Complementary Representation of ALBERT for Text Summarization

Wenying Guo, Bin Wu, Bai Wang, Lianwei Li, Junwei Sun, Maham Nazir
Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia
Beijing University of Posts and Telecommunications, Beijing, China
{gwy, wubin, wangbai, llw, junweisun, maham}@bupt.edu.cn

***Abstract*—Pretraining has proved to be an effective strategy to learn the parameters of the deep neural network. It captures the world knowledge that can be adapted to downstream tasks. Text summarization based on ALBERT [1] outperformed previous work by a large margin. However, they only use the final layer as a contextualized representation of the input text. Multiple studies have proven that intermediate layers also encode the rich hierarchy of linguistic information. In this paper, we propose a Fast Complementary Representation Network (FCRN), which dynamically incorporates linguistic knowledge spread across the entire ALBERT for extractive selection. Different from previous work, we measure the importance of hidden layers by all sentence representations rather than all token embeddings, which can filter nonsignificant words and takes six times less time during training. FCRN first obtains the importance of each layer by sentence embeddings and then automatically absorbs the supplementary information to ALBERT's output. We conduct experiments on CNN/DailyMail and XSum datasets. The results show that our model obtains higher ROUGE scores.**

***Index Terms*—Text summarization, Pretrained Language Model, Fast Complementary Representation network**

I. Introduction

Text summarization is the task of condensing the essence of a document to converse the main information contents and main ideas quickly. There are two paradigms to generate a summary: extractive summarization and abstractive summarization. Extractive summarization generates the summary by scoring and extracting the high-score sentences (or more fine-grained units) from source documents generally, such as [2], [3]. In contrast, abstractive summarization is an ideal form of summarization since it generates the novel sentence required to have recourse to world knowledge. Abstractive summarization generates a summary in a human-written way.

Extractive summarization remains more reliable than abstractive summarization since the summary is composed of the text span (like sentence) in the source document. In this paper, we primarily focus on extractive summarization. The pretrained language model has recently brought Natural Language Processing (NLP) to a new era since the emergence of BERT [4].

Numerous summarization approaches have achieved state-of-the-art results based on the pretrained language models. BertSumExt [5] built a BERT-based minimum-requirements model, which outperformed previous work by a large margin.

DOI reference number: 10.18293/SEKE2021-084

DiscoBert [6] extracted sub-sentential discourse units based on BERT and Graph Convolutional Network. MatchSum [7] adopted a Siamese-BERT architecture to select the candidate summary, which was the most similar to the target summary. TSEM [8] showcased the effectiveness of ALBERT to handle text summarization. However, the improvement on automatic metrics like ROUGE has reached a bottleneck. They only use the final layer of the pretrained language model as contextualized representations of the input text. The popularity of transformer-based models has driven researchers to study what was behind their success.

Multiple studies [9]–[11] have proven that transformer-based models composed a hierarchy of linguistic signals ranging from surface to semantic features. Intermediate layers also encode the rich hierarchy of linguistic information. Taking the output of the pretrained language model restricts the power of pretrained representation when fine-tuning [9], [12], [13]. ALBERT has fewer parameters and focuses on modeling inter-sentence coherence. Meanwhile, TSEM has achieved better performance based on ALBERT than BertSumExt based on BERT in the summarization task. In this paper, we propose an architecture stacked on ALBERT, namely Fast Complementary Representation Network (FCRN), which can leverage the rich linguistic knowledge to enhance the power of ALBERT. There is no research to study the effectiveness of the fused representation of the ALBERT on text summarization. In our model, The ALBERT's embeddings of hidden layers are complementary to the output of ALBERT. To capture the information hierarchically, different from previous work [12], [13], We first obtain the sentence embedding in the document to capture the gist of the individual sentence. The number of sentences is far more than tokens, which can reduce training time. The embeddings of sentences that contain the main information are input to FCRN to catch a complementary representation of the final output. We highlight our contributions as follows:

- We proposed FCRN to capture more language information of sentence. We also study the performance of different language knowledge on text summarization. To our knowledge, we are the first ones to dynamically fuse the rich information spread across the entire model on ALBERT.
- In our model, We propose taking sentence embeddings rather than all token embeddings as the input of FCRN.

It shortens the training and inference time with better performance. Our model can generate a better ground-truth to train the MATCHSUM.

- We identify the importance of linear word order, syntactic rule, and semantic information for the summarization task.

II. RELATED WORK

A. Pretrained Language Model

The state-of-the-art result has been achieved by pretrained language models in many NLP tasks. Pretrained language models have advanced downstream NLP tasks by learning universal language representation. In earlier research, the pretrained language model aims to learn good word embeddings containing semantic meaning. Those pretrained representations are added to downstream tasks as additional features. For BERT, ALBERT, and RoBERTa [14], they obtain the embedding of the token with context. Unlike the earlier pretrained language models, they can capture syntactic structures, semantic rules, and context-dependent natures of words. The representation of a given token in vocabulary depends on the whole text. Those models are stacked with a deep network, fine-tuned on downstream tasks with a better model initialization.

Currently, the transformer-based pretrained language model has drawn more attention. The best known transformer-based model is BERT. The curiosity about transformer-based models has driven over 150 studies of the popular BERT model. Based on this knowledge, many enhanced versions of BERT are proposed. The backbone of their architecture is similar to BERT. RoBERTa improves BERT by dynamic masking. ALBERT replaces the next sentence prediction with sentence order prediction to model inter-sentence coherence. All of them can be fine-tuned to adapt to special downstream tasks.

B. Abstractive Summarization

Neural abstractive summarization models conceptualize the task as a sequence-to-sequence problem, where an input sequence is mapped into another output sequence. In 2015, Rush et al. [15] applied the neural encoder-decoder architecture to abstractive summarization and thus paved the way for using neural networks for abstractive summarization. Nallapati et al. [16] and See et al. [17] applied a pointer generator network which generates words from a fixed vocabulary or copies from the source document, which is an effective method to handle Out of Vocabulary (OOV). See et al. presented a coverage mechanism to discourage repetition. To solve the mismatch between the learning object and the evaluation criterion, reinforcement learning-based models trained by optimizing the ROUGE metric achieved higher performance. Encoder-decoder transformers have shown great successes for abstractive summarization.

C. Extractive Summarization

Extractive summarization has gained more attention with its simplicity and facticity. It is often defined as a binary classification. The label of the text span indicates whether it should be included in the summary. Initially, statistical methods generate summaries leveraging the similarity between sentences. They consider statistical features, including sentence position, term frequency (TF), and the inverse document frequency (IDF).

Neural networks are primarily introduced to text summarization by modeling the semantic meaning of sentences [18]. Recently, deep neural networks have achieved great success in summarization tasks. Neural text summarization generally obtains the sentence representation by a neural encoder. Nallapati et al. [19] instantiated the encoder by recurrent neural network (RNN). Zhong et al. [20] leveraged transformer to encode the semantic meaning by interacting between sentences.

Graph Neural Networks (GNN) can learn from complex structured data. Xu et al. [6] presented how GNN can be usefully applied in text summarization. NEUSUM [21] was a neural extractive document framework that jointly learnt to score and select sentences. The reinforcement learning-based model trained by directly optimizing the ROUGE metric achieved state-of-the-art results [22], [23]. Pretrained language models have achieved state-of-the-art results on extractive summarization. HIBERT [3], BERTSUMEXT, MATCHSUM are based on BERT, and TSEM [8] explored the potential of ALBERT on text summarization. It showed ALBERT performed better than BERT with fewer parameters. The pretrained language model was fine-tuned to adapt to text summarization in the above work. They only use the final layer as a contextualized representation of the input text.

Compared to the models outlined above, we explore the potential of fusing the representation of multiple layers as complementary to the last layer. It makes sentence embeddings capture more language knowledge. Moreover, sentence embeddings are used to weight all hidden embeddings which can take less time and capture the sense of sentence.

III. MODEL

A. Overview

Fig. 1 presents an overview of our model, which consists of a Sentence Encoder and FCRN. For the Sentence Encoder, a pretrained ALBERT is used to output the representation of each sentence in all hidden layers. FCRN takes the output of the Sentence Encoder as input and dynamically summarizes the hidden representation based on the output of Bi-LSTM, which enhances the power of the final output. The outputs of FCRN are used to predict the label of the sentence.

Let D = [$sent_1$, $sent_2$,..., $sent_n$] denote a document with n sentences, where $sent_i$ is the i-th sentence. We formulate extractive summarization as a sequence labeling task, in which the label sequence Y= [y_1, y_2,..., y_n] $\in \{0, 1\}$ indicates whether the corresponding sentence is included in the summary or not. We will calculate the final predicted score $\hat{y}_i$. The loss of our model is the binary classifier entropy between the prediction and the gold label.

$$L = -\sum_{i=1}^{n}(y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)) \quad (1)$$

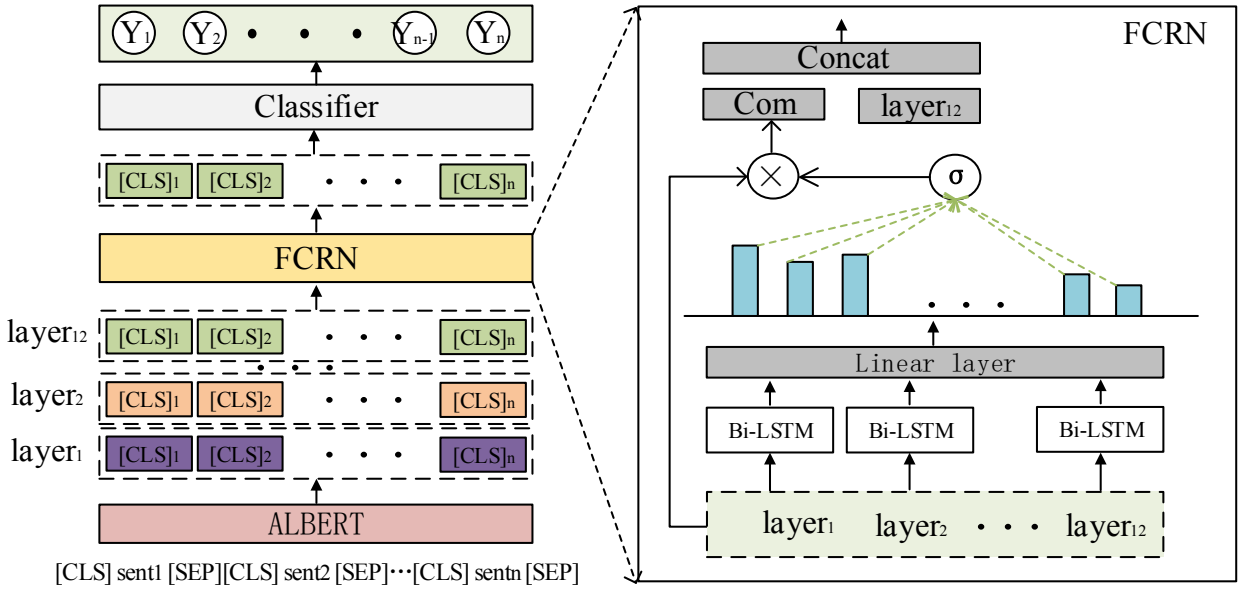

Fig. 1. (left) is an overview of our model, FCRN is stacked on ALBERT encoder. (right) is the architecture of FCRN. σ is a softmax function to calculate the importance of each layer.

B. Sentence Encoder

ALBERT is a transformer-based encoder, which can generate the contextual representation of words based on tokens, segment tokens, and position ids. ALBERT is pretrained on a sentence or sentence pair, and a special classification token ([CLS]) is used as the aggregate representation for the classification task. Given Document D is a document with multiple sentences needing to be classified, we insert [CLS], and [SEP] at the beginning and the ending of each sentence. [CLS] is to capture the meaning of an individual sentence and [SEP] is viewed as the boundary of the sentence. Segment token $\{0, 1\}$ indicates the position of sentences is odd or even in the document. For document D, the input tokens are [CLS], $sent_1$, [SEP], [CLS], $sent_2$, [SEP], ..., [CLS], $sent_n$, [SEP]], and the segment tokens are [0, 1, 0, 1, ...]. We input D as the input to ALBERT like in Fig. 1. All hidden-state embeddings are computed by ALBERT, The formula is as follows:

$$T_1, ...T_l = \text{ALBERT}([\text{CLS}], sent_1, [\text{SEP}], ..., [\text{CLS}], sent_n, [\text{SEP}]) \tag{2}$$

$$s_1, ...s_l = \text{Extractor}(T_1, ..., T_l) \tag{3}$$

where T_i is the output of the i-th layer, and l denotes the number of layers in ALBERT. However, T_i contains the embedding of each token in the document. We select the embeddings of [CLS] as the sentence embeddings by the Extractor in each layer. The sentence embeddings of i-th layer $s_i \in R^{n \times d}$, where n is the number of sentences and d is the hidden size of the encoder. They will be sent to FCRN to generate the final sentence representation.

C. FCRN

FCRN is proposed to generate a more powerful representation. We can select a different number of input layers for FCRN. To capture the long-range dependency of sentences, a single-layer bidirectional Long Short Time Memory (Bi-LSTM) are used to weight all hidden sentence embedding. We apply Layer Normalization (LayerNorm) to Bi-LSTM to stabilize the training. The formulation of Bi-LSTM are as follows:

$$\begin{pmatrix} F_i \\ I_i \\ O_i \\ G_i \end{pmatrix} = LN_h(W_h H_{i-1}) + LN_x(W_x x_i) \tag{4}$$

$$C_i = \sigma(F_i) \odot C_{i-1} + \sigma(I_i) \odot \tanh(G_{i-1}) \tag{5}$$

$$H_i = \sigma(O_i) \odot \tanh(LN_c(C_i)) \tag{6}$$

Where H_i is the hidden state at time i, C_i is the cell state at the time i, and I_i, F_i, G_i, O_i are the input, forget, cell, and output gates, LN_h, LN_x, LN_c are different layer normalization operations. The bias is ignored in the formulation. x_i is the input in step i.

The input to the FCRN is $s_1, s_2, ..., s_l$. We take s_i to Bi-LSTM. Bi-LSTM will obtain a fixed-sized embedding K_i to capture the knowledge in i-th layer. We take the concatenation

TABLE I
THE RESULTS OF OUR PROPOSED MODELS ON CNN/DAILYMAIL DATASET. MODELS WITH SUBSCRIPT * WERE TRAINED AND TESTED ON THE ANONYMIZED-VERSION DATASET. FCRN$_{(w/o)*}$ ($_{(w)*}$) MEANS THE MODEL WITHOUT (WITH) * LAYERS.

Model	R1	R2	RL
ORACLE	51.64	30.48	47.88
LEAD-3	40.37	17.44	36.61
Rnn-ext+RL*	40.55	18.42	36.84
NEUSUM	41.86	19.16	38.20
BERTSUMEXT	43.32	20.32	39.71
FCRN$_{Com}$	43.43	20.36	39.78
TSEM-sentence	43.54	20.47	39.93
FCRN	**43.66**	**20.54**	**40.03**
MATCH-ORACLE	51.08	26.94	46.99
MATCH-ORACLE-FCRN	**51.29**	**27.15**	**47.39**
MATCHSUM-BERT	44.22	20.62	40.38
MATCHSUM-FCRN	**44.36**	**20.70**	**40.51**
FCRN$_{(w/o)high}$	43.48	20.36	39.85
FCRN$_{(w/o)middle}$	43.34	20.27	39.71
FCRN$_{(w/o)low}$	43.61	20.51	40.01
FCRN$_{(w)high}$	43.28	20.22	39.65
FCRN$_{(w)middle}$	43.64	20.53	40.04
FCRN$_{(w)low}$	43.05	19.98	39.40

of each direction's final state as a knowledge of the current layer.

$$K_i = ReLU\left(LayerNormLSTM\left(s_i\right)\right) \tag{7}$$

The importance of each layer will be calculated as:

$$\alpha_i = softmax\left(K_i\right) \tag{8}$$

Given the importance of each layer, we calculate the complementary representation Com to the final output of the encoder, and we concatenate them as the representation of all sentences which captures more language information.

$$Com = \sum_{i=1}^{n} \alpha_i s_i \tag{9}$$

$$S = [s_l \mid Com] \tag{10}$$

A classifier is stacked on FCRN to calculate the label $\hat{y}_i$. The classifier is instantiated by a linear layer and a sigmoid function.

$$\hat{y}_i = \sigma(W_e S_i + b_i) \tag{11}$$

IV. EXPERIMENTS

In this section, we present our experiments from text summarization datasets, evaluative criteria, the implementation details of our model, comparison with multiple previous approaches, hierarchy linguistic knowledge analysis and the compute efficiency studies.

A. Datasets

We evaluate our model on two benchmark datasets, namely CNN/DailyMail and XSum. These datasets possess diverse summary characteristics. The proportion of novel n-grams represents the level of abstraction of the dataset. The highly abstractive dataset can reflect the potential of the model to capture the semantic information of text span. We used a greedy algorithm similar to [19] to obtain an oracle summary for each document to train extractive models.

TABLE II
THE RESULTS OF OUR PROPOSED MODELS ON XSUM DATASET.

Model	R1	R2	RL
ORACLE	28.96	8.19	21.86
LEAD	19.66	2.39	14.80
BERTSUMEXT	23.53	4.54	17.80
TSEM-sentence	23.91	4.69	18.11
FCRN	**24.14**	**4.74**	**18.27**
MATCH-ORACLE	29.54	7.40	22.55
MATCH-ORACLE-FCRN	**30.10**	**7.38**	**23.08**

1) CNN/DailyMail: CNN/DailyMail dataset is generated by modifying a question answering dataset, including 93K articles from CNN and 200k articles from Daily mail websites. Following previous work, we conduct experiments on the non-anonymized version and split the dataset into 287,226/13,368/11,490 for training, validation, and testing. Due to the limitations of the memory and the model, the document is truncated to 512 tokens.

2) XSum: XSum dataset consists of 226,711 BBC articles with single-sentence summaries. It has more novel n-grams in the target summaries that do not appear in their source document than CNN/DailyMail dataset. Following Narayan et al. [24], we split XSum into 204,045/11,332/11,334 for training, validation, and testing.

B. ROUGE

ROUGE is a package for the automatic evaluation of system-generated summaries by counting the number of overlapping units between the generated summary and the reference summary. There are three ROUGE metrics extensively used in text summarization, namely the F1 score of ROUGE-1, ROUGE-2, and ROUGE-L. (R1 and R2 are shorthands for ROUGE-1, ROUGE-2; RL is ROUGE-L). R1 and R2 evaluate the summary by counting the overlapping uni-grams and bi-grams, respectively. RL assesses by longest common subsequence. Following previous work, we report R1, R2, RL as a means of assessing fluency.

C. Experimental Setup

1) Implementation details: We implemented our model based on the "albert-based-v2" version of ALBERT[1]. We set the hidden size of LSTM to 384 due to the Layer Normalization. Following Guo et al. [8], the Adam optimizer with a learning rate of $2e^{-3}, \beta_1 = 0.9$, $\beta_2 = 0.999$ is used during training. The warmup strategy increases the learning rate from 0 to $2e^{-3}$ on the first 15000 steps:

$$lr = 2e^{-3} \cdot \left(step^{-0.5}, step \cdot warmup^{-1.5}\right) \tag{12}$$

We conduct our experiments on 3 GPUs (RTX 2080 Ti) with 50000 steps. The gradient accumulation with two steps is used to enlarge the batch size. Model checkpoints are evaluated on

[1]https://github.com/huggingface/transformers/tree/master/src/transformers/models/albert

TABLE III
THE RESULTS OF COMPUTE EFFICIENCY STUDIES ON CNN/DAILYMAIL, INCLUDING SPEED COMPARISON WITH FCRN-TOKEN AS THE BASELINE

Model	R1	R2	RL	Speedup$_{train}$	Speedup$_{inf}$
FCRN-token	43.58	20.48	39.97	1.0	1.0
FCRN	43.66	20.54	40.03	5.9x	1.65x

the validation set per 1000 steps. We saved the three best checkpoints on evaluated losses and record the performance of the best checkpoints on the test set.

We obtained the scores of the candidate sentences and sorted the sentences in descending order based on these scores. Top-3 sentences are selected to form the summary.

2) Trigram Blocking: This is a simple but effective approach to improve performance during inference. Given a selected summary set and a sentence, the sentence will be selected into the summary set when it has no trigram overlapping with the selected summary. We also apply this approach to this work.

D. Experimental Results and Analysis

A comprehensive experiment is conducted on CNN/DailyMail and XSum. We use the official ROUGE script (version 1.5.5) to evaluate the generated summaries. The results are reported as follows.

1) CNN/DailyMail: Table I summarizes our results on the CNN/DailyMail dataset. We list strong baselines with different learning approaches. **LEAD-3** is a commonly used and strong extractive baseline. It simply selects the first three sentences as a summary. **Rnn-ext+RL** [22] applies policy-based reinforcement learning to the extractor. **NEUSUM** [21] extracts the summary by jointly learning to score and select a sentence. **BERTSUMEXT** [5] extends BERT to text summarization by inserting multiple [CLS], segment information, and stacking a linear layer. **TSEM-sentence** [8] is the first stage of TSEM to assign relevance scores to sentences. We change it as a baseline by choosing three top-ranked sentences to compose a summary. **FCRN**$_{Com}$ predicts the score of a sentence only by complementary representation Com. **MATCH-ORACLE** [7] extracted by BERTSUMEXT is the ground-truth used to train MATCHSUM. This model has achieved the state-of-the-art extractive result on CNN/DailyMail. **MATCH-ORACLE-FCRN** extracts the ground-truth by our model. **MATCHSUM-FCRN** is trained on MATCH-ORACLE-FCRN, while **MATCHSUM-BERT** is trained on MATCH-ORACLE.

From the first block of Table I, we observed that our approach outperformed the baseline models. As compared to TSEM-sentence and FCRN$_{Com}$, which only used the final output of ALBERT and the complementary embedding, respectively, our model improved the ROUGE score. It means that the knowledge of the hidden layers as complementary to the final layer indeed achieves better performance. TSEM-sentence performs than FCRN$_{Com}$ means semantic information is more critical for text summarization. The fusion representation of hidden layers can weaken semantic information of a sentence. FCRN provides syntactic and surface information as complementary to semantic features. This further confirms that text summarization requires comprehensive linguistic knowledge.

The second block in Table I presents that the MATCHSUM ground-truth extracted by our model provides a stronger oracle than BERTSUMEXT. The MATCHSUM approach performs better with the MATCH-ORACLE-FCRN.

2) XSum: Our main results on the XSum dataset are shown in Table II. Again, we report the performance of **ORACLE, LEAD, MATCH-ORACLE-FCRN, MATCH-ORACLE, and BERTSUMEXT** where LEAD baseline simply selects the first two sentences from the document. XSum has a lower ROUGE score than CNN/DailyMail because its summary is more abstractive. Following previous work [5], [8], we process the XSum dataset with a greedy algorithm and then conduct experiments on TSEM-sentence. The experiment results demonstrated the effectiveness of ALBERT on the XSum dataset. We can observe that our model with complementary information to the final layer again is superior to all baselines. MATCH-ORACLE-FCRN generates a better ground-truth to train the MATCHSUM.

3) Hierarchy Linguistic Knowledge Analysis: Previous work has proven that surface features are most prominent in lower layers, syntactic features in middle layers, and semantic features in higher layers. In this work, we split all hidden layers of ALBERT into low layers (1-4), middle layers (5-8), high layers (9-12). We take them as the input of FCRN, respectively. It can change the linguistic knowledge complementary to the output of ALBERT. A comprehensive experiment is conducted to detect the effect of different layers on CNN/DailyMail dataset. The results are reported in the third block of Table I.

We observe that the model without middle layers is the worst, and high layers are more effective than low layers. It means that complementary syntactic knowledge improves the overall system performance better than semantic knowledge on CNN/Daily Mail dataset. The reason is due to the similarity between the gold and sentences in the generated summary. Meanwhile, the final layer of ALBERT has included semantic information. It makes no difference whether or not the lower layers are added since the lower layers possess the most information about linear words. We also observe the knowledge of all hidden layers as the input to FCRN can achieve better performance.

E. Compute Efficiency

We conduct experiments on CNN/DailyMail dataset to verify the compute efficiency. We trained models with all token embeddings and sentence embeddings, namely FCRN-token and FCRN, respectively. We report the running time, inference time, and the performance of models in Table III. FCRN is about 6 times faster than token-level in iterating through the data during training and about 1.65 times during inference. Compared to FCRN-token, FCRN will require significantly less time to train since a sequence of inputs passes through LSTM cell, one at a time. The performance of FCRN-token does not improve with fusion based on token embeddings but,

using sentence embeddings achieves better performance. The above phenomena suggests that using sentence embedding is more compute-efficient and capture the more comprehensive knowledge of each layer. The model performance poorly removing the complementary information, demonstrating the complementary representation is crucial for predicting the summary of a document.

V. Conclusion

In this paper, we propose FCRN to showcase how to fuse all linguistic knowledge in hidden layers and enhance the power of ALBERT. Experiments demonstrate constant improvement over baselines on two benchmark datasets. Furthermore, Using sentence embeddings not only achieved better results but also decreased training and inference time. We find that the syntactic information is worthy of note in our model. We plan to consider more syntactic information and import more world knowledge as a part of our future work.

VI. ACKNOWLEDGMENT

This work is supported by the National Key Research and Development Program of China (2018YFC0831500), the National Natural Science Foundation of China under Grant No.61972047 and the NSFC-General Technology Basic Research Joint Funds under Grant U1936220.

References

[1] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," in *International Conference on Learning Representations*, 2020.

[2] D. Wang, P. Liu, Y. Zheng, X. Qiu, and X. Huang, "Heterogeneous graph neural networks for extractive document summarization," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, Jul. 2020, pp. 6209–6219.

[3] X. Zhang, F. Wei, and M. Zhou, "Hibert: Document level pre-training of hierarchical bidirectional transformers for document summarization," *arXiv preprint arXiv:1905.06566*, 2019.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186.

[5] Y. Liu and M. Lapata, "Text summarization with pretrained encoders," 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: Association for Computational Linguistics, Nov. 2019, pp. 3730–3740.

[6] J. Xu, Z. Gan, Y. Cheng, and J. Liu, "Discourse-aware neural extractive text summarization," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, Jul. 2020, pp. 5021–5031.

[7] M. Zhong, P. Liu, Y. Chen, D. Wang, X. Qiu, and X. Huang, "Extractive summarization as text matching," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, Jul. 2020, pp. 6197–6208.

[8] W. Guo, B. Wu, B. Wang, and Y. Yang, "Two-stage encoding extractive summarization," in *2020 IEEE Fifth International Conference on Data Science in Cyberspace (DSC)*, 2020, pp. 346–350.

[9] G. Jawahar, B. Sagot, and D. Seddah, "What does BERT learn about the structure of language?" in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 3651–3657.

[10] I. Tenney, D. Das, and E. Pavlick, "BERT rediscovers the classical NLP pipeline," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 4593–4601.

[11] A. Rogers, O. Kovaleva, and A. Rumshisky, "A primer in bertology: What we know about how BERT works," *Trans. Assoc. Comput. Linguistics*, vol. 8, pp. 842–866, 2020.

[12] Y. Goldberg, "Assessing bert's syntactic abilities," *arXiv preprint arXiv:1901.05287*, 2019.

[13] C. Zhu, M. Zeng, and X. Huang, "Sdnet: Contextualized attention-based deep network for conversational question answering," *arXiv preprint arXiv:1812.03593*, 2018.

[14] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[15] A. M. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, Sep. 2015, pp. 379–389.

[16] R. Nallapati, F. Zhai, and B. Zhou, "Summarunner: A recurrent neural network based sequence model for extractive summarization of documents," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, S. P. Singh and S. Markovitch, Eds. AAAI Press, 2017, pp. 3075–3081.

[17] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 1073–1083.

[18] M. Kågebäck, O. Mogren, N. Tahmasebi, and D. Dubhashi, "Extractive summarization using continuous vector space models," in *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)*. Gothenburg, Sweden: Association for Computational Linguistics, Apr. 2014, pp. 31–39.

[19] R. Nallapati, F. Zhai, and B. Zhou, "Summarunner: a recurrent neural network based sequence model for extractive summarization of documents," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, 2017, pp. 3075–3081.

[20] M. Zhong, D. Wang, P. Liu, X. Qiu, and X. Huang, "A closer look at data bias in neural extractive summarization models," *arXiv preprint arXiv:1909.13705*, 2019.

[21] S. Narayan, S. B. Cohen, and M. Lapata, "Ranking sentences for extractive summarization with reinforcement learning," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 1747–1759.

[22] Y.-C. Chen and M. Bansal, "Fast abstractive summarization with reinforce-selected sentence rewriting," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 675–686.

[23] Q. Zhou, N. Yang, F. Wei, S. Huang, M. Zhou, and T. Zhao, "Neural document summarization by jointly learning to score and select sentences," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 654–663.

[24] S. Narayan, S. B. Cohen, and M. Lapata, "Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 1797–1807.