Deep Differential Amplifier for Extractive Summarization

Ruipeng Jia[1,2], **Yanan Cao**[1,2], **Fang Fang**[1,2]*, **Yuchen Zhou**[1], **Zheng Fang**[1], **Yanbing Liu**[1,2] and **Shi Wang**[3]*
[1]Institute of Information Engineering, Chinese Academy of Sciences
[2]School of Cyber Security, University of Chinese Academy of Sciences
[3]Institute of Computing Technology, Chinese Academy of Sciences
[1,2]{jiaruipeng, caoyanan, fangfang0703, zhouyuchen, fangzheng, liuyanbing}@iie.ac.cn
[3]wangshi@ict.ac.cn

Abstract

For sentence-level extractive summarization, there is a disproportionate ratio of selected and unselected sentences, leading to flatting the summary features when optimizing the classification. The imbalanced sentence classification in extractive summarization is inherent, which can't be addressed by data sampling or data augmentation algorithms easily. In order to address this problem, we innovatively consider the single-document extractive summarization as a *rebalance problem* and present a deep differential amplifier framework to enhance the features of summary sentences. Specifically, we *calculate and amplify* the semantic difference between each sentence and other sentences, and apply the *residual unit* to deepen the differential amplifier architecture. Furthermore, the corresponding objective loss of the minority class is boosted by a weighted cross-entropy. In this way, our model pays more attention to the pivotal information of one sentence, that is different from previous approaches which model all informative context in the source document. Experimental results on two benchmark datasets show that our summarizer performs competitively against state-of-the-art methods. Our source code will be available on Github.

1 Introduction

Single-document extractive summarization forms summary by copying and concatenating the most important spans (usually sentences) in a document. Sentence-level summarization is a very challenging task, because it arguably requires an in-depth understanding of the source document sentences, and current automatic solutions are still far from human performance. Recent approaches frame the task as a sequence labeling problem, taking advantage of the success of neural network architectures.

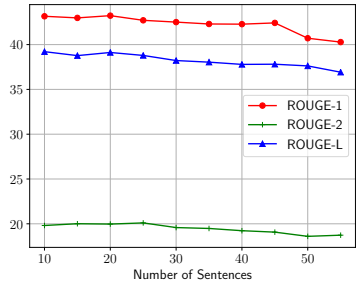


Figure 1: ROUGE score for documents with different length. The result is calculated on the test set of CNN/DM and the trained model is based on BERT.

However, there are still two inherent obstacles for sentence-level extractive summarization:

1) **It should be detrimental to keep tangential information** (West et al., 2019). The intuitive limitation of those approaches is that they always prefer to model and retain all informative content from the source document. This goes against the fundamental goal of summarization, which crucially needs to forget all but the "pivotal" information. Recently, the Information Bottleneck principle (Tishby et al., 2000; West et al., 2019) is introduced to incorporate a tradeoff between information selection and pruning. Length penalty and the topic loss (Baziotis et al., 2019) are used in the autoencoding system to augment the reconstruction loss. However, these methods require external variables or augmentative terms, without enhancing the representation of pivotal information.

2) **Imbalanced classes inherently result in models that have poor predictive performance, specifically for the minority class.** The distribution of examples across the known classes can vary from a slight bias to a severe imbalance, where there is one example in the minority class for dozens of examples in the majority class. For instance, according to the statistics on the popular summarization dataset, only 7.33% sentences of

*Corresponding authors: Fang Fang and Shi Wang

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 366–376
August 1–6, 2021.

CNN/DM (Hermann et al., 2015) are labeled as "1" and others are "0", indicating whether this sentence should be selected as summary or not. Conversely, most machine learning algorithms for classification predictive models are designed and demonstrated on problems that assume an equal distribution of classes. This means that a naive application of a model may only focus on learning the characteristics of the abundant observations, neglecting the examples from the minority class. Furthermore, as shown in Figure 1, the ROUGE score gradually declines along with the number of sentences accumulating, since the valuable summary sentences is generally a tiny minority (with the quantity of 1-4), while more and more majority sentences will swamp the minority ones. Unfortunately, the imbalance in summarization is inherent, which can't be addressed by common data augmentation (He and Ma, 2013; Asai and Hajishirzi, 2020; Min et al., 2020; Zoph et al., 2019; Xie et al., 2020), for there is a rare influence on the 0/1 distribution by adding or deleting the entire document.

These two obstacles are interrelated and interact with each other. Highlighting the pivotal information will strengthen the unique semantic and weaken the common informative content. Additionally, a more balanced distribution would make minority class more attractive. If we can't resolve the category imbalance problem in extractive summarization by data augmentation, how to make the minority class more attractive? Inspired by the differential amplifier of analog electronics[1], we propose a heuristic model, **DifferSum**, as shorthand for **Differ**ential Amplifier for Extractive **Sum**marization to enhance the representation of the summary sentences. Specifically, we calculate and amplify the semantic difference between each sentence and other sentences, by the subtraction operation. The original differential amplifier consists of two terms and the second term is used to avoid making the final output zero. In our model, we use the residual unit instead of the second term to make the architecture deeper. We further design a more appropriate objective function to avoid biasing the data, by making the loss of a minority much greater than the majority. DifferSum shows superiority over other extractive methods in two aspects: 1) enhancing the representation of the pivotal information and 2) compensating the minority class and penalizing the majority ones.

[1]https://en.wikipedia.org/wiki/Differential_amplifier

Experimental results validate the effectiveness of DifferSum. The human evaluation also shows that our model is better in relevance compared with others. Our contributions in this work are concluded as follows:

- We propose a novel conceptualization of extractive summarization as rebalance problem.
- We introduce a heuristic approach, calculating and amplifying the semantic representation of pivotal information by integrating both the differential amplifier and residual learning.
- Our proposed framework has achieved superior performance compared with strong baselines.

2 Related Work

2.1 Extractive Summarization

Recent research work on extractive summarization spans a large range of approaches. These works usually instantiate their encoder-decoder architecture by choosing RNN (Nallapati et al., 2017; Zhou et al., 2018), Transformer (Wang et al., 2019; Zhong et al., 2019b; Liu and Lapata, 2019; Zhang et al., 2019b) or GNN (Wang et al., 2020; Jia et al., 2020b) as encoder, autoregressive (Jadhav and Rajan, 2018; Liu and Lapata, 2019) or RL-based (Narayan et al., 2018; Arumae and Liu, 2018; Luo et al., 2019) decoders. For two-stage summarization, Chen and Bansal (2018) and Bae et al. (2019) follow a hybrid extract-then-rewrite architecture, with policy-based RL to bridge the two networks together. Lebanoff et al. (2019), Xu and Durrett (2019) and Mendes et al. (2019) focus on the extract-then-compress learning paradigm, which will first train an extractor for content selection. Zhong et al. (2020) introduces extract-then-match framework, which employs BERTSUMEXT (Liu and Lapata, 2019) as first-stage to prune unnecessary information. However, these above extractive approaches prefer to model all source informative context and they pay little attention to the imbalance problem.

2.2 Deep Residual Learning

The original deep residual learning is introduced in image recognition (He et al., 2016a) for the notorious degradation problem. Then, residual is introduced to the natural language process by Transformer (Vaswani et al., 2017). Essentially, we cannot determine the depth of the network very well

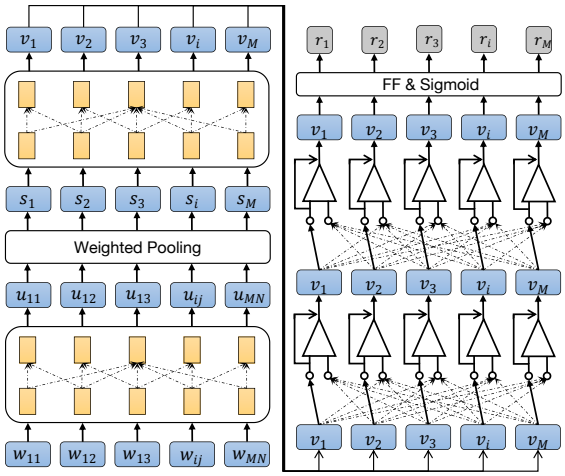


Figure 2: Overview of DifferSum.

when building a deep network. There will be optimal layers in the network, and outside the optimal layer is the redundant layer. We expect the redundant layer to correspond to the input and output, namely identity mapping (He et al., 2016a,b; Veit et al., 2016; Balduzzi et al., 2018). Resnet (He et al., 2016a) addresses the degradation problem by introducing a deep residual learning framework. If an identity mapping were optimal, it would be easier to push the residual to zero than to fit an identity mapping by a stack of nonlinear layers (Huang and Wang, 2017). In this paper, the residual unit serves as the second item of the differential amplifier to keep our architecture deep enough and capture pivotal information.

3 Methodology

3.1 Problem Definition

We model the sentence extraction task as a sequence tagging problem (Kedzie et al., 2018). Given a document D consisting of a sequence of M sentences $[s_1, s_2, ..., s_M]$ and a sentence s_i consisting of a sequence of N words $[w_{i1}, w_{i2}, ..., w_{iN}]$. We denote by h_i and h_{ij} the embedding of sentences and words in a continuous space. The extractive summarizer aims to produce a summary $\mathcal{S}$ by selecting m sentences from D (where $m \leq M$). For each sentence $s_i \in D$, there is ground-truth $y_i \in \{0, 1\}$ and we will predict a label $\hat{y}_i \in \{0, 1\}$, where 1 means that s_i should be included in the summary. We assign a score $p(\hat{y}_i|s_i, D, \theta)$ to quantify s_i's relevance to the summary, where θ is the parameters of neural network model. Finally, we assemble a summary $\mathcal{S}$ by selecting m sentences, according to the probability of $p(1|s_i, D, \theta)$.

3.2 Sentence Encoder

The sentence encoder in extractive summarization models is usually a recurrent neural network with Long-Short Term Memory (Hochreiter and Schmidhuber, 1997) or Gated Recurrent Units (Cho et al., 2014). In this paper, our sentence encoder builds on the BERT architecture (Devlin et al., 2019), a recently proposed highly efficient model which is based on the deep bidirectional Transformer (Vaswani et al., 2017) and has achieved state-of-the-art performance in many NLP tasks. The Transformer aims at reducing the fundamental constraint of sequential computation which underlies most architecture (Liu et al., 2019). It eliminates recurrence in favor of applying a self-attention mechanism which directly models relationships between all words in a sentence.

Our extractive model is composed of a sentence-level Transformer ($\mathcal{T}_S$) and a document-level Transformer ($\mathcal{T}_D$) (Liu et al., 2019). For each sentence s_i in the input document, $\mathcal{T}_S$ is applied to obtain a contextual representation for each word:

$$[u_{11}, u_{12}, ..., u_{MN}] = \mathcal{T}_S([w_{11}, w_{12}, ..., w_{MN}]) \quad (1)$$

And the representation of a sentence is acquired by applying weighted-pooling:

$$\begin{aligned} a_{ij} &= \mathbf{W}_0 u_{ij}^T \\ s_i &= \frac{1}{N} \sum_{j=1}^{N} a_{ij} u_{ij} \end{aligned} \quad (2)$$

Document-level transformer $\mathcal{T}_D$ takes s_i as input and yields a contextual representation for each sentence:

$$[v_1, v_2, ..., v_M] = \mathcal{T}_D([s_1, s_2, ..., s_M]) \quad (3)$$

3.3 Deep Differential Amplifier

In the Transformer model sketched above, inter-sentence relations are modeled by multi-head attention based on softmax functions, which only capture shallow structural information (Liu et al., 2019).

A differential amplifier is a type of electronic amplifier that amplifies the difference between two input voltages but suppresses any voltage common

to the two inputs. The output of an ideal differential amplifier is given by:

$$V_{out} = \mathbf{A}_d(V_{in}^+ - V_{in}^-) \tag{4}$$

where V_{in}^+ and V_{in}^- are the input voltage; $\mathbf{A}_d$ is the *differential-mode gain*.

In practice, the gain should not be quite equal for the two inputs, V_{in}^+ and V_{in}^-. For instance, even if V_{in}^+ and V_{in}^- are equal, the output V_{out} should not be zero. So, modern differential amplifiers are usually implemented with a more realistic expression, which includes a second term:

$$V_{out} = \mathbf{A}_d(V_{in}^+ - V_{in}^-) + \mathbf{A}_c\frac{V_{in}^+ + V_{in}^-}{2} \tag{5}$$

where $\mathbf{A}_c$ is called the *common-mode gain* of the amplifier.

Inspired by the differential amplifier above, we calculate and amplify the semantic difference between each sentence and other sentences by the subtraction operation of the sentence representations $[v_1, v_2, ..., v_M]$. Particularly, for sentence s_i, V_{in}^+ and V_{in}^- are calculated as follows:

$$\begin{aligned} V_{in}^+ &= v_i \\ V_{in}^- &= \frac{\sum_{j\in\{1,2,...,M\}\setminus\{i\}} v_j}{M-1} \end{aligned} \tag{6}$$

The original differential amplifier consists of two terms and the second one avoids making the final output zero. While for the deep neural network: 1) inputs of the differential amplifier are vector instances in the high dimensional space, which is practically impossible for the zero output, compared with scalar; 2) the second term of the differential amplifier is not suitable for the deep iterative architecture, since it is exposed to the degradation problem.

Notably, residual learning is introduced in deep learning as shortcut connections to skip one or more layers, which is naturally an alternative to the second item of the differential amplifier. The advantages of this method are: 1) the residual architecture will highlight the pivotal information as well as reserving the original sentence representation; 2) it is easier to optimize the residual mapping than to optimize the original (He et al., 2016a). Hence, the residual unit is employed as the second item, along with an iterative refinement algorithm to enhance the final representation of sentences.

3.4 Residual Representation for Sentence

The differential amplifier in our architecture consists of a few stacked layers to iteratively refine the pivotal representation. Let us consider $\mathcal{H}(x)$ as an underlying mapping to be fit, with x denoting the inputs to the first of these layers. Since multiple nonlinear layers can asymptotically approximate complicated functions (He et al., 2016a; Montúfar et al., 2014), the differential amplifier mapping $\mathcal{H}(x)$ is recast into a residual mapping $\mathcal{F}(x)$ and an identity mapping x:

$$\mathcal{H}(x) = \mathcal{F}(x) + x \tag{7}$$

Obviously, residual learning is just a variant of the differential amplifier:

$$\begin{aligned} \mathcal{H}(x) &:= V_{out} \\ \mathcal{F}(x) &:= \mathbf{A}_d(V_{in}^+ - V_{in}^-) \end{aligned} \tag{8}$$

where the output voltage V_{out} thus becomes the original mapping $\mathcal{H}(x)$ and the first item of amplifier $\mathbf{A}_d(V_{in}^+ - V_{in}^-)$ equals to residual mapping $\mathcal{F}(x)$,

In our model, the second item of the differential amplifier is replaced by the identity mapping x, which is the shortcut connection and the output is added to the outputs of $\mathcal{F}(x)$. Furthermore, 1) the identity shortcut connections advance the architecture without extra parameter; 2) the identity shortcut doesn't add the computational complexity (He et al., 2016a);

Thus, for sentence respresentation v_i, the deep differential amplifier is:

$$\mathcal{H}(v_i) = \mathbf{A}_d(v_i - \frac{\sum_{j\in\{1,2,...,M\}\setminus\{i\}} v_j}{M-1}) + v_i \tag{9}$$

3.5 Iterative Structure Refinement

The differential amplifier and residual unit specialize in modeling the pivotal information, while deeper neural networks with more parameters are able to infer semantic more accurately. So, an iterative refinement algorithm is introduced to enhance the final representation of pivotal information. For sentence v_i, the fundamental iterative unit is:

$$\begin{aligned} \mathcal{H}(v_i) &= \mathcal{F}(v_i) + v_i \\ v_i &= \mathcal{H}(v_i) \end{aligned} \tag{10}$$

where we iteratively refine the representation v_i for K times; and thanks to the built-in residual mechanism, most shorter paths are needed during training, as longer paths do not contribute any gradient.

Along with the supervision, each iteration will pay more attention to the key semantic difference $\mathcal{F}(v_i)$ of sentences with label 1, while trying to zero other $\mathcal{F}(v_j)$. Conversely, previous extractive approaches without differential amplifier can only classify those sentences by compensating or penalizing v_i / v_j, which is more difficult to model.

Following previous work (Nallapati et al., 2017; Liu et al., 2019), we use a sigmoid function after a linear transformation to calculate the probability r_i of selecting s_i as a summary sentence:

$$r_i = \text{sigmoid}(\mathbf{W}_1 v_i^T) \tag{11}$$

3.6 Weighted Objective Function

To rebalance the bias of minority 1-class and majority 0-class, we have built a deep differential amplifier to amplify and capture the unique information for summary sentences. Besides, another heuristic method is to make our model pay more attention to 1-class: a weighted cross-entropy function.

Particularly, we further design a more appropriate objective function to avoid biasing the data, by making the loss of a minority much greater than the majority. The weight we employed is to rebalance the observations for each class, so the sum of observations for each class are equal. Finally, we define the model's loss function as the summation of the losses of all iterations:

$$L = \sum_{k=1}^{K} \left\{ \frac{1}{M} \sum_{i=1}^{M} \left[\frac{\sum_{s_j \in D} \mathbb{I}(s_j \notin \mathcal{S})}{\sum_{s_j \in D} \mathbb{I}(s_j \in \mathcal{S})} y \log(r_i^k) + (1-y) \log(1-r_i^k) \right] \right\} \tag{12}$$

where $\mathbb{I}(\cdot)$ is an indicator function and K is the number of iterations.

4 Experiments

4.1 Datasets

As shown in Table 1, we employ two datasets widely-used with multiple sentences summary: CNN and Dailymail (CNN/DM) (Hermann et al., 2015) and New York Times (NYT) (Sandhaus, 2008).

Table 1: Data Statistics: CNN/Daily Mail and NYT.

Datasets	avg.doc length		avg.summary length	
	words	sentences	words	sentences
CNN	760.50	33.98	45.70	3.59
DailyMail	653.33	29.33	54.65	3.86
NYT	800.04	35.55	45.54	2.44

CNN/DM We used the standard split (Hermann et al., 2015) for training, validation, and test (90,266/1,220/1,093 for CNN and 196,96/12.148/10,397 for Daily Mail), with splitting sentences by Stanford CoreNLP (Manning et al., 2014) toolkit and pre-processing the dataset following (See et al., 2017) and (Zhong et al., 2020). This dataset contains news articles and several associated abstractive highlights. We use the unanonymized version as in previous summarization work and each document is truncated to 800 BPE tokens.

NYT Following previous work (Zhang et al., 2019b; Xu and Durrett, 2019), we use 137,778, 17,222 and 17,223 samples for training, validation, and test, respectively. We also followed their filtering procedure, documents with summaries less than 50 words were removed from the dataset. Sentences were split with the Stanford CoreNLP toolkit (Manning et al., 2014). Input documents were truncated to 800 BPE tokens too.

4.2 Parameters

Our code is based on Pytorch (Paszke et al., 2019) and the pre-trained model employed in DifferSum is 'albert-xxlarge-v2', which is based on the huggingface/transformers[2]. We train DifferSum two days for 100,000 steps on 2GPUs(Nvidia Tesla V100, 32GB) with gradient accumulation every two steps. Adam with $\beta_1 = 0.9, \beta_2 = 0.999$ is used as optimizer. Learning rate schedule follows the strategy with warming-up on first 10,000 steps. We have tried the iteration steps of $2/4/6/8$ for iterative refinement, and $K = 4$ is the best choice based on the validation set. We select the top-3 checkpoints based on the evaluation loss on the validation set, and report the averaged results on the test set.

Following Jia et al. (2020a) and Jia et al. (2021), we employ the greedy algorithm for the sentence-level soft labels, which falls under the umbrella

[2]https://github.com/huggingface/transformers

Table 2: ROUGE F1 on CNN/DM.

Models	CNN/DM R-1	R-2	R-L
Abstractive			
ABS (2015)	35.46	13.30	32.65
PGC (2017)	39.53	17.28	36.38
TransformerABS (2017)	40.21	17.76	37.09
$T5_{Large}$ (2020)	43.52	**21.55**	40.69
$BART_{Large}$ (2019a)	44.16	21.28	40.90
$PEGASUS_{Large}$ (2019a)	44.17	21.47	41.11
$ProphetNet_{Large}$ (2020)	**44.20**	21.17	**41.30**
Extractive			
Lead-3	40.42	17.62	36.67
Oracle (Sentence)	55.61	32.84	51.88
SummaRuNNer (2017)	39.60	16.20	35.30
Exconsumm (2019)	41.70	18.60	37.80
$PNBERT_{Base}$ (2019a)	42.69	19.60	38.85
$HIBERT_{Large}$ (2019b)	42.37	19.95	38.83
$BERT\text{-}ext\text{+}RL_{Base}$ (2019)	42.76	19.87	39.11
$BERTSUMEXT_{Base}$ (2019)	43.25	20.24	39.63
$BERTSUMEXT_{Large}$ (2019)	43.85	20.34	39.90
$DiscoBERT_{Base}$ (2020)	43.77	20.85	40.67
HSG_{Base} (2020)	42.95	19.76	39.23
$ETCSum_{Base}$ (2020)	43.84	20.80	39.77
$ARedSum_{Base}$ (2020)	43.43	20.44	39.83
$MATCHSUM_{Base}$ (2020)	44.41	20.86	40.55
DifferSum$_{Large}$	**44.70**	**21.36**	**40.83**

Table 3: ROUGE F1 on NYT.

Models	NYT R-1	R-2	R-L
Abstractive			
ABS (2015)	42.78	25.61	35.26
PGC (2017)	43.93	26.85	38.67
TransformerABS (2017)	45.36	27.34	39.53
$BART_{Large}$ (2019a)	48.73	29.25	44.48
Extractive			
Lead-3	41.80	22.60	35.00
Oracle (Sentence)	64.22	44.57	57.27
SummaRuNNer (2017)	42.37	23.89	38.74
Exconsumm (2019)	43.18	24.43	38.92
JECS (2019)	45.50	25.30	38.20
$BERTSUMEXT_{Base}$ (2019)	46.66	26.35	42.62
$HIBERT_{Large}$ (2019b)	49.47	**30.11**	41.63
DifferSum$_{Large}$	**49.52**	29.78	**43.86**

of subset selection. Besides, we employ the Trigram Blocking strategy for decoding, which is a simple but powerful version of Maximal Marginal Relevance (Carbonell and Goldstein, 1998). Specifically, when predicting summaries for a new document, we first use the model to obtain the probability score $p(1|s_i, D, \theta)$ for each sentence, and then we rank sentences by their scores and discard those which have trigram overlappings with their predecessors.

4.3 Metric

ROUGE (Lin, 2004) is the standard metric for evaluating the quality of summaries. We report the ROUGE-1, ROUGE-2, and ROUGE-L of DifferSum by *ROUGE-1.5.5.pl*, which calculates the overlap lexical units of extracted sentences and ground-truth.

5 Results and Analysis

5.1 Results on CNN/DM

Table 2 shows the results on CNN/DailyMail. All of these scores are in accordance with original papers. Following Nallapati et al. (2017); Liu and Lapata (2019), we compare extractive summarization models against abstractive models, and it is certainly that the abstractive paradigm is still on the frontier of summarization. The first part of extractive approaches is the Lead-3 baseline and Oracle upper bound, while the second part includes other extractive summarization models. We present our models finally at the bottom. It is obvious that our DifferSum outperforms all extractive baseline models. Compared with large version BERTSUMEXT, our DifferSum achieves 0.85/1.02/0.93 improvements on R-1, R-2, and R-L, which indicates the pivotal information captured by the differential amplifier is more powerful than the other structures. Compared with early approaches, especially for BERTSUMEXT, we observe that BERT outperforms all previous non-BERT-based summarization systems, and Trigram-Blocking leads to a great improvement on all ROUGE metrics. MATCHSUM is a comparable competitor to our DifferSum, which formulates the extractive summarization task as a two-step problem and extract-then-match summary based on a well-trained BERTSUMEXT. Therefore, we only train a large version DifferSum for a fair comparison.

5.2 Results on NYT

Results on NYT are summarized in Table 3. Note that we use limited-length ROUGE recall as Durrett et al. (2016), where the selected sentences are truncated to the length of the human-written summaries. The parts of Table 3 is similar to Table 2. The first four lines are abstractive models, and the next two lines are our golden baselines for extrac-

Table 4: Ablation Study on CNN/DM.

Models	R-1	R-2	R-L
DifferSum	**44.70**	**21.36**	**40.83**
DifferSum w/o ALBERT	44.41	20.80	40.57
DifferSum w/o Amplifier	44.17	20.74	40.42
DifferSum w/o Iteration	44.32	21.02	40.48

Table 5: Human Evaluation on CNN/DM.

Models	1st	2nd	3rd	4th	MeanR
SummaRuNNer	0.20	0.27	0.30	0.23	2.56
BERTSUMEXT	0.25	0.30	0.28	0.17	2.37
DifferSum	0.48	0.27	0.20	0.05	1.82
Ground-Truth	0.68	0.22	0.07	0.03	1.45

tive summarization. The third part reports the performance of other extractive works and our model respectively. Again, we observe that our differential amplifier modeling performs better than both LSTM and BERT. Meanwhile, we find that extractive approaches show superiority over abstractive models, and the ROUGE scores are higher than CNN/DailyMail.

5.3 Ablation Studies

We propose several strategies to improve the performance of extractive summarization, including differential amplifier (vs. normal residual network), pre-trained ALBERT(vs. BERT), and iterative refinement (vs. None). To investigate the influence of these factors, we conduct experiments and list the results in Table 4. Significantly, 1) differential amplifier is more critical than ALBERT, for the reason that the pivotal information is essential and difficult for ALBERT to model; 2) iterative refinement mechanism enlarges the advantage of the differential amplifier, demonstrating the superiority of deep architecture.

5.4 Human Evaluation for Summarization

It is not enough to only rely on the ROUGE evaluation for a summarization system, although the ROUGE correlates well with human judgments (Owczarzak et al., 2012). Therefore, we design an experiment based on a ranking method to evaluate the performance of DifferSum by humans. Following Cheng and Lapata (2016), Narayan et al. (2018) and Zhang et al. (2019b), firstly, we randomly select 40 samples from CNN/DM test set. Then the human participants are presented with one original document and a list of corresponding summaries produced by different model systems. Participants are requested to rank these summaries (ties allowed) by taking informativeness (Can the summary capture the important information from the document) and fluency (Is the summary grammatical) into account. Each document is annotated by three different participants separately.

The input article and ground truth summaries are also shown to the human participants in addition to the three model summaries (SummaRuNNer, BERTSUMEXT, and DifferSum). From the results shown in Table 5, it is obvious that DifferSum is better in relevance compared with others.

5.5 Trigram Blocking Strategy

Trigram Blocking leads to a great improvement on all ROUGE metrics for many extractive approaches (Liu and Lapata, 2019; Wang et al., 2020). It is has become a fundamental module in extractive summarization. In this paper, DifferSum extracts summary sentences with the Trigram-Blocking algorithm, but whether there is a great improvement along with it, like in SummaRuNNer or BERTSUMEXT?

It has been explained by Nallapati et al. (2017); Liu and Lapata (2019), that picking all sentences by comparing the predicted probability with a threshold may not be an optimal strategy since the training data is very *imbalanced* in terms of summary-membership of sentences. Therefore, the Trigram-Blocking algorithm is introduced to select top-k sentences and reduce the *redundancy*.

Coincidentally, our DifferSum is designed to 1) rebalance the distribution of majority and minority and 2) filter the tangential and redundant information. Thus, the Trigram-Blocking algorithm may be useless for our DifferSum.

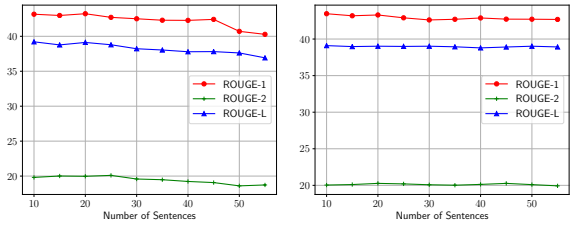
Table 6 further summarizes the performance gain of Trigram-Blocking strategy. It is obvious that this strategy is essential for BERTSUMEXT or SummaRuNNer, achieving more than 2.68 / 0.98 improvements on R-1 separately, for that there is no enough redundancy modeling for both of them. While on the other hand, the efficiency of the Trigram-Blocking strategy is weak for DifferSum.

5.6 Documents with a Different Number of Sentences

In this paper, we emphasize the inherent imbalance problem of the majority 0-class and the minority 1-class. In fact, in CNN/DailyMail dataset, there are plenty of documents with a different num-

Table 6: ROUGE Scores about Trigram-Blocking on CNN/DM Test Set.

Models	R-1	R-L
DifferSum (with Trigram-Blocking)	**44.70**	**40.83**
DifferSum	44.36	40.43
BERTSUMEXT (with Trigram-Blocking)	43.85	39.90
BERTSUMEXT	41.17	36.52
SummaRuNNer (with Trigram-Blocking)	40.58	36.61
SummaRuNNer	39.60	35.30



(a) BERTSUMEXT (b) DifferSum

Figure 3: Comparison Between the ROUGE Scores Tendencies of BERTSUMEXT and DifferSum

ber of sentences, ranging from 3-sentences to 100-sentences. While the number of summary sentences, labeled with 1, is from 1-sentences to 5-sentences, and the average number of sentences labeled 1 in CNN/DailyMail is only 7.33%. What is worse is that the distribution of the number of sentences for documents is a uniform distribution, thus we could not avoid the imbalance by cleaning the data.

In this paper, we design another experiment to analysis the harmful effect of imbalance classes. We train the BERTSUMEXT (12-layers) from scratch on CNN/DailyMail, and evaluate the model on the test set to check the tendency of ROUGE scores, along with the number of sentences accumulating. The result is shown in the line chart of Figure 1 and Figure 3a, and obviously we only pay attention to the document in which the number of sentences less than 55. Specifically, each document is truncated to 2000 BPE tokens to involve more sentences, but this can not cover those whole documents with more than 55-sentences. Therefore, we choose to calculate the ROUGE scores for documents with sentences from 3 to 55.

For comparison, we train our DifferSum (12-layers) from scratch, and each document is truncated to 2000 BPE tokens too. The tendency of our DifferSum is as Figure 3b. Compared with the tendency of BERTSUMEXT, there is no obvious ROUGE decrease, demonstrating that our approach has strengthened the representation of pivotal and rebalanced the disproportionate ratio of summary sentences and other sentences.

Note that more truncated BPE tokens will increase the final average ROUGE slightly, for it may lose some summary sentences when truncating too many tokens. Unfortunately, our 24-layers DifferSum can only be trained with 800 BPE tokens for the limitation of GPU source.

5.7 Map Words Representation into Sentence Representation

A key issue motivating the sentence-level Transformer ($\mathcal{T}_S$) and the document-level Transformer ($\mathcal{T}_D$) is that the features for words after the $\mathcal{T}_S$ might be at different scales or magnitudes. This can be due to some words having very sharp or very distributed attention weights when summing over the features of the other words.

In this paper, we apply two ways to map the words representation into its sentence representation: weighted-pooling at Equation 2 and picking `[CLS]` token as sentence (Liu and Lapata, 2019).

Table 7 shows that `[CLS]` is not enough to convey enough informative information of words for both our DifferSum and BERTSUMEXT. Especially, DifferSum is more sensitive to the word features since our differential amplifier may amplify the semantic features effectively.

Table 7: ROUGE Scores about Sentence Representation on CNN/DM Test Set.

Models	R-1	R-L
DifferSum (Weighted-Pooling)	**44.70**	**40.83**
DifferSum (`[CLS]`)	44.41	40.43
BERTSUMEXT (Weighted-Pooling)	43.92	40.08
BERTSUMEXT (`[CLS]`)	43.85	39.90

6 Conclusion

In this paper, we introduce a heuristic model, DifferSum, 1) to calculate and amplifier the pivotal information and 2) to rebalance the distribution of minority 1-class and majority 0-class. Besides, we employ another weighted cross-entropy function to compensate for the imbalance. Experimental results show that our method significantly outperforms previous models. In the future, we would like to generalize DifferSum to other fields.

Acknowledgements

This research is supported by the National Key Research and Development Program of China

(NO.2017YFC0820700) and National Natural Science Foundation of China (No.61902394). We thank all authors for their contributions and all anonymous reviewers for their constructive comments.

References

Kristjan Arumae and Fei Liu. 2018. Reinforced extractive summarization with question-focused rewards. In *ACL*, pages 105–111.

Akari Asai and Hannaneh Hajishirzi. 2020. Logic-guided data augmentation and regularization for consistent question answering. In *ACL*, pages 5642–5650.

Sanghwan Bae, Taeuk Kim, Jihoon Kim, and Sang goo Lee. 2019. Summary level training of sentence rewriting for abstractive summarization. In *arXiv preprint arXiv:1909.08752*.

David Balduzzi, Marcus Frean, Lennox Leary, JP Lewis, Kurt Wan-Duo Ma, and Brian McWilliams. 2018. The shattered gradients problem: If resnets are the answer, then what is the question? In *ICML*, pages 342–350.

Christos Baziotis, Ion Androutsopoulos, Ioannis Konstas, and Alexandros Potamianos. 2019. Seq^3: Differentiable sequence-to-sequence-to-sequence autoencoder for unsupervised abstractive sentence compression. In *NAACL-HLT*, pages 673–681.

Keping Bi, Rahul Jha, W. Bruce Croft, and Asli Celikyilmaz. 2020. Aredsum: Adaptive redundancy-aware iterative sentence ranking for extractive document summarization.

Jaime Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*, pages 209–210.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. In *ACL*, pages 675–686.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *ACL*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *EMNLP*, pages 1724–1734.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186.

Greg Durrett, Taylor Berg-Kirkpatrick, and Dan Klein. 2016. Learning-based single-document summarization with compression and anaphoricity constraints. In *arXiv preprint arXiv:1603.08887*.

Haibo He and Yunqian Ma. 2013. *Imbalanced learning: foundations, algorithms, and applications*. John Wiley & Sons.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016a. Deep residual learning for image recognition. In *CVPR*, pages 770–778.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016b. Identity mappings in deep residual networks. In *ECCV*, pages 630–645.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NIPS*, pages 1693–1701.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, pages 1735–1780.

Yi Yao Huang and William Yang Wang. 2017. Deep residual learning for weakly-supervised relation extraction. In *EMNLP*, pages 1803–1807.

Aishwarya Jadhav and Vaibhav Rajan. 2018. Extractive summarization with swap-net: Sentences and words from alternating pointer networks. In *ACL*, pages 142–151.

Ruipeng Jia, Yanan Cao, Haichao Shi, Fang Fang, Cong Cao, and Shi Wang. 2021. Flexible non-autoregressive extractive summarization with threshold: How to extract a non-fixed number of summary sentences. In *AAAI*.

Ruipeng Jia, Yanan Cao, Haichao Shi, Fang Fang, Yanbing Liu, and Jianlong Tan. 2020a. Distilsum: Distilling the knowledge for extractive summarization. In *CIKM*, pages 2069–2072.

Ruipeng Jia, Yanan Cao, Hengzhu Tang, Fang Fang, Cong Cao, and Shi Wang. 2020b. Neural extractive summarization with hierarchical attentive heterogeneous graph network. In *EMNLP*, pages 3622–3631.

Chris Kedzie, Kathleen McKeown, and Hal Daumé III. 2018. Content selection in deep learning models of summarization. In *EMNLP*, pages 1818–1828.

Logan Lebanoff, Kaiqiang Song, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019. Scoring sentence singletons and pairs for abstractive summarization. *ACL*, pages 2175–2189.

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

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *EMNLP*, pages 3728–3738.

Yang Liu, Ivan Titov, and Mirella Lapata. 2019. Single document summarization as tree induction. In *NAACL-HLT*, pages 1745–1755.

Ling Luo, Xiang Ao, Yan Song, Feiyang Pan, Min Yang, and Qing He. 2019. Reading like HER: Human reading inspired extractive summarization. In *EMNLP*, pages 3033–3043.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *ACL*, pages 55–60.

Afonso Mendes, Shashi Narayan, Sebastião Miranda, Zita Marinho, André FT Martins, and Shay B Cohen. 2019. Jointly extracting and compressing documents with summary state representations. In *NAACL-HLT*, pages 3955–3966.

Junghyun Min, R. Thomas McCoy, Dipanjan Das, Emily Pitler, and Tal Linzen. 2020. Syntactic data augmentation increases robustness to inference heuristics. In *ACL*, pages 2339–2352.

Guido Montúfar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. 2014. On the number of linear regions of deep neural networks. In *NIPS*, pages 2924–2932.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI*, pages 3075–3081.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Ranking sentences for extractive summarization with reinforcement learning. In *NAACL-HLT*, pages 1747–1759.

Shashi Narayan, Joshua Maynez, Jakub Adamek, Daniele Pighin, Blaž Bratanič, and Ryan McDonald. 2020. Stepwise extractive summarization and planning with structured transformers. In *EMNLP*, pages 4143–4159.

Karolina Owczarzak, John M Conroy, Hoa Trang Dang, and Ani Nenkova. 2012. An assessment of the accuracy of automatic evaluation in summarization. In *Proceedings of Workshop on Evaluation Metrics and System Comparison for Automatic Summarization*, pages 1–9.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *NIPS*, pages 8024–8035.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, pages 140:1–140:67.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *EMNLP*, pages 379–389.

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

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *ACL*, pages 1073–1083.

Naftali Tishby, Fernando C Pereira, and William Bialek. 2000. The information bottleneck method. *arXiv preprint physics/0004057*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, pages 5998–6008.

Andreas Veit, Michael Wilber, and Serge Belongie. 2016. Residual networks behave like ensembles of relatively shallow networks. In *NIPS*, pages 550–558.

Danqing Wang, Pengfei Liu, Yining Zheng, Xipeng Qiu, and Xuanjing Huang. 2020. Heterogeneous graph neural networks for extractive document summarization. In *ACL*, pages 6209–6219.

Danqing Wang, Pengfei Liu, Ming Zhong, Jie Fu, Xipeng Qiu, and Xuanjing Huang. 2019. Exploring domain shift in extractive text summarization. In *arXiv preprint arXiv:1908.11664*.

Peter West, Ari Holtzman, Jan Buys, and Yejin Choi. 2019. Bottlesum: Unsupervised and self-supervised sentence summarization using the information bottleneck principle. In *EMNLP*, pages 3750–3759.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V. Le. 2020. Unsupervised data augmentation for consistency training. In *NIPS*.

Jiacheng Xu and Greg Durrett. 2019. Neural extractive text summarization with syntactic compression. *EMNLP*, pages 3290–3301.

Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Discourse-aware neural extractive text summarization. In *ACL*, pages 5021–5031.

Yu Yan, Weizhen Qi, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. Prophetnet: Predicting future n-gram for sequence-to-sequence pre-training. In *arXiv preprint arXiv:2001.04063*, pages 2401–2410.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J Liu. 2019a. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *arXiv preprint arXiv:1912.08777*, pages 11328–11339.

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019b. Hibert: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *ACL*, pages 5059–5069.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive summarization as text matching. In *ACL*, pages 6197–6208.

Ming Zhong, Pengfei Liu, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2019a. Searching for effective neural extractive summarization: What works and what's next. In *ACL*, pages 1049–1058.

Ming Zhong, Danqing Wang, Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2019b. A closer look at data bias in neural extractive summarization models. In *arXiv preprint arXiv:1909.13705*.

Qingyu Zhou, Nan Yang, Furu Wei, Shaohan Huang, Ming Zhou, and Tiejun Zhao. 2018. Neural document summarization by jointly learning to score and select sentences. In *ACL*, pages 654–663.

Barret Zoph, Ekin D. Cubuk, Golnaz Ghiasi, Tsung-Yi Lin, Jonathon Shlens, and Quoc V. Le. 2019. Learning data augmentation strategies for object detection. In *ECCV*, pages 566–583.