Weakly Supervised Question Answering on SQuAD 2

Mohammad Hosein Moti Birjandi
Tehran Institute of Advanced Studies
m.h.motie@gmail.com

Abstract

Question answering is one of the NLP tasks that fascinated scientists. Reading comprehension is a type of QA task. The goal of this research is to figure out whether it is possible to only provide answer existence information to the model and, the model specifies answers spans with acceptable accuracy. We used two know interpretability techniques: attention score and gradient score. With these two techniques, we can see how models act to predict the existence of answers in the context and specify answer spans. We get 72% accuracy over the SQuAD-v2 dev set for predicting the existence of the answer. The BERT model was used but, the answers span generation was not enough accurate. So with the BERT-Base model and dev set of SQuAD-v2, we can not generate answer spans.

1 Introduction

In this work, we want to analyze a weakly supervised approach to figure out the answer to a question from the given passage. Generating datasets like SQuAD(Rajpurkar et al., 2016) and NewsQA(Trischler et al., 2016) needs a considerable amount of effort to annotate answers in the context. Many people may read the passages and answer the given questions and specify the start and end position of answers in the passage. What happens if we can tell the model whether the following context includes a response for the given question or not, and the model can determine the position of the answer span. The model only learns the existence of answers in the given context. Then we want to answer the following questions; which tokens have the most impact on the model decision, what semantic information model learned from questions and context, and can model return the span of the answer with admissible accuracy.

If the following method succeeds, **we can reduce the cost and effort for generating reading comprehension QA datasets.** There are two main methods to interpret a model; using attention or saliency. The goal of this work is to introducing a method for generating an answer span for a given non annotated text or demonstrate a reason for the failure of this approach. Some similar works tried to train a model by providing fewer questions, answer pairs to the model. These works used semi-supervised techniques like (Qu et al., 2021; Dhingra et al., 2018).

2 Model and Dataset

We first fine-tuned a **BERT base model** over the **SQuAD-v2**(Rajpurkar et al., 2018). We used the Huggingface library tools for the pre-processing dataset.

2.1 Dataset Pre-processing

As we have some constraints on computational power and RAM size we broke each example into examples with lower context length. The Huggingface tool for the SQuAD dataset pre-processing converts each example context to a set of spans. Each span has a possibility label that says the following span of context includes the question's answer or not. We used two sets of hyperparameters for finetuning the model. See the hyperparameters in Table 1.

The most important issue is that percentage of impossible cases will change after preprocessing. This change depends on the value of hyperparameters. You can see how the distribution of training datasets and evaluation datasets changed by pre-processing (Table 2). You can see the second hyperparameters holds dataset's distribution and in

hyperparameter	First	Second
max sequence len.	128	384
doc stride	32	128
max query len.	48	64
max answer len.	30	30
shuffle buffer	512	512
learning rate	2e-5	2e-5

Table 1: The Hyper parameters used for finetuning the model

the following we will see that the second hyperparameters set performs much better.

2.2 Model Specification

As we want to classify the possibility of answering the question by a given context, we need to add a classifier on top of the BERT model. The classifier is a feed-forward network alongside a softmax layer, and classification classes are possible and impossible. You can see the model architecture in Figure 1 (jal, a).

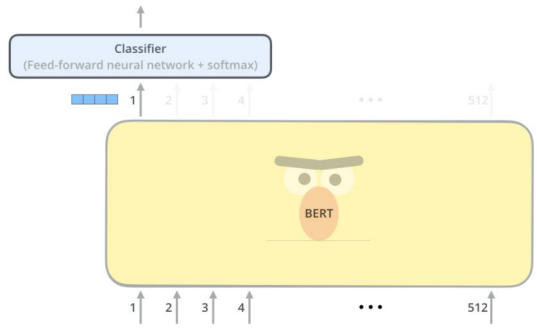

Figure 1: The model architecture. The classifier placed on top of the BERT [CLS] token output.

Then we fine-tuned the model by using the following two hyperparameter sets. You can see the loss and accuracy in Fig 2,3. Then for each example, we extended the Huggingface library to generate a boolean mask vector for contexts answers. In the SQuAD v2 dev set, each example may include four independent answers, but some of the answers are redundant. First, we remove all redundant answers in each example. After that, as the start position of answers in the dev set given by the character index in the context, we need to convert it to the token index. In the continued research, we take two approaches to distinguish whether it's possible to use mentioned idea on weakly supervised learning or not. The first approach is to use attention for interpret model prediction and, the second is to use saliency methods. The **fine-tuned model achieved 72.6% accuracy** in predicting the existence of the answer in the context text following the question. We used a **not fine-tuned BERT-Base model** as a **baseline**. The **accuracy of the baseline model is 49%**.

2.3 Attention-based Analysis

As a first analysis, we used the last layer attention score for the [CLS] token. We computed the **Spearman Correlation (SC)** between attentions weight and boolean masked vector of answers for each span. As some spans have multiple masked vectors, we take the maximum correlation value for that span in the calculation. In the results section, we will show how which category of prediction was the hardest category for the model.

2.4 Gradient-based Saliency

Then we used gradient-based saliency and drew a saliency map to check whether tokens with higher scores contain the answer or not. We want to understand it is possible to generate answer spans using gradient values or not. There are four attribution methods introduced in [] to perform well across various datasets for many kinds of tasks in BERT base models. In this work, we used the Gradient X Inputs method. In Figure 2(jal, b), generation of model output tokens illustrated. The model calculates logits in forwarding path "1" and next select a token based on the logits vector "2". In the end, the gradient of the selected logit is calculating with respect to the inputs by the back-propagation path 2.

The result is a signal of how important each token is in the calculation of predicting the selected token. The idea is that gradients measure how significantly the output change in response to feature small changes.

Then we multiply the resulted gradient vector per token by the tokens input embeddings(). To get the importance score, we need the aggregation method. In this work, we take the L2 norm of resulted vector in the previous step. (Figure 3(jal, b)) Also, we tried the Reduce Sum and Absolute Reduce Sum as aggregation methods and compared the results in the Table.

3 Results

In this section, we will provide performed experiments and results. First, we will issue fine-tuning

First Hyperparameters Set	train set	dev set	processed train set	processed dev set
impossible cases	43498	5945	171628	13270
total cases	130319	11873	274496	26704
percentage	33.38	50.07	62.52	49.69
Second Hyperparameters Set	**train set**	**dev set**	**processed train set**	**processed dev set**
impossible cases	43498	5945	44732	6129
total cases	130319	11873	131944	12232
percentage	33.38	50.07	33.90	50.11

Table 2: Distribution of the train and dev datasets before and after pre-processing

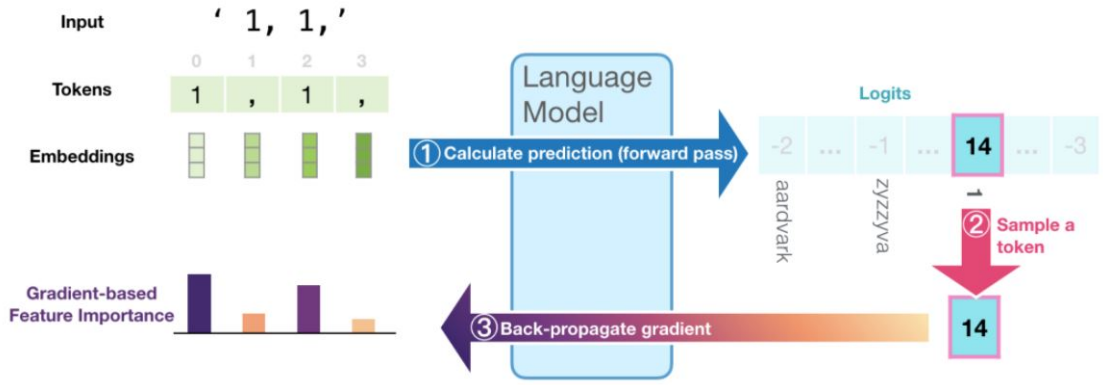

Figure 2: The procedure of calculating gradient of output with respect to input embeddings.

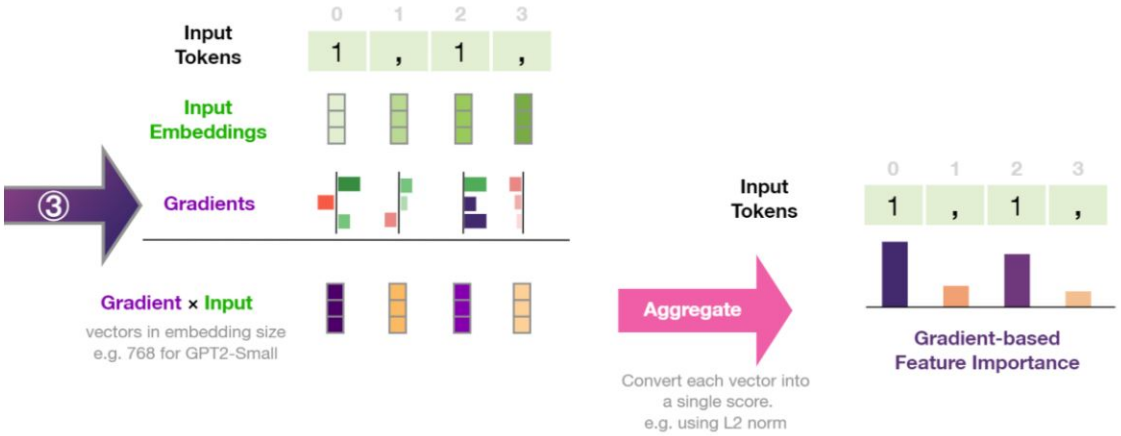

Figure 3: The aggregation of generated vector after multiplication with input embeddings using L2 norm.

challenges, accuracy and, the impact of hyperparameters on it. Then you can see the attention map of the [CLS] token and finally the results of gradients scores and saliency map.

3.1 Model Finetuning

As explained in the dataset pre-processing section, we need to set maximum length and doc stride hyperparameters to get a more accurate model. We tried three sets of hyperparameters and figured out that the second one was the best. The accuracy of the model in each epoch illustrated in Figure 4. After reaching 72.6% accuracy in predicting the existence of the answer in the context, we continued to the attention calculation process.

3.2 Interpretation

As explained before, one of the approaches to interpreting the model and understand how the model predicts the existence of an answer in the context is to visualize and analyze the attention scores. We get the attention of the [CLS] token

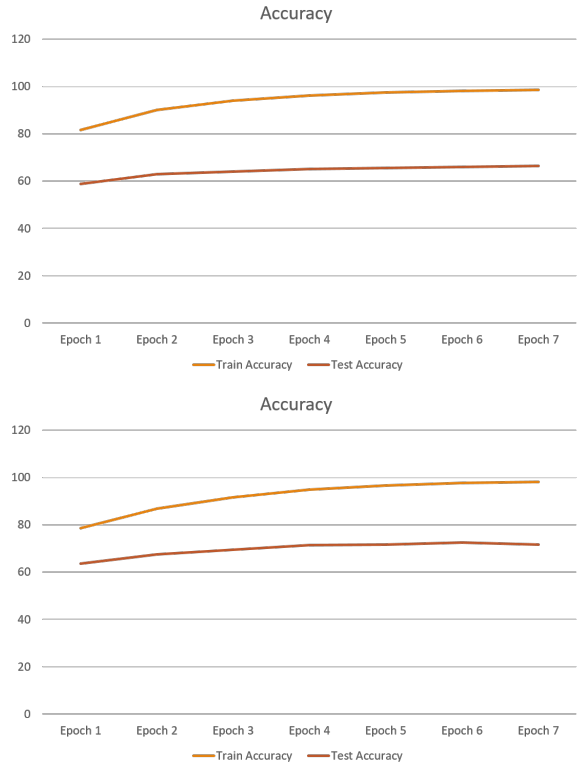

Figure 4: Model accuracy for different hyperparameter sets

by averaging attention's heads. Then we produce boolean mask vectors that are zero for all elements except answer tokens positions. As explained in the pre-processing section, we broke each context into multiple spans and, each span may contain the answer or not. Our focus is on the [CLS] token attention to other tokens; as the classifier only uses the output of it for prediction. We removed the attention score of the question tokens and the [SEP] token. Then we normalized the remaining values and made the sum of the attention on all remaining tokens equal to one. So for spans include answers, we drew an attention map of the [CLS] token to other words of the context(Figure 5).

In addition, we sketch the attention map for impossible to answer spans(Figure 8). You can see the [CLS] token attention at the final layer was on the punctuation and some conjunctions. The answer for the first question and context in Figure 5 is "France", but it did not take a high attention score. The dot punctuation after the answer token takes one of the highest scores. Finally, we calculated the Spearman correlation between attention scores and masked boolean vectors. In idealistic conditions, the [CLS] token must only pay attention to the answer tokens. So calculating the correlation between masked vector and attentions can be a suitable metric to figure out how much attention method is well to generate answer spans. You can see the comparison of the correlation values in the Table 1.

As you can see, our model learned how to detect whether the answer is in the context or not. But the value of SC is too low, so we can say attention is not a good metric to demonstrate answer span positions. In the next step, we computed the gradient of classifier probabilities with respect to input tokens embeddings. The relation between two classes probability and the meaning of softmax function is the only reason for considering probability instead of logit. For the same reasons in the attention section, first, we normalize gradient scores and then calculate the Spearman Correlation for the gradient scores vector and boolean vector mask. You can see the correlation values for the different aggregate techniques in Table 3.

To make it clear why Norm L2 and Absolute methods have better correlation scores, let's draw a heatmap of these three methods for the same example. As illustrated in Figure 8 if the absolute function does not apply on the reduced sum output, some of the tokens will get negative scores, but they are important and have more correlation with boolean masked vectors in other methods.

Unfortunately, non of the above techniques could not achieve a considerable correlation value. We tried to use the Average Precision (AP) metric to compare attention and gradients with mask vectors. If we can hit a higher value in this technique, we can search for an appropriate threshold value for answer span generation. The AP value for explained experiments was as follows.

Approach	**SC**	**AP**
Norm L2.	0.0964	0.0960
Reduce Sum	0.0299	0.0619
Absolute	0.0893	0.0766

Table 3: The correlation and average precision comparison for differnet aggregation techniques.

There is a bit of difference between AP and SC values. Although the SC and AP values increased in comparison with the attention approach, the incrementation is not too high. It forced us to check manually as many as possible examples and compare the output of these three aggregation methods. We must check another hypothesis manually

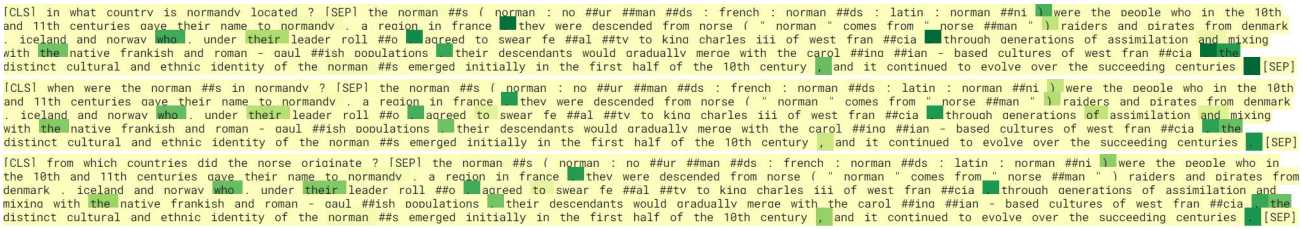

Figure 5: The [CLS] token attention map for existing answer examples.

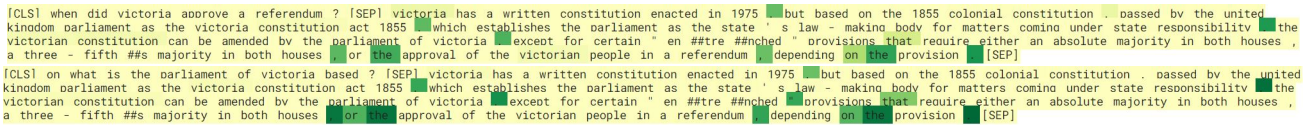

Figure 6: The [CLS] token attention map for spans doesn't include the answer in the context.

in this procedure. It was checking whether the answer was covered by the high gradient scores or not. If we could say most of the examples made the true answer green or evenly pale green, we can search for a threshold named "ground-line" (GL). All the tokens with higher scores than GL can be used for generating answer spans. But by the eye check, the accuracy of the result will be very low in comparison with the SQuAD-v2 dev set. So we checked examples. We found that some spans have overlap in the context. And obviously, some of the possible cases converted to two impossible cases, and only one of them contains an answer start position. These two reasons may have an influence on our results in model prediction and also in gradient score.

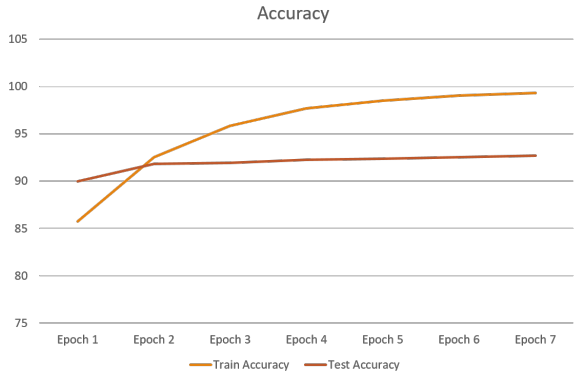

Figure 7: Model accuracy for different hyperparameter sets

As a final step, we tried to finetune the model on SQuAD-v1. We split the training dataset into two parts using an 80-20 percentage approach. The fine-tuned model reached 92.72% accuracy (Figure 7). The Spearman correlation improved for the saliency approach.

The distribution of the dataset after pre-processing is shown in Table 4.

	train set	**dev set**	**train set**	**dev set**
IMP	43498	5945	171628	13270
total	70080	17519	274496	26704
pct	33.38	50.07	62.52	49.69

Table 4: The dataset distribution for 80-20 method before and after pre-processing.

4 Conclusion

In this work, we wanted to use a weakly supervised method to respond to the existence of answers in the given context. In the next step, we tried to use the fine-tuned model and this method to annotate answers span with acceptable accuracy. The first step was successful and we reached 72% for the SQuAD-v2 dev dataset and 92.72 for SQuAD-v1.

Using interpretation techniques like attention scores and gradient scores to demonstrate answer spans was the second part of the research. The Spearman Correlation and Average Precision were two metrics to measure the accuracy in finding the position of the answers. In another language what made the model predict in such away. But the model failed to pass the second part of the research. In our opinion, there are three main factors for this result.

1. The overlap created between context span in our examples.

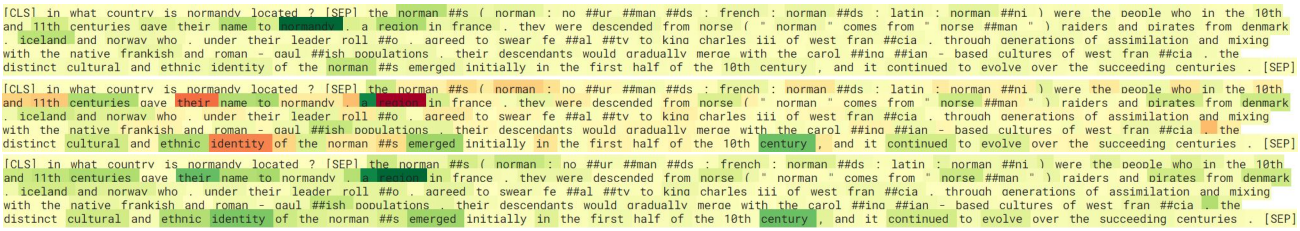

Figure 8: Saliency map for three aggregation methods on the same example. The first one used norm L2, the second used the reduce sum and, the third one used absolute reduce sum. The answer for this example is "France". Only the first one made the "France" token pale green.

2. Some contexts have an answer and are categorized as possible in the SQuAD dataset. But in the preprocessing, the context broke and now we have two kinds of impossible to answer questions. Original ones and generated from possible question and context pairs.

3. The questions in the SQuAD dev set were very close to the context but, answering is impossible based on the label.

5 Future Works

We used the BERT Base model for our research. Using the RoBERTA and ALBERT could be the continuation of this research.

References

Bhuwan Dhingra, Danish Pruthi, and Dheeraj Rajagopal. 2018. Simple and effective semi-supervised question answering. *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 2:582–587, 4.

The illustrated bert, elmo, and co. (how nlp cracked transfer learning) – jay alammar – visualizing machine learning one concept at a time.

Interfaces for explaining transformer language models – jay alammar – visualizing machine learning one concept at a time.

Chen Qu, Liu Yang, Cen Chen, W. Bruce Croft, Kalpesh Krishna, and Mohit Iyyer. 2021. Weakly-supervised open-retrieval conversational question answering. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 12656 LNCS:529–543, 3.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *EMNLP 2016 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, pages 2383–2392, 6.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, 2:784–789, 6.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2016. Newsqa: A machine comprehension dataset. pages 191–200, 11.