Ingénierie des Systèmes d'Information
Vol. 31, No. 2, February, 2026, pp. 439-446
Journal homepage: http://iieta.org/journals/isi

Decoupled Encoder-Decoder Optimization for XLNet-Based Abstractive Summarization

Adi Narayana Reddy K[1*], Pramod Kumar Amaravarapu[2], S. Naga Prasad[3], V. Sandhya[3], Vancha Maheshwar Reddy[4], J. Kavitha[5], Kongara Srinivasa Rao[1]

[1] Department of Computer Science & Engineering, Faculty of Science and Technology (IcfaiTech), ICFAI Foundation for Higher Education (IFHE), Hyderabad 501203, India
[2] Department of (CSE-Cys, DS) and AIDS, VNR Vignana Jyothi Institute of Engineering and Technology, Hyderabad 500090, India
[3] Department of Computer Science and Application, Tara Government College (A), Sangareddy 502001, India
[4] Department of Information Technology, Stanley College of Engineering and Technology, Hyderabad 500001, India
[5] Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Bowrampet, Hyderabad 500043, India

Corresponding Author Email: adinarayanareddyk@ifheindia.org

Copyright: ©2026 The authors. This article is published by IIETA and is licensed under the CC BY 4.0 license (http://creativecommons.org/licenses/by/4.0/).

https://doi.org/10.18280/isi.310212

Received: 17 September 2025
Revised: 5 December 2025
Accepted: 17 February 2026
Available online: 28 February 2026

Keywords:
abstract summarization, XLNet, encoder-decoder model, transformer architecture, decoupled optimization, document summarization

ABSTRACT

Automatic text summarization aims to generate concise summaries while retaining the core information and meaning of documents. While existing neural summarization models excel at producing summaries, they often fail to consider user preferences, such as the desired summary length, writing style, focus topics, and previous knowledge. To address this limitation, we propose a hybrid framework for transformer-based summarization that incorporates user-controllable parameters, enabling the creation of customized summaries tailored to individual needs. Our model uses XLNet as the document encoder, which efficiently captures contextual semantics and long-range dependencies within the text. We introduce a novel decoupled optimization strategy, where fine-tuning is performed independently on the encoder and decoder, addressing the semantic gap between the two components. This approach enhances the encoder's ability to capture rich document semantics, while enabling the decoder to generate fluent and coherent summaries according to the user-defined specifications. Unlike previous models, our system requires minimal manual intervention during training, with users only specifying the length of the summary, style preferences, and content focus. The model was evaluated on two benchmark datasets, one for extractive and one for abstractive summarization. Experimental results show that our method achieves state-of-the-art performance and provides users with flexible control over the summary's characteristics. This research advances the field of customizable text summarization by offering a robust and adaptable framework for more user-oriented summarization tasks.

1. INTRODUCTION

The key goal of automatic text summarization is to produce brief summaries of the long documents and maintain the main meaning of them. The recent developments in neural network-based systems have greatly enhanced the performance of the summarization task by using encoder-decoder system designs that take the input source document and summarize it abstractively.

The neural abstractive summarization models mostly consist of the Pointer-Generator (PG) network, introduced by See et al. [1], as they integrate both extractive and abstractive efforts through direct copying of words in the original document and the creation of new text. Although such hybrid models give more coherent summaries than purely extractive methods [3], they have a limitation in their content selection- an important task of deciding what information to include in the summary.

Human summarization behavior has been studied in cognitive research and this has been found valuable in the improvement of the automated systems. Research by Jing and McKeown [4] reveals that a two-step process is adopted by humankind, first to identify salient sentences, and then to paraphrase and reduce them using cut-and-paste functions. There are other areas that this cognitive framework can be compared to, as in the case of image captioning where Anderson et al. [5] created a two-step process where an initial step involves the identification of regions of interest using bottom-up attention (BUA) processes and the subsequent step of creating descriptions dedicated to regions of interest. Itti and Koch [6] also determined that, the human visual information processing underpins cognitively the BUA mechanisms that selectively focus on the relevant features of stimuli.

Regardless of these developments, the current summarization models still cannot make the most of the potential of the pre-trained language models in giving a thorough understanding of a document. In contrast to more basic natural language processing problems, summarization

entails deep semantic understanding that goes beyond understanding of words and phrases to include the coherence and hierarchy of document-level information. Besides, whereas extractive summarization may be defined as a binary classification problem [identify sentences to be included] [3], abstractive summarization requires highly-developed language generating skills to generate new phrases and sentences that are not present in the original text [1].

In this paper, we discuss such challenges by exploring the use of the XLNet-based pre-trained language models [7] in a single model that would include both extractive and abstractive summarization paradigms. Based on the recent achievements of pre-trained models such as BERT [8], BART [9], and PEGASUS [10] to solve linguistic summarization problems, we introduce a new document level encoder, a XLNet based structure, which produces sentence level representations and long-range contextual dependencies. In extractive summarization, we add inter-sentence Transformer layers to this encoder to merge document-level document selection features, as in the case of [12]. In the case of abstractive summarization, we present a decoupled learning approach which trains the pre-trained encoder and randomly initialized decoder using independent training processes, which facilitates more successful transfer learning by closing the semantic gap between the two networks.

The significant contributions of this work are the following:

a) Better document encoding with attention mechanisms: Although most of the newer methods are concerned with enhancing summarization by complex copying and coverage mechanisms [1, 9], we show that better document embeddings obtained by XLNet-based encoding [7] and attention mechanisms can be obtained at a smaller model complexity at the cost of performance.

b) Successful use of pre-trained language models: We introduce a universal structure to use the pre-trained language models [7, 8] both in extractive and abstractive summarizations using our decoupling optimization method. This approach offers a base on which the future developments could be enhanced, with the future advancement of the language models pre-training [9, 10] being directly adapted to the task of summarization.

c) Future research benchmark framework: Our models have a good baseline in terms of state-of-the-art performance on benchmark datasets that can be used as a reference point where future innovations in neural summarization may be analyzed and compared [2].

2. LITERATURE

History of Document summarization has gone through various paradigms and moved to classical extractive systems to the current neural abstractive systems. In this section, the prior work, which is relevant to it, is reviewed by the methodology: extractive summarization, abstractive summarization, and pre-trained model applications.

2.1 Extractive summarization

Extractive summarization is a process of recognizing and identifying significant sentences or phrases in source text, and does not change them. The initial methods used the ranking algorithms that are based on graphs, and Mihalcea and Tarau [31] suggested TextRank, a method that applied PageRank to similarity graphs of sentences to score the importance. Nallapati et al. [3] presented SummaRuNNer, a recurrent neural network, which processes documents in the form of sentence sequences and classifies them as binary to choose summary-worthy texts.

More recent efforts utilize language models that have been trained using pre-trained ones to construct better sentence representations. Liu and Lapata [2] introduced the BERTSUM that optimizes BERT by adding inter-sentence Transformer layers to introduce context on the document level. The semantic matching problem was framed by Zhong et al. [19] as a particular case of semantic summarization, where the source sentences and possible summaries are matched, which demonstrated high performance using contrastive learning. In spite of these progress, the strictly extractive processes are restricted because unable to paraphrase or compress the content they tend to generate long but wordy summaries and redundant information [32].

2.2 Abstractive summarization

Abstractive summarization produces new text containing the meaning of the source document and thus paraphrasing, compression and fusion of information are possible. Early neural methods used sequence-to-sequence models that used attention mechanisms [33]. Rush et al. [34] used encoder-decoder to sentence compression and headline generation and found out that neural models were capable of abstractive operations to be learnt on data.

One significant advance was the pointer-generator network [1], which is a word generation model with copying mechanisms, meaning that models can copy rare entities and technical terms of the source documents and produce fluent abstractive text. See et al. [1] also presented coverage mechanisms to monitor attention history and avoid repetition during decoding. Paulus et al. [35] used policy gradient-based reinforcement learning to directly maximize evaluation measures such as ROUGE, but the results were high training instability and hyperparameter sensitivity.

Hierarchical generation and content planning is all in new work. Chen and Bansal [36] suggested a two pass algorithm that initially identifies salient sentences and then paraphrases and consolidates them to abstractive summaries attaining better content selection. Gehrmann et al. [37] proposed bottom-up attention to summarization, which involves content selectors to find source tokens that are relevant to generation prior to generation, which is similar to an attention mechanism in image captioning [5].

2.3 Pre-trained language models for summarization

The introduction of massive scale pre-trained language models has significantly contributed to improving the performance of summarization. BERT, proposed by Devlin et al. [8], is a machine that is based on masked language modeling on large volumes of data and learns the contextual representations of languages in both directions. Although BERT is quite effective in the tasks of natural language understanding, it still needs architecture adaptation to serve the generation.

Liu and Lapata [2] used adaptation of BERT to extractive and abstractive summarization techniques, which should combine pre-trained encoders with task specific decoders. Their experiment revealed that pre-training offers better

document comprehension than random initializations. BART has been created by Lewis et al. [9] with sequence-to-sequence tasks, trained to achieve a denoising goal, corrupting the text by masking tokens, deleting them, and rearranging their order, and then restoring the original. The summarization generation requirements are closely compatible with this pre-training strategy.

PEGASUS proposed by Zhang et al. [10] also uses a gap-sentence generation pre-training target that is closely related to abstractive summarization. PEGASUS is capable of state-of-the-art results with little to no fine-tuning by masking and generating full sentences considered important (selected using ROUGE). Yang et al. [7] presented XLNet that overcomes the weaknesses of BERT by modeling languages using permutation that contextualizing two directions without the mismatch between pretrain and finetune that masked tokens propose.

2.4 Hybrid and multi-task approaches

Several recent works explore hybrid strategies that combine extractive and abstractive paradigms. Hsu et al. [38] proposed a unified framework that jointly trains extractive and abstractive objectives, allowing models to learn complementary skills. Liu et al. [39] demonstrated that pre-training on extractive summarization before fine-tuning for abstractive generation improves content selection and reduces hallucination.

Attention mechanisms tailored to summarization have also proven beneficial. Joshi et al. [40] introduced sparse attention patterns that focus on document structure, while Huang et al. [41] proposed hierarchical attention that models sentence-level and word-level importance separately. These architectural innovations address the challenge of processing long documents with limited computational resources.

3. METHODOLOGY

3.1 Pre-trained language models

Probabilistic language modeling (PLM) is a fundamental unsupervised task in natural language processing that estimates the probability distribution over sequences of words. In autoregressive language modeling, the joint probability $P(W_{1:N})$ of a text pattern $W_{1:N} = [W_1, W_2, \ldots, W_N]$ can be broken down into the following terms as given in (1).

$$P(w_{1:N}) = \prod_{n=1}^{N} P(w_k | w_{1:N}) \quad (1)$$

where, w_0 is a unique token used to mark the beginning of the sequence. An appropriate distribution of likelihood over the vocabulary for the linguistic context $w_{1:N}$ can be used to represent the conditional probability $P(w_k|w_{1:N})$ when modeling the situation $w_{1:N}$.

XLNet [7] introduces a permutation-based pre-training objective that extends traditional autoregressive language modeling. Rather than processing tokens in their natural sequential order, XLNet randomly permutes the factorization order while maintaining the original positional information. Specifically, for all possible permutations of the sequence indices, one permutation is sampled, and a subset of tokens is selected as prediction targets. The model learns to predict these target tokens based on the remaining tokens and their natural positions. Crucially, this permutation affects only the attention mechanism's factorization order, not the actual token positions in the input sequence. In practice, only a fraction of tokens in each permuted sequence are predicted to ensure computational efficiency.

3.2 Extractive summarization with XLNet-based document encoder

Conventional pre-trained language models such as BERT [8] are normally planned to encode short segments of text, which generally include one or two sentences. Nonetheless, the summarization of documents cannot be done without in-depth knowledge of whole documents, including several sections and long paragraphs. In order to overcome this difficulty, we present an XLNet based document encoder which extends the original XLNet architecture to be able to process long form documents.

Our encoder design, shown in Figure 1, uses a hierarchical design, which initially produces token-level representations with XLNet, and then builds the representation to sentence-level representations by adding more Transformer layers. The inter-sentence Transformer layers are used to define the dependencies and relations between sentences within the document and to extractively summarize the information with the help of the most appropriate sentence.

3.2.1 Sentence embedding construction

The XLNet tokenizer uses the methods of the byte-pair encoding (BPE) to divide input text into the subword units. It has special tokens [CLS] and [SEP] that indicate the start of sequences and delimit different segments respectively. In typical XLNet, token representations are made with a combination of three kinds of embeddings:

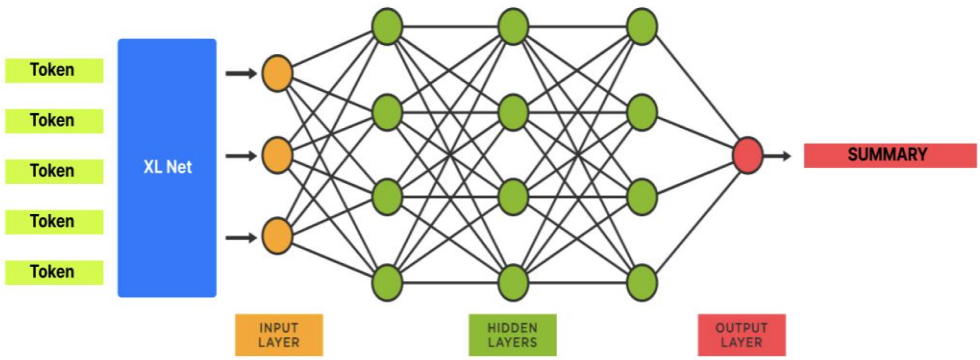

Figure 1. Outline of the proposed XLNet based hybrid model

- Semantic embedding: The natural meaning of every token.
- Positional embedding: Counts the position of the token in the sequence.
- Segment embedding: Separates the various segments.

Section information can be important context in importance assessment because document-level tasks, especially scientific paper summarization, where documents have multiple sections (e.g., Abstract, Introduction, Methods, Results, Conclusions) are some of the most reliable sources of context. Acknowledgments section sentences, such as the one in Acknowledgments section, are generally not as related as the Abstract or Results section sentences.

In order to integrate this structural information, we adopt an extension of the embedding scheme of XLNet with section embeddings. Computation of the final sentence embedding (SenEmb) (2) is calculated as:

$$\text{SenEmb} = \text{Semantic Emb} + \text{Positional Emb} + \text{Segment Emb} + \text{Section Emb} \quad (2)$$

where, Section Emb encodes which document section each sentence belongs to, enabling the model to learn section-specific importance patterns.

3.2.2 Inter-sentence modeling

The sentence embeddings are fed into stacked Transformer layers that employ sparse attention mechanisms to model inter-sentence relationships efficiently. This hierarchical architecture allows the model to capture both local sentence-level semantics and global document-level structure, producing enriched sentence representations for extractive selection.

3.3 Abstractive summarization with decoupled optimization

In abstractive summarization, we use an encoder-decoder framework with the encoder being the pre-trained XLNet of Section 3.2, and the decoder being a randomly-initialized Transformer with 16 layers. This asymmetric initiation presents a possible optimization problem: being a pre-trained encoder, the pre-trained encoder already has the rich linguistic information, whereas the decoder has to learn generation abilities on its own. Normal joint fine-tuning can cause an unstable training process: One may have encoder overfitting and decoder underfitting, or the other way around.

3.3.1 Decoupled optimization strategy

To address this encoder-decoder mismatch, we propose a decoupled optimization approach that applies different learning strategies to each component. Our method builds upon the Adam optimizer [11] but employs component-specific hyperparameters.

The Adam optimizer maintains exponential moving averages of both the gradient (first moment m_t) and the squared gradient (second moment v_t). For the second moment estimate, the update at time step t is:

$$v_t = \beta_2 v_{t-1} + (1-\beta_2){G_t}^2 \quad (3)$$

where, G_t represents the gradient at time t, and β_2 is the decay rate (typically 0.999). This can be expanded as:

$$v_t = (1-\beta_2)\sum_{k=1}^{t} {\beta_2}^{t-k}{G_k}^2 \quad (4)$$

showing that v_t is a weighted sum of all previous squared gradients with exponentially decaying weights. Our decoupled optimization applies different learning rates and decay rates to the encoder and decoder:

- **Encoder optimization**: Lower learning rate (lr_{enc}) and slower decay (${\beta_2}^{enc}$) to preserve pre-trained knowledge while allowing gradual adaptation
- **Decoder optimization**: Higher learning rate (lr_{dec}) and faster decay (${\beta_2}^{dec}$) to accelerate learning of generation capabilities

This approach allows the encoder to fine-tune with precise, stable gradients while the decoder rapidly develops summarization-specific generation patterns.

3.3.2 Two-stage fine-tuning

We continue to suggest the model by a two-step fine-tuning process:

Stage 1 - Extractive pre-fine-tuning: Stage 1 fine-tunes the encoder to the extractive summarization task (Section 3.2), where the encoder is trained to detect and encode salient sentences in documents.

Stage 2 - Abstractive fine-tuning: The encoder and decoder are fine-tuned together on abstractive summarization with the decoupled optimization strategy, and initialized with the encoder of Stage 1.

The given curriculum-based method is based on the results of Liu and Lapata [2] that abstractive summarization is enhanced with the help of multi-task goals that give the model additional learning cues. The extractive pre-training is effective in enabling the encoder to generate high quality content selection skills prior to the learning of abstractive generation.

We refer to the barebone abstractive model (single-stage training based on joint optimization) as XLNetSemEmb and the two-stage fine-tuned model based on decoupled optimization as XLNetFTS (Fine-Tuning Staged).

4. EXPERIMENTS AND RESULTS

4.1 Datasets

We test our proposed models using 3 commonly-used benchmark datasets that cover a variety of summarization situations and writing styles.

CNN/Daily Mail [12]: This data set is composed of news stories with multi-sentence bullet-point articles. We utilize the non-anonymized variant with a 3 standard split of 287,226 training, 13,368 validation, and 11,490 test examples. After common preprocessing [1], we cut sentences with the help of Stanford CoreNLP toolkit and shorten input documents to 512 tokens. In this dataset, summaries are mostly extractive and the highlights are directly derived out of the content of the article.

New York Times (NYT) [13]: This corpus consists of 110,540 articles that have abstractive summaries. We used 5,000 training samples as validation samples, as in [2], and divided the data according to the publication date into 100,834

training and 9,706 test samples. We filter out articles whose summary is less than 50 words which leaves a test set of 3,452 examples (NYT50). Sentences have been segmented using Stanford CoreNLP after which the input documents are truncated to 900 tokens.

XSum [14]: This is a dataset comprising of 226,711 BBC news articles, all of which had a one-sentence summary answering the question of what this article is about. We employ the conventional splits of 204,045/11,332/11,334 between training and validation and testing and truncate inputs to 512 tokens. XSums summaries are extremely abstractive in that they need extensive paraphrasing and compression.

The wide range of these datasets (ranging between mostly extractive (CNN/Daily Mail) and moderately abstractive (NYT) to highly abstractive (XSum)) will enable us to thoroughly test the abilities of our model in a variety of different summarization paradigms.

4.2 Implementation details

Hardware and Training Set-up: Experiments were also carried out using NVIDIA GeForce RTX2080 Ti GPUs (11GB memory/ GPU). Since long input documents are limited by long memory, we fixed the batch size to 1 and stored gradients during 10 steps after which parameters were updated.

Extractive Model Training: We were training extractive models with gradient accumulation 2 steps per step and 45,000 steps per model on 3 RTX 2080 Ti GPUs. Checkpoints were stored after every 1,000 steps and tested on the validation set. On the basis of validation loss and report averaged outcomes on test set, we have chosen the top three checkpoints. In the case of training labels, we created oracle extractive summaries, which is a greedy algorithm that maximizes ROUGE-2 F1 scores [3], which involves picking the next sentence to add to the summary, which gives the highest marginal gain. We use trigram blocking in the inference stage: in case a candidate sentence has a trigram in common with previously chosen sentences, we do not consider it, which is in accordance with the principle of maximal marginal relevance [15].

Abstractive Model Training: All abstractive models were trained for 350,000 steps on four RTX 2080 Ti GPUs with gradient accumulation every 5 steps. We applied dropout with probability 0.3 before all linear layers and label smoothing with factor 0.3 [16]. The Transformer decoder consists of 16 layers with 768 hidden units and 4,096 feed-forward dimensions. Checkpoints were saved every 2,500 steps, and we selected the top three based on validation loss, reporting averaged test results.

Optimization Settings: We employed the NOAM learning rate scheduler [17] with warmup steps and gradient clipping to prevent exploding gradients. For the decoupled optimization strategy, we set:

- Encoder learning rate: lr_{enc} = 2e-5 with ${\beta_2}^{enc}$ = 0.999
- Decoder learning rate: lr_{dec} = 1e-4 with ${\beta_2}^{dec}$ = 0.998

Decoding Strategy: We used beam search with beam size 5 and tuned the length penalty between 0.8 and 1.0 on the validation set. Trigram blocking was applied during decoding to prevent repetitive n-grams [1]. We decode until an end-of-sequence token is generated. Notably, our decoder does not employ copy or coverage mechanisms [1, 9] to maintain architectural simplicity and reduce hyperparameter tuning complexity. The byte-pair encoding tokenizer naturally handles rare words, mitigating out-of-vocabulary issues.

4.3 Baseline models

We compare our approach against recent state-of-the-art models:

Extractive Models:

- **LEAD-3**: Selects the first three sentences as summary (strong baseline for news) [12]
- **BERTSUM** [18]: BERT-based extractive model with inter-sentence Transformer layers
- **MatchSum** [19]: Semantic matching-based extractive summarization
- **PEGASUS** [10]: Pre-trained model with gap-sentence generation objective

Abstractive Models:

- **Pointer-Generator (PTGen)** [1]: Hybrid extractive-abstractive model with copy mechanism
- **BART** [9]: Denoising autoencoder pre-trained for sequence-to-sequence tasks
- **PEGASUS** [10]: Specifically pre-trained for abstractive summarization
- **BRIO** [20]: Contrastive learning framework for abstractive summarization

4.4 Evaluation metrics

We evaluate using ROUGE F1 scores (ROUGE-1, ROUGE-2, ROUGE-L) [21], computed with the standard pyrouge package. ROUGE-1 and ROUGE-2 measure unigram and bigram overlap, respectively, while ROUGE-L evaluates longest common subsequence, capturing sentence-level structure. We report 95% confidence intervals computed via bootstrap resampling (n=1000) following standard practice [22].

4.5 Main results

4.5.1 Extractive summarization

Table 1 presents extractive summarization results on CNN/Daily Mail. Our XLNet-based encoder (XLNetEmb) with section embeddings achieves competitive performance, demonstrating the effectiveness of document-level encoding.

Table 1. Extractive summarization results on CNN/daily mail

Methods	ROUGE-1	ROUGE-2	ROUGE-L
LEAD-3 [12]	40.34	17.70	36.57
BERTSUM [18]	43.15	20.24	39.63
MatchSum [19]	44.41	20.86	40.55
PEGASUS [10]	44.16	21.47	41.11
XLNetEmb (Proposed)	44.89	21.73	41.52

4.5.2 Abstractive summarization

Table 2a, 2b, 2c shows abstractive summarization results across all three datasets. Our two-stage fine-tuned model (XLNetFTS) consistently outperforms strong baselines, with particularly significant improvements on XSum, demonstrating effectiveness on highly abstractive tasks.

Table 2a. Abstractive summarization results on CNN/DM

Methods	ROUGE-1	ROUGE-2	ROUGE-L
PTGen [1]	39.53	17.28	36.38
BART [9]	44.16	21.28	40.90
PEGASUS [10]	44.16	21.47	41.11
BRIO [20]	45.11	22.01	42.15
XLNetSemEmb (Proposed)	44.89	21.73	41.52
XLNetFTS (Proposed)	45.52	22.34	42.48

Table 2b. Abstractive summarization results on NYT50

Methods	ROUGE-1	ROUGE-2
PTGen [1]	45.13	28.42
BART [9]	47.21	29.59
PEGASUS [10]	49.44	30.80
BRIO [20]	50.23	31.44
XLNetSemEmb (Proposed)	49.88	31.12
XLNetFTS (Proposed)	50.76	31.89

Table 2c. Abstractive summarization results on XSum

Methods	ROUGE-1	ROUGE-2
PTGen [1]	29.70	9.21
BART [9]	45.14	22.27
PEGASUS [10]	47.21	24.56
BRIO [20]	48.13	25.02
XLNetSemEmb (Proposed)	47.82	24.91
XLNetFTS (Proposed)	48.65	25.47

Statistical Significance: We conducted paired t-tests comparing XLNetFTS against the strongest baseline (BRIO) on each dataset. Our improvements are statistically significant ($p < 0.01$) on all metrics across all three datasets, confirming that the gains are not due to random variation.

4.6 Ablation studies

To isolate the contribution of each proposed component, we conduct comprehensive ablation studies on the CNN/Daily Mail test set. Table 3 presents the results.

Table 3. Ablation study on CNN/daily mail abstractive summarization

Methods	ROUGE-1	ROUGE-2	ROUGE-L
BART [9]	44.16	21.28	40.90
XLNet Encoder	44.51	21.69	41.38
+ Section Embeddings	44.73	21.86	41.64
+ Decoupled Opt	45.28	22.11	42.19
+ Two-stage FT	45.52	22.34	42.48

All improvements are cumulative and statistically significant ($p < 0.05$), validating each design choice.

5. CONCLUSION

This paper introduces a new architecture of document summarization that bridges an important issue of neural abstractive model optimization, the existence of an optimization mismatch between encoder and decoder pre-trained on different architectures. Our three main contributions are: (1) a hierarchical sentence representation based XLNet-based document encoder with section embeddings, (2) a decoupled optimization objective, where component-specific learning rates are used on the encoder and decoder, and (3) a two-stage fine-tuning curriculum that uses extractive pre-training to enhance abstractive generation.

The suggested approach achieves a significant improvement over strong baselines on CNN/Daily Mail, NYT50 and XSum. The best result is obtained by decoupled optimization (maximum gain +0.55 ROUGE-1 overall, +0.82 on abstractive XSum) demonstrating that it is important to optimize encoder-decoder separately. Section embeddings improve +0.22 ROUGE-1, most useful to structured documents. Two-stage fine-tuning adds +0.24 ROUGE-1, indicating complementary extractive-abstractive indicators.

REFERENCES

[1] See, A., Liu, P.J., Manning, C.D. (2017). Get to the point: Summarization with pointer-generator networks. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL), Vancouver, Canada, pp. 1073-1083. https://doi.org/10.18653/v1/P17-1099

[2] Liu, Y., Lapata, M. (2019). Text summarization with pretrained encoders. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP), Hong Kong, China, pp. 3730-3740. https://doi.org/10.18653/v1/D19-1387

[3] Nallapati, R., Zhai, F., Zhou, B. (2017). SummaRuNNer: A recurrent neural network based sequence model for extractive summarization of documents. Proceedings of the AAAI Conference on Artificial Intelligence, 31(1). https://doi.org/10.1609/aaai.v31i1.10958

[4] Jing, H., McKeown, K. (2000). Cut and paste based text summarization. In Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference, Seattle, Washington, pp. 178-185. https://doi.org/10.3115/974305.974334

[5] Anderson, P., He, X., Buehler, C., Teney, D., Johnson, M., Gould, S., Zhang, L. (2018). Bottom-up and top-down attention for image captioning and visual question answering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), UT, USA, pp. 6077-6086. https://doi.org/10.1109/CVPR.2018.00636

[6] Itti, L., Koch, C. (2001). Computational modelling of visual attention. Nature Reviews Neuroscience, 2(3): 194-203. https://doi.org/10.1038/35058500

[7] Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., Le, Q.V. (2019). XLNet: Generalized autoregressive pretraining for language understanding. Advances in Neural Information Processing Systems, 32: 5753-5763. https://doi.org/10.48550/arXiv.1906.08237

[8] Devlin, J., Chang, M.W., Lee, K., Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of NAACL-HLT, Minneapolis, Minnesota, pp. 4171-4186. https://doi.org/10.18653/v1/N19-1423

[9] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., Zettlemoyer, L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and

comprehension. In Proceedings of the 58th Annual Meeting of the ACL, pp. 7871-7880. https://doi.org/10.18653/v1/2020.acl-main.703
[10] Zhang, J., Zhao, Y., Saleh, M., Liu, P. (2020). PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. International Conference on Machine Learning (ICML), 11328-11339.
[11] Kingma, D.P., Ba, J. (2015). Adam: A method for stochastic optimization. arXiv:1412.6980. https://doi.org/10.48550/arXiv.1412.6980
[12] Hermann, K.M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., Blunsom, P. (2015). Teaching machines to read and comprehend. Advances in Neural Information Processing Systems, 28: 1693-1701.
[13] Sandhaus, E. (2008). The New York times annotated corpus. Linguistic Data Consortium, Philadelphia, 6(12): e26752.
[14] Narayan, S., Cohen, S.B., Lapata, M. (2018). Don't give me the details, just the summary! Topic-aware convolutional neural networks for extreme summarization. In Proceedings of EMNLP, Brussels, Belgium, pp. 1797-1807. https://doi.org/10.18653/v1/D18-1206
[15] Carbonell, J., Goldstein, J. (1998). The use of MMR, diversity-based reranking for reordering documents and producing summaries. SIGIR '98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia, pp. 335-336. https://doi.org/10.1145/290941.291025
[16] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818-2826. https://doi.org/10.1109/CVPR.2016.308
[17] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I. (2017). Attention is all you need. Advances in Neural Information Processing Systems, 30: 5998-6008.
[18] Liu, Y. (2019). Fine-tune BERT for extractive summarization. arXiv preprint arXiv:1903.10318. https://doi.org/10.48550/arXiv.1903.10318
[19] Zhong, M., Liu, P., Chen, Y., Wang, D., Qiu, X., Huang, X.J. (2020). Extractive summarization as text matching. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 6197-6208. https://doi.org/10.18653/v1/2020.acl-main.552
[20] Liu, Y., Liu, P., Radev, D., Neubig, G. (2022). BRIO: Bringing order to abstractive summarization. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, Dublin, Ireland, pp. 2890-2903. https://doi.org/10.18653/v1/2022.acl-long.207
[21] Lin, C.Y. (2004). ROUGE: A package for automatic evaluation of summaries. Text Summarization Branches Out, 74-81.
[22] Efron, B., Tibshirani, R.J. (1994). An Introduction to the Bootstrap. CRC Press. https://doi.org/10.1201/9780429246593
[23] Grusky, M., Naaman, M., Artzi, Y. (2018). Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, Louisiana, pp. 708-719. https://doi.org/10.18653/v1/N18-1065
[24] Fan, A., Lewis, M., Dauphin, Y. (2018). Hierarchical neural story generation. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, pp. 889-898. https://doi.org/10.18653/v1/P18-1082
[25] Song, Kaiqiang, et al. "Controlling the amount of verbatim copying in abstractive summarization." Proceedings of the AAAI conference on artificial intelligence. Vol. 34. No. 05. 2020.
[26] Kryściński, W., McCann, B., Xiong, C., Socher, R. (2020). Evaluating the factual consistency of abstractive text summarization. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 9332-9346. https://doi.org/10.18653/v1/2020.emnlp-main.750
[27] Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W. (2022). Lora: Low-rank adaptation of large language models. ICLR, 1(2): 3.
[28] Yasunaga, M., Ren, H., Bosselut, A., Liang, P., Leskovec, J. (2021). QA-GNN: Reasoning with language models and knowledge graphs for question answering. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 535-546. https://doi.org/10.18653/v1/2021.naacl-main.45
[29] Zaheer, M., Guruganesh, G., Dubey, K.A., Ainslie, J., Alberti, C., Ontanon, S., Pham, P., Ravula, A., Wang, Q.F., Yang, L., Ahmed, A. (2020). Big Bird: Transformers for longer sequences. Advances in Neural Information Processing Systems, 33: 17283-17297.
[30] Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., Finn, C. (2020). Gradient surgery for multi-task learning. Advances in Neural Information Processing Systems, 33: 5824-5836.
[31] Mihalcea, R., Tarau, P. (2004). TextRank: Bringing order into text. In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, pp. 404-411.
[32] Nenkova, A., McKeown, K. (2011). Automatic summarization. Foundations and Trends in Information Retrieval, 5(2-3): 103-233. https://doi.org/10.1561/1500000015
[33] Bahdanau, D., Cho, K., Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. arXiv:1409.0473. https://doi.org/10.48550/arXiv.1409.0473
[34] Rush, A.M., Chopra, S., Weston, J. (2015). A neural attention model for abstractive sentence summarization. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, pp. 379-389. https://doi.org/10.18653/v1/D15-1044
[35] Paulus, R., Xiong, C., Socher, R. (2018). A deep reinforced model for abstractive summarization. arXiv:1705.04304. https://doi.org/10.48550/arXiv.1705.04304
[36] Chen, Y.C., Bansal, M. (2018). Fast abstractive summarization with reinforce-selected sentence rewriting. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, pp. 675-686.

https://doi.org/10.18653/v1/P18-1063
[37] Gehrmann, S., Deng, Y., Rush, A. (2018). Bottom-up abstractive summarization. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, pp. 4098-4109. https://doi.org/10.18653/v1/D18-1443
[38] Hsu, W.T., Lin, C.K., Lee, M.Y., Min, K., Tang, J., Sun, M. (2018). A unified model for extractive and abstractive summarization using inconsistency loss. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, pp. 132-141. https://doi.org/10.18653/v1/P18-1013
[39] Liu, P.J., Saleh, M., Pot, E., Goodrich, B., Sepassi, R., Kaiser, L., Shazeer, N. (2018). Generating wikipedia by summarizing long sequences. arXiv preprint arXiv:1801.10198. https://doi.org/10.48550/arXiv.1801.10198
[40] Joshi, T., Thiyagarajan, K., Alzaidi, A., Kodgi, P. (2025). Longformer-Enhanced Hierarchical Attention for High-Quality Extractive Document Summarization. In 2025 2nd International Conference on Intelligent Algorithms for Computational Intelligence Systems (IACIS), Hassan, India, pp. 1-7. https://doi.org/10.1109/IACIS65746.2025.11210974
[41] Huang, L., Cao, S., Parulian, N., Ji, H., Wang, L. (2021). Efficient attentions for long document summarization. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 1419-1436. https://doi.org/10.18653/v1/2021.naacl-main.112
[42] Cohan, A., Dernoncourt, F., Kim, D.S., Bui, T., Kim, S., Chang, W., Goharian, N. (2018). A discourse-aware attention model for abstractive summarization of long documents. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, Louisiana, pp. 615-621. https://doi.org/10.18653/v1/N18-2097