# SpanBERT-based Multilayer Fusion Model for Extractive Reading Comprehension

Pu Zhang, Lei He, Deng Xi
School of Computer Science and Technology
Chongqing University of Posts and Telecommunications, Chongqing, P. R. China

*Abstract*—**Extractive reading comprehension is a prominent research topic in machine reading comprehension, which aims to predict the correct answer from the given context. Pre-trained models have recently shown considerable effectiveness in this area. However, during the training process, most existing models face the problem of semantic information loss. To address this problem, this paper proposes a model based on the SpanBERT pre-trained model to predict answers using a multi-layer fusion method. Both the outputs of the intermediate layer and the prediction layer of the transformer are fused to perform answer prediction, thereby improving the model's performance. The proposed model achieves F1 scores of 92.54%, 84.02%, 80.86%, 71.32%, and EM scores of 86.27%, 81.25%, 69.10%, 56.42% on the SQuAD1.1, SQuAD2.0, Natural Questions and NewsQA datasets, respectively. Experimental results show that our model outperforms a number of existing models and has excellent performance.**

***Keywords—Machine reading comprehension; pre-trained model; transformer***

## I. Introduction

With the advent of the big data era, getting the right answers from massive amounts of data in a timely and accurate manner has become an urgent task. As one of the most popular natural language processing (NLP) tasks in recent years, machine reading comprehension (MRC) aims to enable machines to learn how to read and understand texts, that is, to find answers to given questions from relevant articles. Extractive reading comprehension, a subtask of MRC, has also made significant progress in recent years, requiring models to extract a continuous passage of text from the given input text as the final answer.

Extractive reading comprehension can overcome the limitations of relying solely on individual words or entities to answer questions. Its task is to use a model to extract an answer from a given passage or paragraph based on a given question. As shown in Fig. 1, given the question "when does season 2 of lethal weapon come out" and the passage "Lethal Weapon is an American buddy cop action comedy ...", the model reads and understands the passage and question, and then extracts a continuous segment "September 26, 2017" from the passage as the answer. The commonly used datasets for extractive reading comprehension include the SQuAD dataset [1], the NewsQA dataset [2], the TriviaQA dataset [3], and so on.

In deep learning-based reading comprehension models, the prediction of answer boundaries relies heavily on information interactions, and scholars have proposed one-way attention models to employ attention mechanisms to enhance the interaction of information between passage and question. For example, Hermann et al. [4] used the one-way attention model as the basis for contextualizing questions, calculating the weight of each word in a passage, and generating a final representation of the passage. However, because one-way attention mechanisms only account for unidirectional attention, resulting in limited interaction between passage and question, researchers later proposed bidirectional attention models. For example, Seo et al. [5] proposed the bidirectional attention model BiDAF, which computes attention between question and passage separately in both directions to enhance information interaction and achieve good results.

In 2018, Google introduced the bidirectional pre-training language model BERT [6]. Since its release, BERT has achieved outstanding results in the field of NLP. Its remarkable performance is attributed to its internal multi-layer transformer structure, and directly using BERT with a simple answer predictor can achieve good performance on the SQuAD dataset [6].

The study by Ganesh et al. [7] showed that different layers of the transformer encoder focus on different semantic information. Ramnath S et al. [8] experimentally demonstrated that the prediction layer of BERT focuses more on contextual understanding and answer prediction, but ignores the interaction between context and question, while the earlier layers of the transformer focus more on the latter. Thus, some semantic information is lost during the learning process.

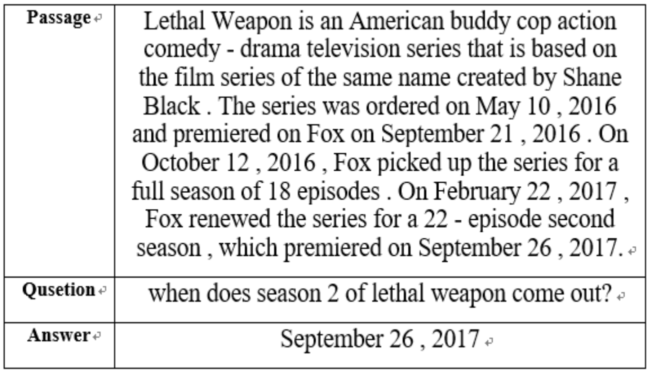
Since BERT was proposed, a number of pre-training models have been successively proposed. Among them, SpanBERT [9] is a representative model. Compared to BERT, SpanBERT [9] is more suitable for extractive tasks. However, it does not take into account the representational information emphasized by intermediate transformer layers, which may affect the final answer extraction in subsequent iterations.

This paper aims to address the potential problem of semantic information loss during learning in SpanBERT by investigating the fusion of semantic information from the intermediate and prediction layers of the transformer. A method is proposed to predict answers using a multi-layer transformer based on the SpanBERT model. The lower layer and the prediction layer of the transformer work together to predict answers. The predicted answer span information from both layers is combined to improve the accuracy of the predicted answers. The contributions of this work are outlined as follows:

*1)* We propose a model that can address the problem of semantic information loss during the learning process of the pre-trained model SpanBERT. Our model enhances the interaction between the paragraph text and the question by utilizing the SpanBERT model, the representation information emphasized by the intermediate layer and the final prediction layer of the transformer can be fused to improve the performance of the model's answer extraction.

*2)* A new approach to vector fusion is proposed in this study, which can effectively combine semantic information. An attention mechanism is utilized to fuse the outputs of the intermediate and prediction layers of the transformer, resulting in a new fused vector. This vector can be used to generate a probability distribution vector of the answer span, which is then multiplied by the answer prediction vector to obtain the predicted answer span. Combining the representational information that the final prediction layer and the middle layer focused on will improve the accuracy of the model's answer extraction.

*3)* We conduct comparison experiments on four datasets, including SQuAD1.1, SQuAD2.0, NaturalQA, and NewsQA, with two evaluation metrics, including F1 score and EM score, experiment results show that the proposed model has excellent performance.



| | |
|---|---|
| **Passage** | Lethal Weapon is an American buddy cop action comedy - drama television series that is based on the film series of the same name created by Shane Black . The series was ordered on May 10 , 2016 and premiered on Fox on September 21 , 2016 . On October 12 , 2016 , Fox picked up the series for a full season of 18 episodes . On February 22 , 2017 , Fox renewed the series for a 22 - episode second season , which premiered on September 26 , 2017. |
| **Qusetion** | when does season 2 of lethal weapon come out? |
| **Answer** | September 26 , 2017 |

Fig. 1. Example of extractive reading comprehension.

## II. Related Work

MRC technology was first developed in the 1970s. In 1977, W.G. Lehnert et al. [10] designed the QUALM system, which used question-answering rules. In the 21st century, researchers integrated machine learning methods into MRC research. However, there were still drawbacks such as weak model generalization and insufficient feature extraction. The development of neural networks provided an opportunity for the advancement of MRC technology. From unidirectional attention mechanisms to bidirectional attention mechanisms, MRC technology has made significant strides and remarkable advancements. In recent years, the bidirectional pre-training language model BERT has achieved superior results in multiple task domains.

Reading comprehension tasks can be categorized as cloze tests, multiple-choice, span extraction, and generative reading comprehension.

Cloze-style tests prompt the machine to select the correct answer from a finite number of alternatives by removing words from the sentence. Representative datasets include CBT [11] and CNN/Daily Mail [4]. Representative models include the Gated Attention Reader [12] and others.

Multiple-choice reading comprehension has a more flexible answer format than cloze tests, as it is not limited to words or entities in context. However, the answers to the questions must still be provided in advance. Representative datasets for this type of task include MCTest [13] and RACE [14], while representative models include DCMN+ [15].

Currently, span extractive reading comprehension is the most popular task in this field, which is more challenging than traditional machine reading comprehension. The goal is to extract a contiguous span from a given text paragraph, which is not selected from a list of options.

Extractive reading comprehension models typically consist of four network architecture components: an embedding module, a feature extraction module, an information interaction module, and an answer prediction module. The embedding module converts each word in the passage and question into a fixed-length vector representation. To achieve this, a classical word vector encoding method such as Word2vec [16] can be used. The feature extraction module is often positioned after the embedding layer to extract context and question features separately. This module typically uses classical deep neural networks, such as recurrent neural networks and convolutional neural networks, to extract contextual information. The information interaction module is responsible for combining the encoded information of the paragraph and the question. It also captures the relationships between the words in the paragraph and the question to obtain their representations. The answer prediction module is located at the end of MRC systems and provides answers to questions based on the primary context.

In the extractive reading comprehension task, two classifiers are typically trained to predict the starting and ending indices of the answer. Common datasets for extractive reading comprehension include SQuAD, NewsQA, TriviaQA, SearchQA [17], and so on. Representative models include SpanBERT, BLANC [18], etc.

Although extractive reading comprehension has made significant advancements, its capabilities remain insufficient. Specifically, confining answers to a specific span within the context is still unrealistic. Generative reading comprehension requires machines to infer, summarize and provide open-ended answers from multiple passages of text. Of the four different types of tasks, generative reading comprehension is the most difficult. NarrativeQA [19] is a dataset that represents generative reading comprehension, and UniLMv2 [20] is a model that represents this type of task.

In addition to the task form, reading comprehension models can be structurally divided into a reading comprehension module and an answer prediction module. The reading comprehension module aims to answer the given questions based on the given passages. It is considered the core part of the model, where the model learns information from the input text passage and question and generates the input text representation. For instance, Seo et al. [5] proposed BiDAF, which uses a bi-directional attention mechanism to improve the interaction between the question and the text passage, resulting in a more effective representation of the input text. BERT, on the other hand, enhances word embedding through multiple layers of transformer. SpanBERT, which is built on top of the BERT architecture, further improves text comprehension in the continuous span extraction task by training with span mask and span boundary objective. In this paper, we use SpanBERT as the foundational architecture of our model.

The answer prediction module is divided into different types of tasks. For the cloze reading comprehension task, the module predicts the probability values of multiple candidate answers based on the text vector information and selects the option with the highest probability as the predicted answer. For the extractive reading comprehension task, the answer is a continuous segment of the given text. The answer prediction module should generate two probability distributions based on the text representation: one for the starting position of the answer and the other for the ending position.

## III. Model

Our model utilizes SpanBERT, a pre-trained model with 12 layers of transformer encoder, similar to BERT. The 12th layer is typically used for final answer prediction. However, language is complex and contains not only grammar and semantic information, but also hidden information such as emotion and deduction. Therefore, each layer of the transformer encoder learns different information during the training process. This model addresses the limitation of existing models that ignore other potentially helpful layers for answer prediction, by incorporating them into the prediction process. To improve answer prediction performance, this paper proposes a model that utilizes an information fusion approach. The basic architecture of our model is shown in Fig. 2.

First, both the passage and the question are input into the embedding layer for learning, resulting in the output of each layer of the transformer. Subsequently, the output of the middle layer and the output of the prediction layer are combined to obtain a fusion vector that contains the semantic information from both the middle layer and the prediction layer. Finally, the answer interval information predicted from the fusion vector is further fused with the answer information extracted from the prediction layer to obtain the final answer.

A thorough explanation of our model, including the encoding layer, encoder attention block, answer extraction, loss function, and other components, is given in this section.

### A. Embedding

Suppose the question sequence is Q=$[q_1, q_2, q_3, \dots, q_m]$ and the passage sequence is P=$[p_1, p_2, p_3, \dots, p_n]$. They are separated by a separator when inputting to the model, as shown in the following formula:

$$[\text{CLS}], q_1, q_2, q_3, \dots, q_m, [\text{SEP}], p_1, p_2, p_3, \dots, p_n, [\text{SEP}]\# \quad (1)$$

After inputting the text into SpanBERT, the encoding sequences $T_n$ $(n = 1\sim12)$ of each layer of the transformer encoder can be obtained through the encoder modules. When $n = 12$, $T_{12}$ represents the encoding sequence of the SpanBERT prediction layer, and the example of $T_n$ is as follows:

$$T_n = T_{[\text{CLS}]}, T_{q_1}, T_{q_2}, \dots, T_{q_m}, T_{[\text{SEP}]}, T_{p_1}, T_{p_2}, \dots, T_{p_n}, T_{[\text{SEP}]} \quad (2)$$

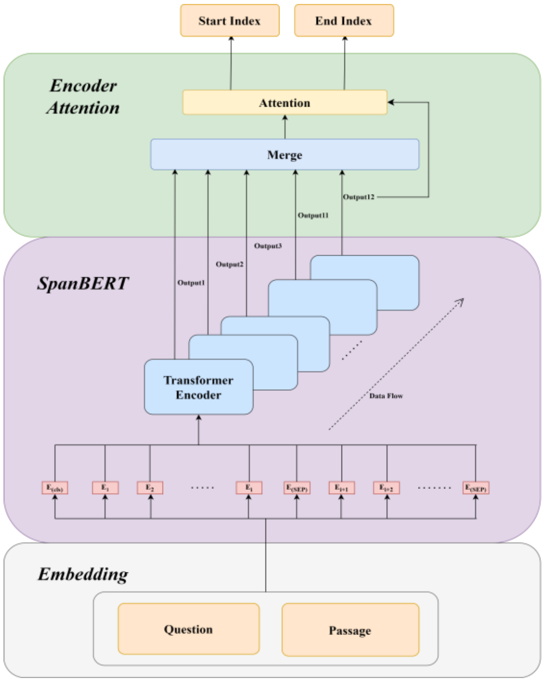


Fig. 2. Model architecture.

### B. Encoder Attention

The specific steps of information fusion are as follows:

*1)* Take out the results of the n-th layer and the last layer of the transformer encoder, and the encoding of the question is separately extracted to obtain the encoding information of the question for the two layers, denoted as $Q_n$ and $Q_{Last}$, respectively.

*2)* The semantic information is combined between the two layers by means of dot product, and then calculate the weight of each of the two layers through the fully connected layer. The formula is as follows:

$$Q'_n = SUM(Q_n) \quad (3)$$

$$Q'_{Last} = SUM(Q_{Last}) \quad (4)$$

$$E_{Attention} = Q'_n * Q'_{Last} \quad (5)$$

$$w_n, w_{Last} = Linear(E_{Attention}) \quad (6)$$

$Q_n$ and $Q_{Last}$ represent the encoded vectors of the question part in the *n*-th layer and the prediction layer of the transformer encoder. $SUM(*)$ is used to avoid the problem of inconsistent lengths of $Q$ in the input text by stacking the encoding information of $Q_n$ and $Q_{Last}$ according to the word encoding dimension. $E_{Attention}$ represents the fused semantic information. $Linear(*)$ is a linear function. $w_n$ and $w_{Last}$ represent the weights calculated for the *n*-th layer and prediction layer when predicting the answer, respectively.

*3)* By calculating the weights, we can obtain the fusion vector encoding as follows.

$$T_{merge} = w_n * T_n + w_{Last} * T_{Last} \tag{7}$$

$T_n$ and $T_{Last}$ represent the encoding sequences obtained from the *n*-th layer and the *12*-th layer of the transformer encoder, respectively. $T_{merge}$ represents the fused vector of the word encoding information.

### C. Answer-span Prediction

In the previous section, we fused the encoding of the *n*-th layer and the prediction layer to obtain a new fused vector. In this section, we propose a new approach for span prediction to improve the performance of the model. The formula is as follows:

$$pred_s = \frac{\exp(W_a T_{merge} + b_s^a)}{\sum_j \exp(W_a T_{merge_j} + b_s^a)} \tag{8}$$

$$pred_e = \frac{\exp(V_a T_{merge} + b_e^a)}{\sum_j \exp(V_a T_{merge_j} + b_e^a)} \tag{9}$$

$T_{merge}$ represents the fusion vector that fuses the word encoding information from the *n*-th and prediction layers. $W_a$ 、 $V_a$ 、 $b_s^a$ and $b_e^a$ represent the trainable weights and bias parameters. $pred_s$ and $pred_e$ represent the relative probability of each word as the beginning and ending position of the answer.

After obtaining the probability distributions of the answer span, which are represented by the fused vector of $pred_s$ and $pred_e$, the calculation of the guided-layer attention vector can be initiated. The formula is as follows:

$$Attention = pred_s * pred_e \tag{10}$$

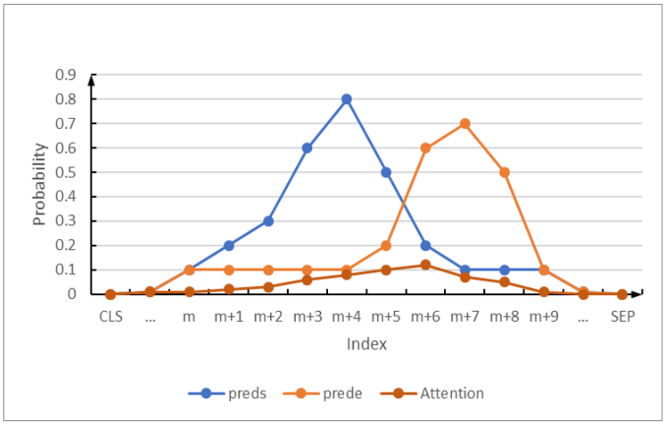


Fig. 3. Distribution of guided-layer attention vector.

The guided-layer vector represents the probability distribution of the predicted answer region. By multiplying $pred_s$ and $pred_e$, as shown in Fig. 3, the predicted probability value of words belonging to the answer span is higher than both ends, and the farther the distance is, the lower the probability value is. That is, the overall distribution of *Attention* is normal. The overall distribution exhibits a normal distribution, which could facilitate the ability to predict answer span for the guided-layer attention vector.

Finally, the answer prediction layer $T_{Last}$ and the probability distribution vector of the answer region are dot-multiplied, and then the final answer prediction is calculated by $softmax(*)$. The calculation formulas are as follows.

$$logits_s, logits_e = Split(T_{Last}) \tag{11}$$

$$Ans_s = logits_s * Attention \tag{12}$$

$$Ans_e = logits_e * Attention \tag{13}$$

$$p_{i=s_a} = \frac{\exp(W_a Ans_{s_i} + b_s^a)}{\sum_j \exp(W_a Ans_{s_j} + b_s^a)} \tag{14}$$

$$p_{i=e_a} = \frac{\exp(V_a Ans_{e_i} + b_e^a)}{\sum_j \exp(V_a Ans_{e_j} + b_e^a)} \tag{15}$$

$Split(*)$ represents the vector splitting operation. $logits_s$ and $logits_e$ are used to predict the start and end indices of context. $Ans_s$ and $Ans_e$ denote answer prediction vectors that have merged the information of the probability distribution vector of answer region. $W_a$ 、 $V_a$ 、 $b_s^a$ and $b_e^a$ represent the trainable weights and bias parameters.

### D. Loss Function

The loss function used in our model is the joint cross-entropy loss function, which is based on the negative log probabilities of the true answer's start and end positions in the predicted distribution. The formula is as follows:

$$L_{12} = -\frac{1}{N}\sum_{i=1}^{N}\left[\log\left(P_{y_i^s}^{start}{}_{Last}\right) + \log\left(P_{y_i^e}^{end}{}_{Last}\right)\right] \tag{16}$$

$P^{start}$ and $P^{end}$ are the probability distributions of the start and end positions of the answers predicted by the model. *Last* represents the prediction layer. $y_i^s$ and $y_i^e$ are the start and end positions of the real answer in the *i*-th training sample.

The same answer prediction is done for the fusion vector to obtain the loss function $L_n$.

$$L_n = -\frac{1}{N}\sum_{i=1}^{N}\left[\log\left(P_{y_i^s}^{start}{}_{n}\right) + \log\left(P_{y_i^e}^{end}{}_{n}\right)\right] \tag{17}$$

We define our final loss function as the weighted sum of the two loss functions:

$$L_{total} = (1 - \lambda)L_{12} + \lambda L_n \tag{18}$$

$\lambda$ is a hyper-parameter moderating the ratio of two loss functions.

## IV. Experimental Setup

### A. Datasets

To validate the effectiveness of the proposed model, experiments were conducted and analyzed on four datasets: SQuAD1.1, SQuAD2.0 [21], Natural Questions [22] and NewsQA.

The Stanford Question Answering Dataset (SQuAD) is a large-scale English reading comprehension dataset constructed by Stanford University, which has been an indispensable dataset for MRC tasks since its release and has a milestone significance for the development of MRC technology. SQuAD1.1 contains 536 high-quality articles from English Wikipedia, which are divided into natural paragraphs. In addition, there are 107,785 questions and corresponding answers, all of which are manually annotated.

SQuAD 2.0 builds on SQuAD 1.1 by adding unanswerable questions. Dataset creators provide an unanswerable question for each paragraph to interfere with the model's prediction. The training dataset contains 87k answerable and 43k unanswerable questions.

Natural Questions is a dataset of natural language queries. Each example consists of a Google query and a Wikipedia passage, where the answer is a span of the Wikipedia passage. The Natural Questions dataset contains 300,000 natural questions with human annotated answers.

The NewsQA dataset contains examples selected from over 10,000 news articles from CNN, along with 119,633 manually generated question-answer pairs. The answers are snippets of any length from the news article, and the dataset also includes partially unanswerable questions.

### B. Evaluation Metrics

F1 is the most widely used evaluation metric in existing extractive reading comprehension models. The Precision value is computed as follows:

$$Precision = \frac{TP}{TP+FP} \tag{19}$$

*TP* represents the number of true positive samples and *FP* represents the number of false positive samples. Then the Recall value is then calculated as follows:

$$Recall = \frac{TP}{TP+FN} \tag{20}$$

*FN* represents the number of false negative samples. F1 value is calculated from Precision and Recall with the following formula.

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{21}$$

EM (Exact Match) is a common evaluation metric for question answering systems, and it is also one of the main metrics for SQuAD. It measures the percentage of all predictions that exactly match the ground-truth answer. The calculation formula is as follows:

$$EM = \frac{N_{right}}{N_{all}} \tag{22}$$

$N_{right}$ indicates the number of correct predictions and $N_{all}$ represents the number of all predictions.

### C. Experimental Setup

The implementation of this model is based on Python and its third-party libraries, with PyTorch serving as the deep learning framework. The hyperparameters λ are set to 0.8 in joint loss functions. The experimental datasets used are SQUAD1.1, SQUAD2.0, NaturalQA, and NewsQA. To facilitate model performance testing and due to limited computing resources, the training batch size is set to 16 for the SQUAD2.0 dataset and 8 for the other three datasets. The learning rate is set to 2*$e^{-5}$, and the maximum length of input text is set to 384. The word vector dimension is set to 768, and the number of training epochs is set to 4 on the SQuAD2.0 dataset and 3 on the other three datasets.

### D. Baselines

BLANC [18]: To improve the accuracy of the final answer prediction, the model primarily employs a context prediction method. The model first predicts a soft label, then uses this soft label to calculate the context boundary probability, and finally uses the context boundary probability to optimize the final answer boundary prediction.

BERT-base [6]: BERT-base is a pre-trained language representation model that generates a deep bidirectional language representation using a masked language model (MLM). It is regarded as a landmark model in MRC, significantly advancing the field's development.

SpanBERT [9]: This BERT variation is tuned for fragment extraction tasks, resulting in more accurate representations. Two aspects contribute to the optimization. Firstly, it recommends adopting span masking rather than single-word masking for learning at fragments. Second, it trains the masked boundary words representation to anticipate masked fragment information.

ALBERT [23]: Compared to BERT, ALBERT overcomes the difficulties of extensive model parameterization and growing training time. It incorporates three major innovations: factorized embedding parameters, shared parameters across layers, and Sentence Order Prediction (SOP). The SOP creates not only positive examples by establishing the correct order of two consecutive sentences, but also negative examples by reversing their order.

LinkBERT [24]: LinkBERT is a cross-document language modeling training method that takes advantage of document links. In contrast with BERT, this approach has a distinct benefit in that it uses the links between documents to improve language modeling. LinkBERT treats the corpus as a document graph and employs linked documents as supplementary input to the model rather than modeling a single document.

DeBERT-base [25]: The proposed model aims to increase the robustness and effectiveness of the system when dealing with incomplete data. This is achieved by reconstructing hidden embeddings for sentences containing missing words.

OneS [26]: This model is based on the human learning model and introduces a new task of extracting essential knowledge from different knowledge sets for model pre-training.

KALA [27]: To solve the problem of catastrophic forgetting that occurs during adaptive pre-training, this model adjusts the intermediate hidden layer representation of a pre-trained model by incorporating knowledge from multiple domains.

ALBERTbase+V4ES [28]: This model proposes a verification mechanism that divides the machine learning process into two modules: general reading and fine-grained reading. The general reading module involves reading the text and question to obtain a preliminary answer. The fine-grained reading module reads again and generates a final answer.

RoBERTa-base [29]: This model enhances its performance by increasing the number of parameters and training data.

## V. Results and Discussion

### A. Results

In this section, two evaluation metrics (F1 and EM) are used on four datasets to verify the effectiveness of the proposed model in the paper. The experimental results are shown in Table I, Table II, Table III, and Table IV.

From the experimental results on the SQuAD1.1 dataset, the results in Table I show that the proposed model achieves good performance. The F1 score of our model improved by 0.6% and the EM score improved by 0.84% compared to SpanBERT. Compared to other models such as BERT-base, ALBERT-large, DeBERT-base, OneS and BLANC, the F1 scores are improved by 4.04%, 1.94%, 0.44%, 2.84% and 0.67%, and the EM scores are improved by 5.47%, 2.37%, 0.17%, 3.27% and 0.97%, respectively. In the experimental results of the SQuAD2.0 dataset, as shown in Table II, the F1 score of our model is improved by 0.59% and the EM score is improved by 0.76% compared to SpanBERT. In addition, compared to models such as BERT-base, OneS and ALBERTbase+V4ES, the F1 scores of our model are improved by 3.62%, 3.82% and 0.62%, and the EM scores are improved by 3.65%, 4.15% and 0.95%, respectively.

From the results of the NaturalQA dataset, as shown in Table III, the F1 score of our model is improved by 2.55% and the EM score is improved by 2.50% compared to SpanBERT. Compared to other models such as BERT-base, ALBERT and BLANC, the F1 scores of the model can be improved by 4.47%, 4.97% and 0.82% respectively, and the EM scores can be improved by 4.62%, 5.29% and 0.77%, respectively. From the experimental results of the NewsQA dataset, as shown in Table IV, the F1 score of our model improved from 67.93% of SpanBERT to 71.32%, with an improvement of 3.39%, and the EM score of the model is improved by 3.57%.

All of the above results show that our model has excellent performance.

Overall, our model performs better on the NaturalQA and NewsQA datasets when compared to the SQuAD1.1 and SQuAD2.0 datasets. This is because the SQuAD datasets have a flaw where the questions and answers are very similar, resulting in the front layer of the transformer losing less semantic information during the learning process. On the other hand, the NaturalQA and NewsQA datasets are generated based on real questions and answers, so more semantic information is lost in the front layer transformer during model learning.

### B. Effect of Different Layers on Results

There are 12 layers of transformer encoders in SpanBERT, but each layer focuses on different information. During its iterative learning process, some information about the answer in the earlier transformer encoder layers may be forgotten. In this section, F1 and EM are used as evaluation metrics to investigate the effectiveness of each layer in guiding the prediction layer to predict the answer, using SQuAD1.1, SQuAD 2.0, NaturalQA and NewsQA datasets as experimental datasets. The experimental results are shown in Fig. 4.

TABLE I. Results (%) of Experiments on the SQuAD1.1

| Models | F1 | EM |
|---|---|---|
| BERT-base | 88.50 | 80.80 |
| SpanBERT | 91.94 | 85.43 |
| ALBERT-large | 90.60 | 83.90 |
| LinkBERT | 90.10 | - |
| DeBERT-base | 92.10 | 86.10 |
| OneS | 89.70 | 83.00 |
| BLANC | 91.87 | 85.30 |
| ALBERTbase+V4ES | 91.10 | 83.40 |
| Our model | **92.54** | **86.27** |

TABLE II. Results (%) of Experiments on the SQuAD2.0

| Models | F1 | EM |
|---|---|---|
| BERT-base | 80.40 | 77.60 |
| SpanBERT | 83.43 | 80.49 |
| ALBERT-large | 82.30 | 79.40 |
| DeBERT-base | 82.50 | 79.30 |
| OneS | 80.20 | 77.10 |
| ALBERTbase+V4ES | 83.40 | 80.30 |
| RoBERTa-base | 83.70 | 80.50 |
| Our model | **84.02** | **81.25** |

TABLE III. Results (%) of Experiments on the NaturalQA

| Models | F1 | EM |
|---|---|---|
| BERT-base | 76.39 | 64.48 |
| SpanBERT | 78.31 | 66.60 |
| ALBERT-large | 75.89 | 63.81 |
| LinkBERT | 78.30 | - |
| BLANC | 80.04 | 68.33 |
| Our model | **80.86** | **69.10** |

TABLE IV. Results (%) of Experiments on the NewsQA

| Models | F1 | EM |
|---|---|---|
| BERT-base | 65.07 | 50.11 |
| SpanBERT | 67.93 | 52.85 |
| ALBERT-large | 66.02 | 51.18 |
| LinkBERT | 69.30 | - |
| KALA | 68.27 | 54.25 |
| BLANC | 70.31 | 55.52 |
| Our model | **71.32** | **56.42** |

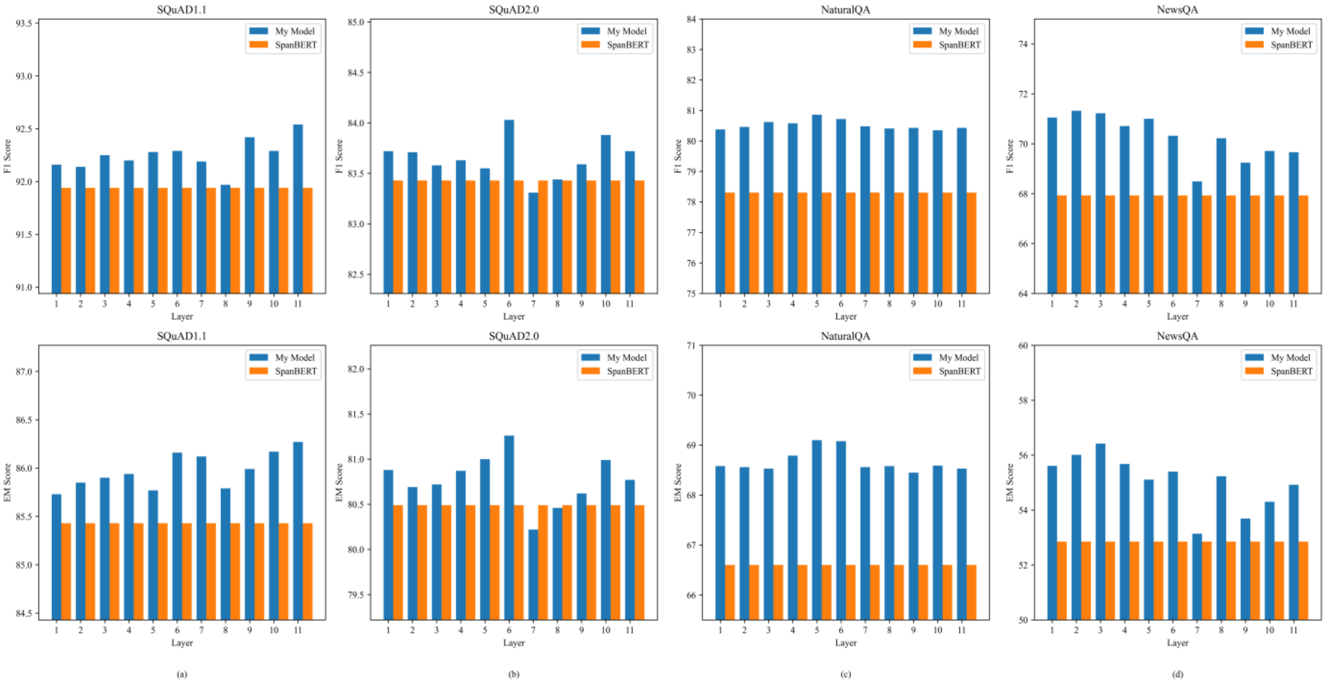


Fig. 4. (a), (b), (c), (d) represent the experimental results (F1/EM score) of different layers of SpanBERT as a guided-layer on SQuAD1.1, SQuAD2.0, NaturalQA, and NewsQA datasets (%), respectively.

From Fig. 4(a), we can see that on the SQuAD1.1 dataset, our model achieves optimal performance when using the output of the 11-th layer of the transformer encoder for fusion vector calculation. Specifically, compared to the SpanBERT model, the F1 score increased by 0.6% to 92.54% and the EM score increased by 0.84% to 86.27%. From Fig. 4(b), it can be observed that when using the prediction layer of the SpanBERT to directly predict the answer, the model achieves 78.31% (F1) and 66.6%(EM) on the NaturalQA dataset. However, when using different intermediate layers of the SpanBERT with the prediction layer to generate a new fused vector through encoder attention, the model achieves an improvement of 2.04% to 2.41% in F1 score and 1.93% to 2.5% in EM score. Among these experiments, the best performance was achieved when using the 5-th layer. This suggests that there is some semantic information loss in the early layers of the SpanBERT encoder during the iterative process. It also demonstrates the effectiveness of the proposed model.

As shown in Fig. 4(c) and Fig. 4(d), experiments on the SQuAD2.0 and NewsQA datasets, it was found that the best performance could be achieved when using the outputs of the 6-th and 2-nd layers of transformer encoder, respectively. Choosing different layers of transformer encoder to conduct experiments on different datasets always resulted in the best performance, indicating that our model is effective, and the 12 layers of transformer encoder in the SpanBERT model have differences in processing semantic information, and each layer focuses on different semantic information.

### C. *Effects of Hyperparameter* $\lambda$

In the experiment, a weighted sum of the joint loss function $L_{merge}$ and the loss function $L_{12}$ of the answer prediction layer was used as the total loss function of the model, and a hyperparameter λ was introduced to balance the weight of the two loss values. In this section, we use SQuAD1.1 and NewsQA as experimental datasets, with F1 score and EM score as evaluation metrics, to verify the optimal value of the hyperparameter λ in the joint loss function. The experimental results are shown in Fig. 5.

We set λ to [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0], and conduct experiments by incorporating λ into the calculation of the joint loss function $L_{total}$. As λ increases, the accuracy of the model in predicting answers increases. In the experiments on the SQuAD1.1 dataset, the model achieved the best performance when λ was set to 0.8, with an F1 score of 92.54% and an EM score of 86.27%. In the experiments on the NewsQA dataset, the model achieved the best performance when λ was set to 0.7, with an F1 score of 71.35% and an EM score of 56.13%. As λ increases further, the model's performance decreases.

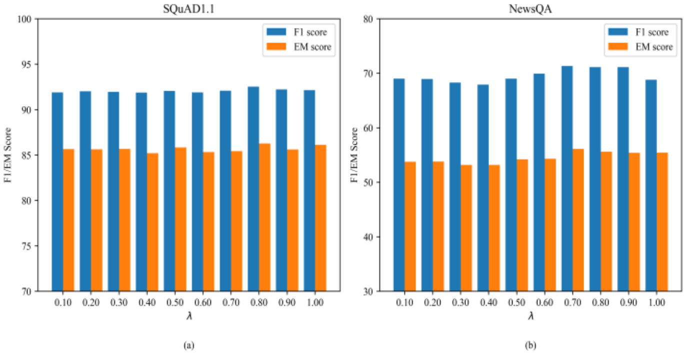


Fig. 5. (a) Results (%) for different values of λ on SQuAD1.1 dataset. (b) Results (%) for different values of λ on NewsQA dataset.

### D. Different Pre-trained Models

In the above experiments, we used SpanBERT as a pre-training model to verify the effectiveness of our model. In order to verify the applicability of this method to other pre-trained models, we also conducted comparative experiments by using two pre-trained models including BERT and SpanBERT. The experimental results are shown in Table V and Table VI.

TABLE V. RESULTS WITH DIFFERENT PRE-TRAINED MODELS (SQUAD1.1)

| | | SQuAD1.1 | |
|---|---|---|---|
| | | **F1** | **EM** |
| Bert-BASE | BASELINE | 88.10 | 80.49 |
| | +our method | **88.76** | **81.34** |
| SpanBERT | BASELINE | 91.58 | 84.97 |
| | +our method | **92.54** | **86.27** |

TABLE VI. RESULTS WITH DIFFERENT PRE-TRAINED MODELS (NEWSQA)

| | | NewsQA | |
|---|---|---|---|
| | | **F1** | **EM** |
| Bert-BASE | BASELINE | 65.07 | 50.11 |
| | +our method | **66.74** | **51.69** |
| SpanBERT | BASELINE | 67.93 | 52.85 |
| | +our method | **71.32** | **56.42** |

According to Table V, we can see that when using BERT as the pre-training model, our model can improve the model's F1 score by 0.66% to reach 88.76%. The model's EM score can be improved by 0.85% to reach 81.34%. When using SpanBERT as the pre-training model, the model's F1 score can be improved by 0.96% to reach 92.54%, and the model's EM score can be improved by 1.30% to reach 86.27%. From Table VI, on the NewsQA dataset, when using BERT as the pre-training model, our model can improve the model's F1 score and EM score from 65.07% and 50.11% to 66.74% and 51.69%, respectively. Using SpanBERT as the pre-training model, the model's F1 score can be improved by 3.39% to reach 71.32%, and the model's EM score can be improved by 3.57% to reach 56.42%. These results suggest that our method is applicable to other pre-training models.

### E. Effects of $T_{merge}$

In this section, we conduct comparative experiments using just the fusion vector $T_{merge}$ as the prediction layer and evaluate the performance. The experimental results are shown in Table VII and Table VIII.

TABLE VII. RESULTS OF $T_{merge}$ (NATURALQA)

| | NaturalQA | |
|---|---|---|
| | **F1** | **EM** |
| $T_{merge}$ | 78.96 | 67.21 |
| SpanBERT | 78.31 | 66.60 |
| Our model | **80.86** | **69.10** |

TABLE VIII. RESULTS OF $T_{merge}$ (NEWSQA)

| | NewsQA | |
|---|---|---|
| | **F1** | **EM** |
| $T_{merge}$ | 68.54 | 53.40 |
| SpanBERT | 67.93 | 52.58 |
| Our model | **71.32** | **56.42** |

According to the experimental results shown in Table VII, on the NaturalQA dataset, when using the fusion vector $T_{merge}$ as the prediction layer, the model achieved an F1 score of 78.96% and an EM score of 67.21%, which is higher than the pre-trained model SpanBERT's 78.31% and 66.60%. However, there is still a performance gap compared to the experimental results of the proposed model in this paper. Similar experimental results were also verified on the NewsQA dataset. We can see that our model still has the highest performance, which indicates that our model is effective.

## VI. CONCLUSION

In this paper, a SpanBERT-based multi-layer fusion extractive reading comprehension model is proposed. By fusing the representational information obtained from the intermediate transformer layer with the representational information obtained from the prediction layer, a new fusion vector is obtained through an encoder attention mechanism. Using the fusion vector, the distribution probability vector of the answer region is then computed and used together with the prediction layer to jointly predict the answer. Finally, answer extraction is performed. We have conducted extensive experiments to demonstrate the effectiveness of our model. Although the comparative experiments have shown the clear performance advantages of our model on the respective datasets, there are still certain problems and room for improvement. Specifically, even though the learning process of the pre-training model no longer suffers from the loss of semantic information, the learning process of our model still relies solely on the input data and does not use external knowledge, whereas people often use external knowledge to improve their understanding of textual data during the reading comprehension process. As a result, future studies can use the incorporation of external knowledge to enhance the semantic data, thereby improving the performance of the model. In the future work, we will also explore other pre-trained models for machine reading comprehension tasks.

## REFERENCES

[1] Rajpurkar P, Zhang J, Lopyrev K, Liang P. (2016). SQuAD: 100,000+ Questions for Machine Comprehension of Text[C]. Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, 2383-2392.

[2] Adam Trischler, Tong Wang, Xingdi Yuan, et al. (2017). NewsQA: A Machine Comprehension Dataset[C]. Proceedings of the 2nd Workshop on Representation Learning for NLP, 191-200.

[3] Joshi M, Choi E, Weld DS, Zettlemoyer L. (2017). TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension[C]. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, 1601-1611.

[4] Hermann K M, Kocisky T, Grefenstette E, et al. (2015). Teaching machines to read and comprehend[C]. Proceedings of the 28th

International Conference on Neural Information Processing Systems, 1693-1701.

[5] Seo M., Kembhavi A., Farhadi A., et al. (2017). Bidirectional attention flow for machine comprehension[C]. Proceedings of the 5th International Conference on Learning Representations, 1437-1450.

[6] Devlin J, Chang M W, Lee K, Toutanova K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding[C]. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics, 4171- 4186.

[7] Ganesh J, Sagot B, Seddah D. (2017). What does BERT learn about the structure of language?[C]. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 3651–3657.

[8] Ramnath S, Nema P, Sahni D, et al. (2020).Towards interpreting BERT for reading comprehension based QA[C]. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), 3236–3242.

[9] Joshi M., Chen D., Liu Y., et al. (2020). Spanbert: Improving pre-training by representing and predicting spans[J]. Transactions of the Association for Computational Linguistics, 64-77.

[10] Lehnert W. G. The process of question answering[M]. Yale University, 1977: 35-76.

[11] Hill F., Bordes A., Chopra S., et al. (2016). The Goldilocks Principle: Reading children's books with explicit memory representations[C]. Proceedings of the 4th International Conference on Learning Representations, 1124-1137.

[12] Dhingra B, Liu HX, Yang ZL, Cohen W, Salakhutdinov R. (2017). Gated-Attention Readers for Text Comprehension[C]. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, 1832–1846.

[13] Richardson M，Burges C J C，Renshaw E. (2013). MCTest：a challenge dataset for the open-domain machine comprehension of text[C]. roceedings of the 2013 Conference on Empirical Methods in Natural Language Processing，193-203.

[14] Lai GK, Xie QZ, Liu HX, Yang YM, and Hovy E. (2017). RACE: Large-scale ReAding Comprehension Dataset From Examinations [C]. Proceedings of the 2017 conference on empirical methods in natural language processing,785-794.

[15] Zhang S, Zhao H, Wu Y, et al. (2020). DCMN+: Dual co-matching network for multi-choice reading comprehension[C]. Proceedings of the AAAI Conference on Artificial Intelligence, 9563-9570.

[16] Mikolov T., Chen K., Corrado G., et al. (2013). Efficient estimation of word representations in vector space[C]. Proceedings of the 1st International Conference on Learning Representations, 976-988.

[17] Dunn M, Sagun L, Higgins M, et al. SearchQA: A new Q&A dataset augmented with context from a search engine[J]. arXiv preprint arXiv:1704.05179, 2017.

[18] Seonwoo Y., Kim J. H., Ha J. W.,Oh A. (2020). Context-Aware answer extraction in question answering[C]. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, 2418-2428.

[19] Kociský T, Schwarz J, Blunsom P, et al. (2018). The NarrativeQA Reading Comprehension Challenge [J]. Transactions of the Association for Computational Linguistics, 317-328.

[20] Bao H, Dong L, Wei F, et al. (2020). Unilmv2: Pseudo-masked language models for unified language model pre-training[C]. International Conference on Machine Learning, 642-652.

[21] Rajpurkar P., Jia R., Liang P. (2018). Know What You Don’t Know: Unanswerable questions for SQuAD[C]. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, 784-789.

[22] Kwiatkowski T, Palomaki J, Redfield O, et al. (2019). Natural Questions: a benchmark for question answering research[J]. Transactions of the Association for Computational Linguistics, 453-466.

[23] Lan Z., Chen M., Goodman S., et al. (2020). ALBERT: A Lite BERT for Self-supervised Learning of Language Representations[C]. Proceedings of the 8th International Conference on Learning Representations, 1362-1379.

[24] Yasunaga M, Leskovec J, Liang P. (2022). LinkBERT: Pretraining Language Models with Document Links[C]. Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics , 8003–8016.

[25] He P., Liu X., Gao J., et al. (2021). DeBERTa: Decoding-enhanced bert with disentangled attention[C]. Proceedings of the 9th International Conference on Learning Representations, 1278-1301.

[26] Xue FZ, He XX, Ren XZ, et al. One Student Knows All Experts Know: From sparse to dense[J]. arXiv preprint arXiv:2201.10890, 2022.

[27] Kang M, Baek J, Hwang S J. (2022). KALA: Knowledge-Augmented Language Model Adaptation[C]. In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 5144–5167.

[28] Peng Y, Li XY, Song JK, et al. (2021). Verification mechanism to obtain an elaborate answer span in machine reading comprehension[J]. Neurocomputing, 80-91.

[29] Liu Y, Ott M, Goyal N, et al. Roberta: A robustly optimized bert pretraining approach[J]. arXiv preprint arXiv:1907.11692, 2019.