A Span-Based Question Answering Model with Disentangled Attention and Dual-Phase Reading

Gandhi Tejas, Kumar Prabhat, Sarji Elijah See Bona, Tan Jun Heng
CS4248 Group Project
National University of Singapore

1 Introduction

Question Answering (QA) remains a crucial area of research within the field of Natural Language Processing (NLP), it enables machines to comprehend and respond to queries when given relevant information.

In this report, we present a novel span-based Question Answering model, named the RetroReader, capable in handling answerable and unanswerable questions; designed to maximise the accuracy and reliability of such answer extraction tasks. The model draws inspiration from the human reading process, incorporating a Sketchy Reading phase for initial analysis, an Intensive Reading phase for span determination, and an answer verifier that assesses the credibility of the predicted span.

2 Exploring Related Work

In recent years, the field of Question Answering has experienced significant advancement, especially with the advent of neural network architectures. Span-based QA models have largely utilised transformer-based architectures like Bidirectional Encoder Representations from Transformers (BERT) as its cornerstone. A typical model will utilise a dual-head output design, where each respective head predicts the start and end position of the answer within the provided context (Gritsai et al., 2024).

Our model differentiates itself through its distinctive integration of a multi-phase reading process. Specifically, our model features an additional "Sketchy" Reading phase and Verification phase, alongside the standard "Intensive" Reading phase, which predicts the start and end positions of the answer span. The "Sketchy" phase of our model involves the evaluation of answer likelihood, assisting in determining unanswerable questions. Whereas the Verification phase assesses the overall confidence in the predicted answer span, serving as an additional layer of validation. The model's tri-phased approach refines its ability in discerning unanswerable questions, an area often overlooked by other models.

3 Dataset Preparation

Dataset used in Model include SQuAD-v1.1 and SQuAD-v2.0, the model can be trained separately on both datasets, where SQuAD-v1.1 gives the capability to predict the answer span, SQuAD-v2.0 gives the model ability to handle scenarios where the answer is not in the given context (Face, n.d.).

To prepare the input dataset, multiple tokenizers were selected and experimented upon, generating varied input representations for each question-context pair. This approach was initially intended to support a bagging ensemble model, where outputs from different tokenized models would be aggregated. However, due to computational constraints, the ensemble idea was quickly abandoned. Instead, the experimentation was utilised to help identify a single tokenizer that produced the best result when utilised by the model.

4 DeBERTa-Base:

Table 1: Performance Comparison Across Encoders

Model	Size	EM(%)	F1(%)	TT(min)
BERT	Base	80-81	88-89	20-25
BERT	Large	84-85	91-92	35-45
RoBERTa	Base	83-84	90-91	30-35
RoBERTa	Large	87-88	93-94	45-55
ALBERT	Base	79-80	87-88	15-20
ALBERT	Large	85-86	91-92	25-30
ALBERT	xxLarge	89-90	94-95	60-70
DeBERTa	Base	85-86	91-92	30-35
DeBERTa	Large	88-89	93-94	50-60
DeBERTa V2	xLarge	90-91	94-95	90-100
DeBERTa V2	xxLarge	91-92	95-96	120-140

TT - Training Time

Central to our model is the pre-trained language model DeBERTa-Base, employed for both tokenization and encoding. DeBERTa-Base was utilised for its optimal balance between computational efficiency during training and high accuracy in answer extraction, as shown in Table 1.

DeBERTa's performance can be primarily traced to the disentangled attention mechanism it employs. Unlike BERT, RoBERTa, and ALBERT, where a single vector for token embeddings that combines both content and positional information are employed, DeBERTa utilises two separate vectors: one for content and one for position (He et al., 2021). Such separation effectively provides a more accurate representation of the relationship between words and their positions in a sentence. This is evidenced by the observation that the attention weight of a word pair depends not solely on their contents but relative position as well. For instance, the relationship between the words "question" and "answering" is much stronger in occurrences next to each other as compared to in separate sentences.

5 Model Architecure:

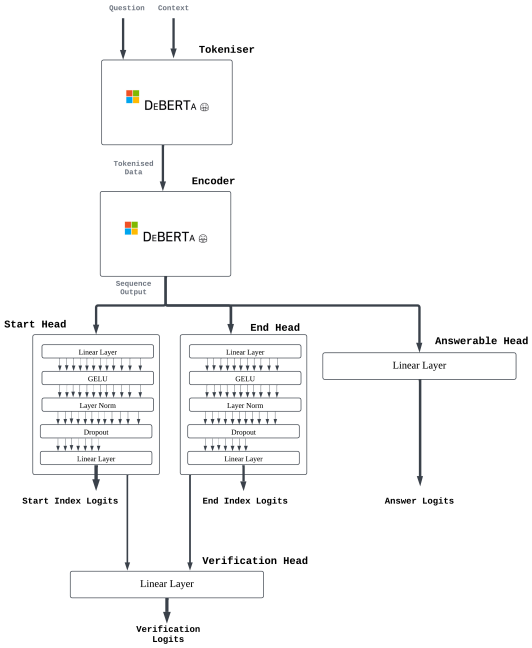

Figure 1: RetroReader Model Architecture

The first layer of the model begins with the encoding of each token in the input sequence, which is a concatenation of both the question and the context. It generates contextual embedding for each token, with each representing the contents of the token in the context of the entire input. Each contextual embedding is a derivation of three specific components: the token embedding, which encodes the lexical and semantic property of the token; the positional embedding, which encodes sequential structure of the token; and the segment embedding, which discerns between the question and the context. Mathematically, for each token i, the combined representation can be expressed as:

$$\begin{aligned} \text{Input}_i &= \text{Token\_Embedding}_i \\ &+ \text{Positional\_Embedding}_i \\ &+ \text{Segment\_Embedding}_i \end{aligned}$$

The contextual embeddings are then fed through the multi-layer encoder, processing them using the self-attention mechanism inherent to it. Dynamically, adjusting each embedding based on its relation with other tokens in the input sequence. Ultimately, generating refined and contextually rich embeddings for each token, that effectively encapsulates their context and role.

Following encoding of the tokenized input sequence, they are then passed through several task-specific heads:

- **Start and End Heads:** The Start and End Heads, belonging to the Intensive Reading Phase, are responsible for the identification of the exact span of text in the context that corresponds to the answer. The Start Head predicts the token position where the answer begins, while the End Head predicts the end. Both heads are implemented as neural networks, taking the contextual embeddings generated by the encoder as input. Both neural networks are structured similarly, with a series of transformations, including linear layers, activation functions, and layer normalisation steps that are applied to derive logits that represent the likelihood of each token being the start or end position of the answer span.

  Diving deeper into the model for each head, the process first begins with a linear transformation applied to each token embedding, setting up an initial weighted representation of the token's features. It then follows a non-linear activation function GELU (Gaussian Error Linear Unit), to introduce non-linearity in the prediction. Ensuing this, layer normalization and dropout is applied for better stability,

generalization and optimisation in the training process. The final step involves passing the output through a second linear layer to derive a logit representing the probability of the token being the start or end of the answer span. Mathematically, for each token embedding h_i at position i, the transformation is represented as:

$$h_i' = \text{Linear}(\text{Dropout}(\text{LayerNorm}(\text{GELU}(\text{Linear}(h_i)))))$$

To determine the predicted start and end positions, the model applies the argmax function to the derived logits, selecting the token index with the highest score for each start and end logits. The positions returned by the argmax function correspond to the tokens in the context that the model predicts as the most likely boundaries for the answer span.

- **Answerable Head (SQuAD 2.0 Specific):** For SQuAD v2, which contains questions that may not have an answer, the model includes an answerable head. This answerable head acts as the Sketchy Reading phase, it is responsible for determining whether an answer exists within the passage. Implemented as a linear layer, the answerable head produces a binary classification output that classifies the questions as either answerable or unanswerable. It takes in the embeddings of the [CLS] token generated by the encoder, which serves as a summary of the entire input to make this binary decision.

- **Verifier Head (SQuAD 2.0 Specific):** The Verifier Head in the model utilised in the Verification phase, assists in determining the validity of the answer predicted by the model. Using a linear layer, it processes a representation derived from the start and end token predictions of the context to compute a verification score. This score reflects the confidence in the predicted answer span being correct. Unlike the Start and End Heads, as well as the Answerable Head, which directly utilises the embeddings of the encoded context, the Verifier Head takes in the predictions for the start and end positions that it will combine to a single representation as input. This combined representation is done simply through an averaging technique. The final process in the Verifier Head can be mathematically represented

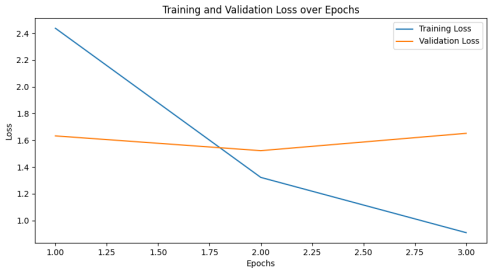

Figure 2: Training and Validation Loss over Epochs

below, where s' and e' are the embeddings corresponding to the predicted start and end positions that are extracted from the Start and End Heads.

$$s' = \textit{start\_pred}, \quad e' = \textit{end\_pred}$$
$$\text{VerifierInput} = \frac{s'+e'}{2}$$
$$\text{VerificationScore} = \text{Linear}(\text{VerifierInput})$$

The final prediction of the RetroReader model is made through the combination of results provided by each head. For non-SQuAD 2.0 tasks, the answer is simply generated by selecting the tokens with the highest probability as the start and end positions of the answer span. In contrast, for SQuAD 2.0 tasks, the logits from the Answerable Head are first passed through a softmax function to obtain an answerability score, which will be compared to a predefined threshold to determine its label. If the question is deemed answerable, the verification of the answer span will follow. This verification uses the verifier score produced by the Verification Head and compares it with another predefined threshold to ultimately determine the final answer span.

6 Loss Function

The loss function utilised in the RetroReader model is a multi-part loss function that optimises the model's performance across different tasks. It is the summation of the Cross Entropy Loss for the Start and End Heads, the Cross Entropy Loss for the Answerable Head, and the Binary Cross-Entropy Loss for the Verification Head. The aggregated loss is used to guide the optimization process, leading to better performance in both the span prediction, answerability, and verification tasks. During training, a learning rate scheduler is also incorporated to improve model convergence.

7 Hyperparameter Tuning

Key hyperparameters were experimented upon to optimise the model's efficiency and performance. It was discovered that the model achieved effective convergence within 2 epochs, as indicated by the alignment of training and validation losses, shown in Fig 2. Extending training beyond this point showed diminishing returns, with minimal improvements in performance and a risk of overfitting. For batch size, various values—specifically 8, 12, and 16—were tested. The batch size of 12 provided the best balance between computational efficiency and model performance. The smaller batch size resulted in longer training times within significant accuracy gains, while larger batch sizes introduced greater fluctuations in training loss.

In addition, several different strategies were deployed in the model to prevent overfitting, namely: Weight Decay, Layer Freezing, Dropout Regularisation, and L2 Regularization.

8 Results and Analysis

The proposed model was evaluated with two benchmark MRC challenges: SQuAD 1.1 and SQuAD 2.0. The SQuAD 1.1 offers detailed evaluation for the performance in the Start and End Heads. In contrast, the SQuAD 2.0 dataset goes further by challenging the model's ability to determine question answerability.

Table 2: Performance Comparison on SQuAD 1.1 and SQuAD 2.0

Metric	**SQuAD 1.1**	**SQuAD 2.0**
Exact Match	86.94	82.40
F1 Score	92.97	85.57
Total	10,570	11,873
Has Answer (Exact Match)	86.94	79.52
Has Answer (F1 Score)	92.97	85.87
Has Answer (Total)	10,570	5,928
No Answer (Exact Match)	-	85.26
No Answer (F1 Score)	-	85.26
No Answer (Total)	-	5,945

Two primary metrics, Exact Match (EM) and F1 score, were utilized in the model's evaluation. Exact Match (EM) calculates the percentage of predictions that match the ground truth answer exactly, focusing on the assessment of the model's capability in producing fully accurate responses. The F1 score measures the average token-level overlap between the predicted answer and the ground truth, allowing the observance of partial credit for predictions that capture significant sections of the correct answer.

The above performance metrics were achieved with the following hyperparameter configuration: the model was trained for 2 epochs with a learning rate of 2e-5 and a batch size of 8. Additionally, a threshold of 0.5 was used for both answerability and verifier heads. For SQuAD 1.1, the model was specifically trained on SQuAD 1.1 dataset. Whereas for SQuAD 2.0, the dataset used for training the model comprises a combination of SQuAD 1.1 and SQuAD 2.0.

The model demonstrates exceptional performance across both datasets, with high F1 scores indicating a strong ability to produce answer spans that align well with ground truths. The convergence of exact match and F1 scores, particularly on unanswerable questions in SQuAD 2.0, suggests that the model is well-calibrated to handle ambiguous contexts effectively. However, the slight dip in exact match scores in the SQuAD 2.0 answerable subset implies room for improvement in exact answer prediction in more complex datasets.

9 Challenges

The challenges faced in the development of the RetroReader model for Question Answering included: (i) Computational Constraints (ii) Tokenization Complexity, truncation was necessary slightly impacting answer accuracy. (iii) Overfitting Risks (iv) Small dataset, unable to develop a larger model.

10 Conclusion

RetroReader presents a robust solution for QA tasks, particularly in the handling of complexities with unanswerable questions. With the latest PrLM as the backbone and its mimicking of human reading comprehension, the model's strong performance across both SQuAD 1.1 and SQuAD 2.0 datasets confirms its potential for real-world applications. As future work, further refinements could focus on improving the model's ability to handle longer contexts and more diverse question types. Particularly, approaching it from the decoder-side, ensuring that they are optimised for better cooperation with the encoders to enable more advanced machine reading comprehension.

References

Yuanay An. 2023. Question answering using a pre-trained model in hugging face. *Medium*.

Hugging Face. n.d. Question answering.

German Gritsai, Ildar Khabutdinov, and Andrey Grabovoy. 2024. Multi-head span-based detector for ai-generated fragments in scientific papers. *Preprint*, arXiv:2411.07343.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention. *Preprint*, arXiv:2006.03654.

Gajanan Lokare. 2023. Question answering with hugging face transformers: A beginner's guide. *Medium*.

Zhuosheng Zhang, Hai Zhao, and Rui Wang. 2020. Machine reading comprehension: The role of contextualized language models and beyond. *Preprint*, arXiv:2005.06249.