headline generation performance. The presence of uncertainties in the multiple hypotheses stems from both the confidence of the model and the potential ambiguity arising from linguistic variations [16], offering an opportunity to enhance machine translation performance [17,18]. Model uncertainty typically gives rise to the K-best hypotheses obtained through a beam search [19]. In Figure 1, we also provide hypotheses generated based on a well-trained NHG model which are indicated as Hypothesis 1~4. The hypotheses generated using various sampling methods are expected to have differing headline words while striving to maintain consistent semantics.

To this end, we introduce a new model called the **M**ultiple-**H**ypotheses-based NHG (MH-NHG) model. Our approach leverages multiple hypotheses generated by a well-trained basic NHG model, which serve as soft templates to enhance the performance of the NHG model. The MH-NHG model is a hierarchical multi-stage architecture consisting of four layers to effectively integrate the multiple hypotheses. The initial layer consists of a contextual embedding layer, where each hypothesis forms a node with the original article and is then transformed into word embeddings using a pre-trained language model. The second layer is a inter-node interactive attention layer. Nodes constructed with different templates contain unique information, and the inter-node interactive attention layer aids in capturing the crucial information more effectively. The third layer is a intra-node interactive attention layer. Within each node, there exists a relationship between document words and template words. The intra-node interactive attention layer aims to capture this relationship. The fourth layer is a node selection attention layer, which aims to control the proportion of different nodes in the final fine-grained node representation.

We evaluate our model on a large-scale English headline generation dataset extracted from Gigaword [20]. The experimental results demonstrate that our model consistently outperforms the plain baseline system in each setting, verifying the significance and the robustness of the proposed model. The novelty of this study lies in the fact that our method can integrate a hypotheses-based soft-template in headline generation while keeping a simple overall framework. The main contributions of this study are as follows:

1. We utilize multi-hypotheses as soft templates to incorporate the diverse information contained in the approximate decoding candidates and assist in the target headline generation.
2. Different from previous template-based studies that require template retrieval and heuristic rules to draw templates from the training set, our approach only needs the NHG model itself. Hence, it can easily be applied to arbitrary NHG models trained for arbitrary language pairs.
3. We evaluate our model on the English headline generation task. We further conduct a pre-trained language model-based evaluation in addition to the traditional automatic evaluation metric, Recall-oriented Understudy for Gisting Evaluation (ROUGE) [21], to ensure more reliable evaluation results. The experimental results demonstrate that our method is not only effective, but also more interpretive.

2. Method

As shown in Figure 2, our headline generation system consists of two modules, i.e., a basic NHG model and the proposed MH-NHG model. Given the input article **x**, the basic NHG model generates multiple hypotheses employing various sampling methods. Subsequently, the MH-NHG model integrates both the original article and the multiple hypotheses to generate a headline of improved quality.

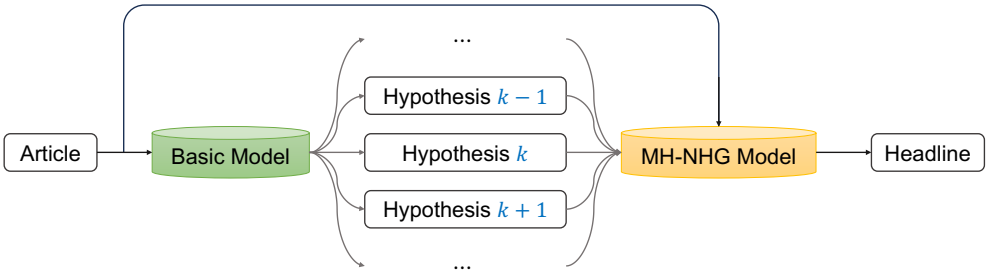

Figure 2. Flow chat of the proposed method.

2.1. Model Architecture

We propose herein MH-NHG, a hierarchical multi-stage NHG architecture that includes four layers to better integrate the multiple hypotheses. Figure 3 depicts the entire framework of the proposed model.

1. The **contextual embedding layer** maps each node into hidden representations with a pre-trained language model.
2. The **inter-node interactive attention layer** fully connects all nodes and attentively reads tokens in the nodes to gather supporting information to build fine-grained node representations.
3. The **intra-node interactive attention layer** makes the connection between the article words and the template words within each node to build more confirmative word representations.
4. The **node selection layer** assigns selection attention scores to the inter-node interactive node representations and calculates the final fine-grained node representations.
5. The **decoder layer** employs a Transformer decoder to decode the provided representations and output a predicted headline.

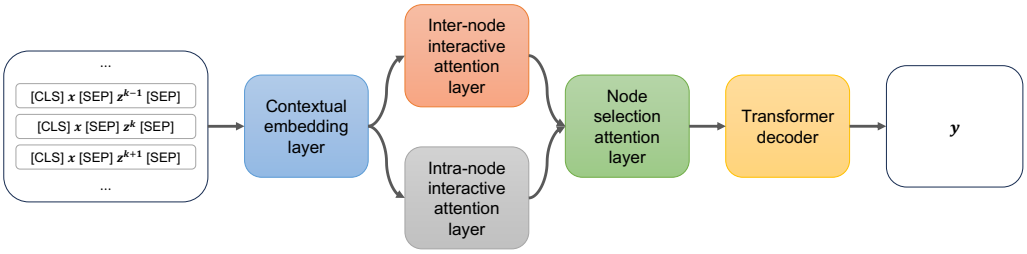

Figure 3. Overview of the proposed MH-NHG model.

2.1.1. Contextual Embedding Layer

Given a source news article $\mathbf{x} = (x_1, \cdots, x_i, \cdots, x_M)$ with M words, a target headline $\mathbf{y} = (y_1, \cdots, y_j, \cdots, y_N)$ with N words ($N < M$), and K corresponding soft templates $\mathbf{z} = [z^1, \cdots, z^k, \cdots, z^K]$, each with $\hat{N}^k$ words generated by an NHG model, we first regard each article-template pair $\langle \mathbf{x}, z^k \rangle$ as a node. Pre-trained language models (e.g., BERT [22]) demonstrate their potentials of producing contextual token representations for various NLP tasks. We adopt the pre-trained language model, such as BERT, to obtain the representations of each token in all nodes. For node $\langle \mathbf{x}, z^k \rangle$ with M source article tokens and $\hat{N}^k$ template tokens, the node representation $\boldsymbol{H}^k$ encoded by the pre-trained language model is presented in Equation (1):

$$\begin{aligned} \boldsymbol{H}^k &= \text{BERT}(\text{[CLS]}\ \mathbf{x}\ \text{[SEP]}\ \mathbf{z}^k\ \text{[SEP]}) \\ &= \{H_0^k, \cdots, H_{M+\hat{N}^k+2}^k\} \end{aligned} \tag{1}$$

where H_0^k and $H_{M+\hat{N}^k+2}^k$ denote the "[CLS]" and "[SEP]" representations, respectively.

2.1.2. Inter-Node Interactive Attention Layer

We believe that the different nodes constructed with different soft-templates contain unique information. Moreover, the semantic interaction between the different nodes will help our model to better capture salient information. The fully connected inter-node interactive attention layer intends to implement the thought.

For H_i^k, which is the initial representation of the i-th token in the k-th node, we first calculate the node interaction attention weight $\alpha_j^{p \to k}$ according to Equation (2) based on the initial representation of the j-th token in the p-th node:

$$\alpha_j^{p \to k} = \text{softmax}_j((H_i^k)^T) \cdot W_{Inter} \cdot H_j^p) \tag{2}$$

where W_{Inter} denotes a matrix parameter.

The representations for the i-th token in the k-th node that aggregated the information from the p-th node is calculated based on Equation (3):

$$G_i^{p \to k} = \sum_{j=0}^{M+\hat{N}^p+2} (\alpha_j^{p \to k} \cdot H_j^p). \tag{3}$$

According to Equation (3), we can further build the aggregated representation for the k-th node with regard to the p-th node based on Equation (4):

$$\boldsymbol{G}^{p \to k} = \{G_0^{p \to k}, \cdots, G_i^{p \to k}, \cdots, G_{M+\hat{N}^k+2}^{p \to k}\} \tag{4}$$

2.1.3. Intra-Node Interactive Attention Layer

The k-th node is constructed with the source article and the k-th soft template. The article and template words within each node also have relationships. The intra-node interactive attention layer intends to capture these relationships. Inspired by the attention-over-attention mechanism [23], we first calculate the pair-wise matching matrix M^k to

indicate the pair-wise matching degree of the source article and the soft template in the k-th node. For each element M^k_{ij} of M^k is calculated with the contextual representation of the i-th source article token and the j-th soft template token based on Equation (5):

$$M^k_{ij} = (H^k_i)^T \cdot W_{Intra} \cdot H^k_{M+1+j}, \tag{5}$$

where W_{Intra} denotes the weight matrix.

We then obtain the attention score along the source dimension β^{ks}_i and the attention score along the soft template dimension β^{kt}_j according to Equation (6):

$$\begin{aligned} \beta^{ks}_i &= \frac{1}{\hat{N}^k + 1} \sum_{j=0}^{\hat{N}^k+1} \text{softmax}_i(M^k_{ij}) \\ \beta^{kt}_j &= \frac{1}{M+1} \sum_{i=0}^{M+1} \text{softmax}_j(M^k_{ij}) \end{aligned} \tag{6}$$

Thereafter, the representations of the source article and the soft template are calculated based on Equation (7):

$$\begin{aligned} h^{ks} &= \sum_{i=0}^{M+1} \beta^{ks}_i \cdot H^k_i \\ h^{kt} &= \sum_{j=0}^{\hat{N}^k+1} \beta^{kt}_j \cdot H^k_{M+1+j} \end{aligned} \tag{7}$$

2.1.4. Node Selection Attention Layer

The node selection layer intends to measure the proportion of $\boldsymbol{G}^{p\to k}$ in the final fine-grained node representation $\boldsymbol{G}^k$ of the k-th node based on Equation (8):

$$\gamma^k = \text{softmax}_k(\text{Linear}((h^{ks} \circ h^{kt}); h^{ks}; h^{kt})), \tag{8}$$

where $\circ$ represents element-wise multiplication, and ; indicates concatenation. Given the node selection attention score γ^k, the final representation for the i-th token of the k-th node is calculated based on Equation (9):

$$G^k_i = \sum_{p=1}^{K} \gamma^k \cdot G^{p\to k}_i \tag{9}$$

According to Equation (9), the final representation of the k-th node is calculated based on Equation (10):

$$\boldsymbol{G}^k = \left\{ G^k_0, \cdots, G^k_i, \cdots, G^k_{M+\hat{N}^k+2} \right\} \tag{10}$$

2.1.5. Decoder Layer

We simply adopt the traditional Transformer decoder to generate the headline word by word. The conditional probability of generating the j-th target word is calculated based on Equation (11):

$$\Pr(y^k_j | \mathbf{y}^k_{<j}, \boldsymbol{G}^k) = \text{softmax}(\text{FFN}(r^k_{L,j})), \tag{11}$$

where $r^k_{L,j}$ is a vector from the target representation matrix $\boldsymbol{R}^k_L$; L is the decoder depth; and $\text{FFN}(\cdot)$ represents the feed forward network. The $\boldsymbol{R}^k_L$ is defined according to Equation (12):

$$\boldsymbol{R}^k_L = \text{LN}(\text{FFN}(\boldsymbol{S}^k_L + \boldsymbol{C}^k_L)), \tag{12}$$

where LN$(\cdot)$ indicates the layer normalization [24], and $\boldsymbol{S}_L^k$ is computed according to Equation (13):

$$\boldsymbol{S}_L^k = \mathrm{LN}(\boldsymbol{T}_L^k + \mathrm{FFN}(\boldsymbol{S}_{L-1}^k)), \tag{13}$$

where $\boldsymbol{S}_{L-1}^k$ is from the $(L-1)$-th layer. $\boldsymbol{T}_L^k$ is the result of the self-attention layer which is computed according to Equation (14). This stage is intended to capture the relationships between the target words, and the query, key, and value matrix are all set as $\boldsymbol{S}_{L-1}^k$:

$$\begin{aligned} \boldsymbol{T}_L^k &= \mathrm{Att}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) \\ &= \mathrm{Att}(\boldsymbol{S}_{L-1}^k, \boldsymbol{S}_{L-1}^k, \boldsymbol{S}_{L-1}^k), \end{aligned} \tag{14}$$

where Att$(\cdot)$ is the self-attention network and $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$ are the query, key, and value matrix, respectively.

$\boldsymbol{C}_L$ in Equation (12) is used to capture the source-target relationships by using the self-attention mechanism, and is calculated according to Equation (15). The query matrix is equal to the target side representation $\boldsymbol{S}_L^k$, while the key and the value matrix is set as the source side representation matrix $\boldsymbol{G}^k$ with the corresponding definition:

$$\begin{aligned} \boldsymbol{C}_L^k &= \mathrm{Att}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) \\ &= \mathrm{Att}(\boldsymbol{S}_L^k, \boldsymbol{G}^k, \boldsymbol{G}^k). \end{aligned} \tag{15}$$

2.2. End-to-End Training

We train our model on a constructed corpus using the maximum likelihood estimation as shown in Equation (16):

$$\hat{\boldsymbol{\theta}} = \arg\max_{\theta} \{\mathcal{L}(D, \boldsymbol{\theta})\}, \tag{16}$$

where the log-likelihood is defined ~~as~~ in Equation (17)

$$\begin{aligned} \mathcal{L}(D, \boldsymbol{\theta}) &= \sum \log \mathrm{P}(\mathbf{y}|\mathbf{x}, \mathbf{z}, \boldsymbol{\theta}) \\ &= \sum \sum_{k=1}^{K} \log \mathrm{P}(\mathbf{y}|\mathbf{x}, \boldsymbol{z}^k, \boldsymbol{\theta}) \end{aligned} \tag{17}$$

where $\boldsymbol{\theta}$ contains all the trainable model parameters, $\hat{\boldsymbol{\theta}}$ indicates a set of optimized parameters; $\mathbf{x}$, $\mathbf{y}$, and $\boldsymbol{z}^k$ denote the source article, target headline and the k-th template, respectively; D is the training set size; and K is the number of constructed templates for each article.

2.3. Generating Multi-Hypotheses

As shown in Figure 3, the other module in our headline generation system is a basic NHG model which is utilized to generate multiple hypotheses.We assume that the utilization of multiple NHG hypotheses can contribute to the model performance enhancement in two ways. Firstly, the variations observed among these hypotheses can serve as indicators of model uncertainty and shed light on the potential ambiguity or intricacy of the source content. Secondly, they present an additional source of information, whereby, in cases where the initial NHG output diverges from the target headline due to acceptable linguistic variations, the supplemental evidence contained within the multiple NHG hypotheses can compensate for the missing evidence. Obtaining appropriate hypotheses has a large impact on the performance of the proposed model. In the context of sequence models, there are several sampling methods available to draw exact hypotheses from a model. In this study, we use four methods to generate hypotheses and investigate their impact on model performance.

2.3.1. Greedy Search

At each time step of decoding, this method always chooses the word with the highest generation probability to be the input of the next time step. This method is a greedy decoding process; thus, we denote it as **greedy search**.

2.3.2. Stochastic Sampling

Stochastic sampling is a standard solution for approximating the full search space [25]. At each time step of decoding, an output word is selected based on a multinomial distribution over the entire vocabulary given by the model. We denote this as **sampling**. This method could not only introduce more diverse data, but is also less time-consuming.

2.3.3. Beam Search

The hypothesis spaces in the neural sequence model are huge, and exhaustively exploring them to obtain an optimal solution is not feasible. Beam search is traditionally used for decoding in NHG by exploring the search space in a greedy left-to-right manner, retaining the top-B candidates with the highest probability. B denotes the beam width. The top-B extensions with the highest scores are selected at each time step of the beam search decoding. We denote this method as the **beam search**.

2.3.4. Stochastic Beam Search

While influential in selecting a likely translation, the beam search tends to result in a list of B-best translations that lack linguistic diversity [26,27]. Sampling can make the generated text more interesting by adding lower-probability words. The beam search makes the generated text more consistent by maximizing the total sequence probability. The stochastic beam search [27] explicitly applies the Gumbel-Top-k trick to sample the k sequences without replacement from a sequence model. We denote this method as the **stochastic beam search**.

3. Experiments

This section describes the experimental datasets, baseline systems, various implementation details, and evaluation methods used in this work.

3.1. Experimental Data

The experimental data in our work comes from English Gigaword [20], which is one of the largest static news corpus to date. It contains nearly 10 million news articles from seven major news sources with more than 4 billion words. It is also annotated with the following different annotation layers: sentence segmentation and tokenization tags, Treebank-style component parse tree tags, syntax dependency tree tags, and named entity tags. We performed a series of preprocessing to apply this data to the headline generation task, following [1]. Table 1 presents the statistical information.

Table 1. Statistics of English Gigaword. "Train", "Valid", and "Test" refers to the training, validation, and test set, respectively. art.avg.tok and head.avg.tok indicate the average token number of news articles and headlines in the training dataset, respectively.

Dataset	Statistics				
	Train	Valid	Test	art.avg.tok	head.avg.tok
English Gigaword	3,799,588	394,622	381,197	31.35	8.23

3.2. Baseline Systems

We compare our proposed model with the following systems:

- **Re^{3}Sum:** [15] proposed the retrieval of the input analogies in the corpus and picked their summaries as the candidate templates.

- **BiSET**: [4] also constructed a template-based model as **Re^{3}Sum**. The difference here was that they utilized an ingenious bidirectional selective encoding layer.
- **MASS:** [28] utilized an encoder–decoder framework to reconstruct a sentence fragment.
- **UniLM:** [8] was pre-trained using three types of language modeling tasks: unidirectional, bidirectional, and sequence-to-sequence prediction.
- **PEGASUS:** [7] proposed a new self-supervised training objective for abstractive text summarization.
- **PEGASUS + DotProd:** [6] enhanced the PEGASUS using a meta-learning algorithm.
- **ProphetNet:** [9] introduced a future N-Gram prediction and an n-stream self-attention mechanism to simultaneously predict the next n tokens.
- **OFA:** [12] is a unified paradigm for the multimodal pre-training framework.

3.3. Evaluation Method

3.3.1. ROUGE

The ROUGE [21] is the most widely used evaluation method for the document summarization task. It is officially adopted by DUC to evaluate the performance of a headline generation system. Inspired by the automatic evaluation criteria of machine translation, the basic idea behind ROUGE is to count the number of repeating items between the system-generated and reference summaries. ROUGE-N is the N-Gram recall value between the system-generated headline $\mathbf{y}'$ and reference headline $\mathbf{y}$. In our experiments, we adopt two kinds of N-Gram, namely, uni-Gram and bi-Gram, which correspond to ROUGE-1 and ROUGE-2, respectively. ROUGE-L measures the similarity of two sequences by counting the longest common subsequence.

3.3.2. BertScore

BERTScore [29] leverages the pre-trained contextual embeddings from BERT and matches words in the candidate and reference sentences by using the cosine similarity. For the same word in different sentences depending on the surrounding words, BERTScore first generates different vector representations with regard to contextual embeddings (e.g., BERT [22]). These vector representations allow BERTScore for the soft measure of similarity instead of exact matching. Cosine similarity is adopted for the calculation. According to detailed experiments [29], BERTScore has been demonstrated to correlate with human judgment on sentence and system-level evaluations and F1 performs reliably well across all different settings. Hence, we adopt F1 as the main metric.

4. Results

This section presents the experimental results, including the effects of different basic model setups and sampling methods, the main results, and the case study.

4.1. Effect of Basic Model Architecture

The selection of the basic model architecture plays a significant role in determining the system performance. Within this subsection, we explore diverse architectures in order to ascertain the most optimal configuration.

The Transformer [30] architecture is employed due to its notable proficiency in neural encoder–decoder functionality. To leverage the success of pre-trained models in natural language processing tasks, we utilize them as encoders and connect the output of the last layer to the Transformer decoder. Specifically, we employ the base versions of BERT [22] and ELECTRA [31] as our pre-trained language models.

For the Transformer, we leverage the base version consisting of a six-layer encoder and a six-layer decoder with a 512 embedding/hidden size and a 2048 feed-forward filter size. We set the max sentence lengths to 60 and 20 for the source and target sentences, respectively. Adam [32] with $\beta 1 = 0.9$, $\beta 2 = 0.999$ is used for the optimization. The learning rate is 1×10^{-8}. The dropout rate is 0.1. The weight decay is 0.01. The batch size is 64. We train

our models on four GeForce RTX 2080 GPU cards. The training takes approximately 4 h for 10,000 steps. We inherit the huggingface's PyTorch implementation for all experiments.

Table 2 shows the experimental results of each model structure. Aside from the ROUGE metrics, we also utilize BERTScore to perform the evaluation. BERTScore mainly focuses on solving the ROUGE problem, an evaluation metric based on the exact N-Gram matching, where semantically similar words would be considered incorrect. BERT + Transformer achieved higher ROUGE scores than the basic Transformer. The Electra + Transformer decoder outperformed both the basic Transformer and BERT + Transformer decoder. The BERTScore evaluation also presented the same results. BERT is a widely used bidirectional (or non-directional) pre-trained language model that reuses the encoder block from the Transformer and adopts self-supervised learning to learn the deep meaning of words and contexts. BERT is effective when fine-tuning on downstream tasks. In our experiments, incorporating BERT as the encoder in the NHG model could also bring improvements. ELECTRA comprises a generator and a discriminator, essentially two BERTs, with the generator focusing on masked language modeling and the discriminator specializing in token replacement detection. The training objective of the discriminator aligns more closely with headline generation, involving the replacement of words in a source document to produce a succinct headline. As a result, the Electra + Transformer decoder architecture outperformed the other two systems, leading us to adopt it as the foundational model structure for subsequent experiments.

Table 2. Experimental results on Gigaword test dataset. $RG-1$, $RG-2$ and $RG-L$ stand for F-measure scores of ROUGE-1, ROUGE-2 and ROUGE-L, respectively. F_{BERT} indicates the F-measure score of BERTScore. The highest scores are displayed in bold.

Model Architecture	$RG-1$	$RG-2$	$RG-L$	F_{BERT}
Transformer	37.56	18.79	34.94	59.56
Bert + Transformer decoder	38.31	19.33	35.57	60.04
Electra + Transformer decoder	**38.92**	**20.01**	**36.07**	**60.33**

4.2. Effect of Different Templates

We employ four methods for constructing soft templates and conduct a fair comparison by generating one template per article for each method (i.e., the template number K was set to 1). Figure 4 shows the experimental results on the Gigaword test dataset.

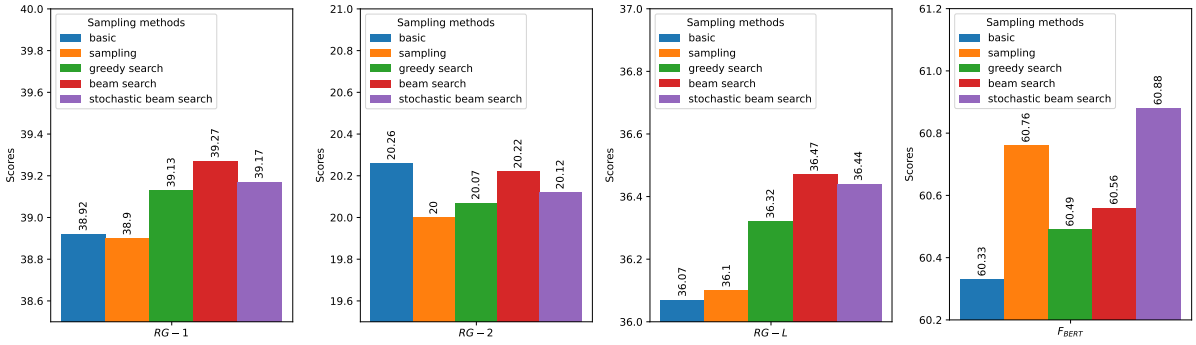

Figure 4. Experimental results of different templates. $RG-1$, $RG-2$ and $RG-L$ stand for F-measure scores of ROUGE-1, ROUGE-2 and ROUGE-L, respectively. F_{BERT} indicates the F-measure score of BERTScore.

The beam search consistently achieves the highest scores across all ROUGE metrics compared to the basic model and the other three sampling methods. This is attributed to the utilization of the beam search algorithm, which generates a high-quality soft template that has been proven effective in the sequence generation. The stochastic beam search method attains the second-best performance in terms of ROUGE scores. This approach introduces

Gumbel noise during the template generation process, enhancing diversity. On the other hand, both the sampling and greedy search methods exhibit inferior performances. Notably, a comparison between these two methods reveals that templates with greater diversity yield lower ROUGE scores.

Based on BERTScores, the stochastic beam search achieves a better score than the beam search. The sampling method outperforms the greedy search method. One possible explanation for this result is that stochastic beam search and sampling are the diverse versions of the beam and greedy search, respectively. Models trained with stochastic beam search and sampling generate more diverse words expressing the same meaning, which were captured by BERTScore rather than ROUGE metrics. Considering the aforementioned observations, we combine the soft templates generated by the four methods to conduct the follow-up experiments.

4.3. Main Results

Table 3 provides the experimental results on the Gigaword test dataset. We compare our model with several models introduced in Section 3.2. The first row presents the evaluation results of the models trained only on the paired data without any pre-training information; the second row presents the experimental results of the models trained in the paired data and the template information retrieved from the training data; and the final row corresponds to the model performance when pre-training methods were adopted. *R*1, *R*2 and *RL* stand for the F1 scores of ROUGE-1, ROUGE-2 and ROUGE-L, respectively. We illustrate the model architecture of each system in Table 4 to better analyze the possible impact factors on the model performance.

Table 3. Experimental results on Gigaword test dataset. The highest scores are displayed in bold.

Models	**GigaWord**		
	RG-1	**RG-2**	**RG-L**
Re^{3}Sum	37.04	19.03	34.46
BiSET	39.11	19.78	36.87
MASS	38.73	19.71	35.96
UniLM	38.90	20.05	36.00
PEGASUS	39.12	19.86	36.24
PEGASUS + DotProd	**40.60**	**21.00**	**37.00**
ProphetNet	39.51	20.42	36.69
OFA	39.81	20.66	37.11
Basic model	38.92	20.01	36.07
MH-NHG	39.44	20.02	36.69

Comparison with models without pre-training: As expected, our model, the baseline, and the proposed MH-NHG model exhibit a superior performance compared to the models without pre-training. This superiority can be attributed not only to the absence of a pre-training information module in these models but also to the fundamental architecture of the baseline model. Our model incorporates the Transformer [30], a proven architecture known for its successful application in sequence learning tasks, featuring multi-head attention and feed-forward layers that contribute significantly and effectively to the learning process.

Comparison with template-based models: Re^{3}Sum and BiSET are two template-based models that brought significant improvements in NHG. Our model outperforms Re^{3}Sum on all three ROUGE evaluation metrics and BiSET on the ROUGE-1 and -2 scores. However, our model exhibited an inferior performance to BiSET on the ROUGE-L score. One possible explanation for this is that the soft template in our model is made of system-generated hypotheses, while theirs is an actual headline from the training dataset. Re^{3}Sum and BiSET demonstrate the exact template retrieval mechanism. Given a source article, they first utilize the information retrieval library, Apache Lucene (https://lucene.apache.org (accessed on 1 January 2022)), to retrieve related source articles from the training dataset and

take the corresponding headlines as soft templates. While the retrieval process mitigates the need for manual rule creation in traditional template-based methods, it necessitates meticulous design to ensure its effectiveness. Enhancing the precision and efficacy of the retrieved information involves eliminating all non-alphabetic characters, thereby minimizing their impact on article matching. The retrieval system identifies a limited set of candidate articles based on the processed text. In contrast, our method solely relies on the model itself, simplifying its applicability to various NHG models.

Table 4. Model Architectures Corresponding to Table 3. In the "extra input" column, "-" means a model only takes word embeddings as the encoder input. In the "extra tool" column, "-" means a model does not utilize other data processing toolkit. In the "encoder" column, "weighted bow", "GRU-BRNN", "LSTM-BRNN" and "N-layer Transformer" stand for bag-of-word encoder with weight information, bidirectional gated recurrent RNN encoder, bidirectional long-short term memory RNN encoder and Transformer encoder with *N*-layers, respectively.

Models	Extra Input	Extra Tool	Encoder	Decoder
Re^{3}Sum BiSET	True headline template	Lucene	LSTM-RNN	LSTM-RNN
MASS	WMT News data	-	6-layer Transformer	6-layer Transformer
UniLM	English Wikipedia BookCorpus		24-layer Transformer	24-layer Transformer
PEGASUS PEGASUS + DotProd	C4 HugeNews		12-layer Transformer	12-layer Transformer
ProphetNet	English Wikipedia BookCorpus			
Basic Model	English Wikipedia BookCorpus	-	Electra	6-layer Transformer
MH-NHG	English Wikipedia BookCorpus multiple hypotheses			

Comparison with pre-trained models: We also compare our model with the previous pre-training methods for the headline generation tasks. Utilizing unsupervised pre-trained language models in supervised tasks has become a common practice in NLP. When training a pre-trained language model, the training data and model sizes are the two key factors significantly affecting the model performance. The authors of Pérez-Mayos et al. [33] find that the more the pre-straining data language model is fed, the lower its perplexity. The models pre-trained on more data generally performed better when fine-tuning on downstream tasks. For the model size, it is common knowledge that, under the same model architecture, the deeper the model, the higher the number of the model parameters, and the better the model performance.

MASS is pre-trained on English news data from the WMT News Crawl datasets with 6-layer Transformer architecture. PEGASUS is pre-trained on two large text corpora, C4 [34] and HugeNews, with a 12-layer Transformer architecture. MH-NHG outperforms both MASS and PEGASUS on three ROUGE scores, demonstrating the effectiveness of introducing multi-hypotheses information. MH-NHG utilizes ELECTRA as an encoder pre-trained on English Wikipedia (Wikipedia version: enwiki-20181101) and BookCorpus [35]. UniLM and ProphetNet share the same pre-training datasets as ELECTRA. UniLM and ProphetNet adopt 24- and 12-layer Transformer architectures containing much more model parameters than ours. The better ROUGE-1 and -L scores of MH-NHG when compared to UniLM, and their competitive scores compared to ProphetNet, could further prove its effectiveness.

However, MH-NHG does not achieve as good performance as **PEGASUS+DotProd** and **OFA** on all three metrics. This may be caused by the model design of their work.

The authors of Wang et al. [12] formulate both pretraining and finetuning tasks in a unified sequence-to-sequence abstraction via handcrafted instructions. In the regimes of NLP, language models with prompt instruction tuning prove powerful learners. The authors of Kedia et al. [6] use finite differences to calculate the gradient from the dot-product of gradients, and take this gradient as a regularization technique, boosting the model performance.

4.4. Case Study

We provide two examples for comparison (Table 5). The source article, corresponding reference title, headline generated based on our baseline model, and proposed MH-NHG are listed for each example. In addition, we present four hypotheses generated based on the baseline model adopting the four different sampling methods described in Section 2.3. In the first example, compared to the reference headline, the baseline system-generated headline miss the salient information "00-billion-euro" while the multi-hypotheses-based model-generated headline capture it. An observation of the hypotheses demonstrates that "00-billion-euro" appears in the greedy search hypothesis. This example further verifies that the key messages in the original news article could be emphasized with the help of the multiple hypotheses information, thereby increasing the informativeness of the final headline. In the second example, essential information such as "Sri Lanka", "Jaffna", and "offensive" are contained in the baseline system. Nonetheless, their word order results in entirely distinct meanings. This issue is rectified in the MH-NHG model, showcasing the model's capacity to integrate semantic information from the source article and the soft templates.

Table 5. Examples of the generated templates and headlines by our model. '0' refers to masked numbers. We also perform post-editing to improve readability.

Source article	Ireland's government urged prudence on Thursday as the first Irish savers began to benefit from a state savings scheme that will mean a 00-billion-euro (00-billion-dollar) payout to almost 0.0 million people over the next year.
Reference	Irish urged to continue saving as 00-billion-euro payout begins, by Andrew
Sampling	Irish backtracks as first savings scheme falls
Greedy search	Irish savers obtain 00-billion-euro payout castle, as contributed by reporting
Beam search	Ireland's first savers benefit from state savings scheme
Stochastic beam search	Ireland braces for insure at own savings deal
Baseline	Ireland's first savers benefit from state savings scheme
Our model	Irish government urges prudence as 00-billion-euro savings scheme begins
source article	The shooting down of the largest transport plane in the Sri Lankan air force has wrecked supply lines and slowed a major government offensive against the Tamil rebel citadel of Jaffna, analysts said.
Reference	Downing of plane slows Sri Lanka's army onslaught on Jaffna by Amal Jayasinghe
Sampling	Downing plane may mean help ahead of Jaffna offensive Sri Lanka
Greedy search	Sri Lanka's air force plane crash slows Jaffna offensive side, as contributed by reporting
Beam search	Sri Lankan air force plane crash slows Jaffna offensive
Stochastic beam search	Downing of Sri Lankan major military plane splits supply lines as tigers talk up against Tamil rebels
Baseline	Sri Lankan air force plane crash slows Jaffna offensive
Our model	Downing of plane slows Sri Lanka offensive

In conclusion, the aforementioned examples highlight that the inclusion of multiple hypotheses as soft templates within the NHG system can enhance the system from a specific perspective, such as capturing salient information or improving fluency. This underscores the effectiveness of the model.

5. Related Work

Section 5.1 briefly summarizes the neural-network-based approaches for headline generation. Section 5.2 describes studies related to the template-based methods.

5.1. NHG

The sequence-to-sequence architecture has found success in headline generation, leading to significant attention on NHG. Consequently, numerous efforts have been made to enhance model performance. Early-stage investigations focused on exploring different model architectures [1,3,36]. The impact of fixed vocabulary size on model performance was also explored, resulting in the integration of pointer networks into NHG [2,37,38]. Researchers, such as Kikuchi et al. [39], attempted to control headline length in their systems. Utilizing variational auto-encoders, Zhou et al. [40], Miao and Blunsom [41], Li et al. [42] aimed to capture latent information within headlines. Addressing the issue of random word generation, Cao et al. [43] proposed a dual-attention architecture. Furthermore, Takase and Kiyono [5] introduced perturbations as a regularization technique to mitigate overfitting in neural network models. Kedia et al. [6] introduced a meta-learning algorithm that leverages finite-differences to compute the gradient from the dot-product of gradients, thereby enhancing the model's generalization ability.

In recent years, there has been notable progress in pre-training techniques such as ELMo [44], GPT-2/GPT-3 [45,46], BERT [22], and RoBERTa [47], which have greatly influenced the machine learning and natural language processing communities. These models are initially pre-trained using unsupervised text-to-text objectives on large-scale datasets to capture contextual representations of input data. Subsequently, the acquired knowledge is integrated into downstream tasks. Experimental results have demonstrated significant improvements in various downstream tasks, including summarization [7,9,10]. Additionally, [48] focused on distilling large pre-trained sequence-to-sequence Transformer models (such as T5 [34], BART [10], and PEGASUS [7]) into smaller ones to achieve faster inference with minimal performance loss. Similarly, [49] utilized a multilingual BERT (BERTmultilingual) [22] to initialize their Transformer encoder, constructing a multi-task framework for cross-lingual abstractive summarization in low-resource scenarios. The authors of Li et al. [11] introduced a unified-modal pre-training architecture that utilizes non-paired text corpus and image collections on a large scale to facilitate cross-modal learning. The authors of Wang et al. [12] proposed a unified paradigm for multimodal pre-training framework supporting task comprehensiveness.

5.2. Template-Based Methods

A template-based method is a traditional technique in summarization. A template is typically an incomplete sentence that can be filled with input text with regard to manually defined rules. However, manually constructed templates require a tremendous amount of domain knowledge and are incredibly time consuming. Hence, Cao et al. [15] proposed a soft template-based approach to improve the headline quality. They specifically believed that reference headlines with similar structure and semantics in the dataset can constitute a specific template, and that using these reference headlines as soft templates to guide the model's learning process will improve the generated headline quality. Given an input article and the corresponding reference headline, they first retrieved similar reference headlines from all reference headlines in the training set. They then utilized the top-ranked reference headlines as soft templates as an extra input to the sequence-to-sequence model. Wang et al. [4] shared the same motivation and further improved the model architecture with two selective gates: the template-to-article gate, which aimed to use a template to filter the source article representation, and the article-to-template gate, which attempted to control the proportion of the template-filtered source article representation in the final article representation.

6. Conclusions

In this study, we propose leveraging the NHG model uncertainty to generate multiple hypotheses, using them as soft templates to enhance the model's performance. Besides the conventional evaluation metric, ROUGE, we employ a pre-trained language model-based evaluation metric to assess the model's performance. Results from both evaluations

and detailed case studies confirm the importance of multiple hypotheses in headline generation tasks. The proposed model is straightforward yet outperforms baseline systems significantly. Future extensions include investigating hypothesis reranking methods to filter higher-quality hypotheses and incorporating more sophisticated information retrieval techniques within the NHG framework.

Author Contributions: Conceptualization, Y.A. and Z.L.; methodology, Z.L.; software, Y.A.; validation, Y.A., Z.L. and F.B.; formal analysis, S.D.; investigation, S.D.; resources, F.B.; data curation, S.D.; writing—original draft preparation, Y.A.; writing—review and editing, Z.L.; visualization, Y.A.; supervision, Z.L.; project administration, Z.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Natural Science Foundation of China under Grant No. 62006129, No. 12265020; the Natural Science Foundation of Inner Mongolia under Grant No. 2022MS06001; the Research Program of science and technology at Universities of Inner Mongolia Autonomous Region under Grant No. NJZY21263; the MOE (Ministry of Education in China) Humanities and Social Sciences Foundation under Grant No. 20YJC860005; the Inner Mongolia Autonomous Region Science and Technology Planning Project under Grant No. 2020GG0105.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used to support the findings of this study are available from the first author upon request.

Conflicts of Interest: The authors declare that they have no known competing financial interest or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Rush, A.M.; Chopra, S.; Weston, J. A neural attention model for abstractive sentence summarization. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 379–389.
2. Gu, J.; Lu, Z.; Li, H.; Li, V.O. Incorporating copying mechanism in sequence-to-Sequence learning. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016; pp. 484–494.
3. Ayana; Shen, S.Q.; Lin, Y.K.; hao Tu, C.; Zhao, Y.; Liu, Z.Y.; Sun, M.S. Recent Advances on Neural Headline Generation. *J. Comput. Sci. Technol.* **2017**, *32*, 768–784. [CrossRef]
4. Wang, K.; Quan, X.; Wang, R. BiSET: Bi-directional Selective Encoding with Template for Abstractive Summarization. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 2153–2162.
5. Takase, S.; Kiyono, S. Rethinking Perturbations in Encoder-Decoders for Fast Training. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Online, 6–11 June 2021; pp. 5767–5780.
6. Kedia, A.; Chinthakindi, S.C.; Ryu, W. Beyond Reptile: Meta-Learned Dot-Product Maximization between Gradients for Improved Single-Task Regularization. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event, 16–20 November 2021; pp. 407–420.
7. Zhang, J.; Zhao, Y.; Saleh, M.; Liu, P.J. PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization. In Proceedings of the Thirty-seventh International Conference on Machine Learning, Virtual, 13–18 July 2020.
8. Dong, L.; Yang, N.; Wang, W.; Wei, F.; Liu, X.; Wang, Y.; Gao, J.; Zhou, M.; Hon, H.W. Unified Language Model Pre-training for Natural Language Understanding and Generation. In Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, BC, Canada, 8–14 December 2019; pp. 13063–13075.
9. Yan, Y.; Qi, W.; Gong, Y.; Liu, D.; Duan, N.; Chen, J.; Zhang, R.; Zhou, M. ProphetNet: Predicting Future N-gram for Sequence-to-Sequence Pre-training. *arXiv* **2020**, arXiv:2001.04063.
10. Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; Zettlemoyer, L. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; pp. 7871–7880.
11. Li, W.; Gao, C.; Niu, G.; Xiao, X.; Liu, H.; Liu, J.; Wu, H.; Wang, H. UNIMO: Towards Unified-Modal Understanding and Generation via Cross-Modal Contrastive Learning. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Virtual Event, 1–6 August 2021; pp. 2592–2607.
12. Wang, P.; Yang, A.; Men, R.; Lin, J.; Bai, S.; Li, Z.; Ma, J.; Zhou, C.; Zhou, J.; Yang, H. OFA: Unifying Architectures, Tasks, and Modalities Through a Simple Sequence-to-Sequence Learning Framework. *arXiv* **2022**, arXiv:2202.03052 .

13. Koehn, P.; Knowles, R. Six Challenges for Neural Machine Translation. In Proceedings of the First Workshop on Neural Machine Translation, Vancouver, BC, Canada, 30 July–4 August 2017.
14. Liang, Z.; Hovy, E. Template-filtered headline summarization. In *Acl Workshop Text Summarization Branches out*; Association for Computational Linguistics: Barcelona, Spain, 2004 .
15. Cao, Z.; Li, W.; Li, S.; Wei, F. Retrieve, Rerank and Rewrite: Soft Template Based Neural Summarization. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, 15–20 July 2018; pp. 152–161.
16. Fomicheva, M.; Specia, L.; Guzmán, F. Multi-Hypothesis Machine Translation Evaluation. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; pp. 1218–1232.
17. Wang, S.; Liu, Y.; Wang, C.; Luan, H.; Sun, M. Improving Back-Translation with Uncertainty-based Confidence Estimation. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 791–802.
18. Liu, Z.; Yi, X.; Sun, M.; Yang, L.; Chua, T.S. Neural Quality Estimation with Multiple Hypotheses for Grammatical Error Correction. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Online, 6–11 June 2021; pp. 5441–5452.
19. Ott, M.; Auli, M.; Grangier, D.; Ranzato, M. Analyzing Uncertainty in Neural Machine Translation. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018.
20. Napoles, C.; Gormley, M.; Van Durme, B. Annotated Gigaword. In Proceedings of the Proc. the Joint Workshop on Automatic Knowledge Base Construction and Web-Scale Knowledge Extraction, Montreal, QC, Canada, 7–8 June 2012; pp. 95–100.
21. Lin, C.Y. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the Association for Computational Linguistics 04 Workshop*; Association for Computational Linguistics: Barcelona, Spain, 2004 ; Volume 8.
22. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, 2–7 June 2019; pp. 4171–4186.
23. Cui, Y.; Chen, Z.; Wei, S.; Wang, S.; Liu, T.; Hu, G. Attention-over-Attention Neural Networks for Reading Comprehension. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Minneapolis, MI, USA, 4 June 2017; pp. 593–602.
24. Ba, J.L.; Kiros, J.R.; Hinton, G.E. Layer normalization. *arXiv* **2016**, arXiv:1607.06450.
25. Shen, S.; Cheng, Y.; He, Z.; He, W.; Wu, H.; Sun, M.; Liu, Y. Minimum Risk Training for Neural Machine Translation. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016; pp. 1683–1692.
26. Vijayakumar, A.K.; Cogswell, M.; Selvaraju, R.R.; Sun, Q.; Lee, S.; Crandall, D.; Batra, D. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv* **2016**, arXiv:1610.02424.
27. Kool, W.; van Hoof, H.; Welling, M. Stochastic Beams and Where to Find Them: The Gumbel-Top-k Trick for Sampling Sequences Without Replacement. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 10–15 June 2019.
28. Song, K.; Tan, X.; Qin, T.; Lu, J.; Liu, T.Y. MASS: Masked Sequence to Sequence Pre-training for Language Generation. In Proceedings of the Thirty-Sixth International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019.
29. Zhang, T.; Kishore, V.; Wu, F.; Weinberger, K.Q.; Artzi, Y. BERTScore: Evaluating Text Generation with BERT. In Proceedings of the International Conference on Learning Representations, Addis Ababa, Ethiopia, 26–30 April 2020.
30. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.u.; Polosukhin, I. Attention is All you Need. In Proceedings of the Advances in Neural Information Processing Systems 30, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
31. Clark, K.; Luong, M.T.; Le, Q.V.; Manning, C.D. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. *arXiv* **2020**, arXiv:2003.10555.
32. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2015**, arXiv:1412.6980.
33. Pérez-Mayos, L.; Ballesteros, M.; Wanner, L. How much pretraining data do language models need to learn syntax? In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Punta Cana, Dominican Republic, 10 May 2021; pp. 1571–1582.
34. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* **2020**, *21*, 1–67.
35. Zhu, Y.; Kiros, R.; Zemel, R.; Salakhutdinov, R.; Urtasun, R.; Torralba, A.; Fidler, S. Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 19–27.
36. Chopra, S.; Auli, M.; Rush, A.M.; Harvard, S. Abstractive sentence summarization with attentive recurrent neural networks. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics, San Diego, CA, USA, 12–17 June 2016; pp. 93–98.
37. Gulcehre, C.; Ahn, S.; Nallapati, R.; Zhou, B.; Bengio, Y. Pointing the Unknown Words. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016; pp. 484–494.

38. Cao, Z.; Luo, C.; Li, W.; Li, S. Joint Copying and Restricted Generation for Paraphrase. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–7 February 2017.
39. Kikuchi, Y.; Neubig, G.; Sasano, R.; Takamura, H.; Okumura, M. Controlling Output Length in Neural Encoder-Decoders. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, TX, USA, 1–5 November 2016; pp. 1328–1338.
40. Zhou, Q.; Yang, N.; Wei, F.; Zhou, M. Selective Encoding for Abstractive Sentence Summarization. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2017; pp. 1095–1104.
41. Miao, Y.; Blunsom, P. Language as a Latent Variable: Discrete Generative Models for Sentence Compression. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, TX, USA, 1–5 November 2016; pp. 319–328.
42. Li, P.; Lam, W.; Bing, L.; Wang, Z. Deep Recurrent Generative Decoder for Abstractive Text Summarization. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 7–11 September 2017; pp. 2091–2100.
43. Cao, Z.; Wei, F.; Li, W.; Li, S. Faithful to the Original: Fact Aware Neural Abstractive Summarization. In Proceedings of the 32nd AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; pp. 4784–4791.
44. Peters, M.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; Zettlemoyer, L. Deep Contextualized Word Representations. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, LA, USA, 1–6 June 2018; pp. 2227–2237.
45. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language Models are Unsupervised Multitask Learners. *OpenAI Blog* **2019**, *1*, 9.
46. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language Models are Few-Shot Learners. In Proceedings of the Advances in Neural Information Processing Systems, Online, 6–12 December 2020; Volume 33, pp. 1877–1901.
47. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv* **2019**, arXiv:1907.11692.
48. Zhang, S.; Zhang, X.; Bao, H.; Wei, F. Attention Temperature Matters in Abstractive Summarization Distillation. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, Dublin, Ireland, 22–27 May 2022; pp. 127–141.
49. Bai, Y.; Gao, Y.; Huang, H.Y. Cross-Lingual Abstractive Summarization with Limited Parallel Resources. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Virtual Event, 1–6 August 2021; pp. 6910–6924.