Impact of Attention Mechanism on Question Answering in Transformers

Anonymous ACL submission

Abstract

This research explores the understanding of attention mechanisms in Transformer-based models, specifically BERT, to improve their performance on the question-answering task. We aim to identify the optimal number of attention heads that balance accuracy and computational efficiency. By contrasting the fine-tuned BERT model with varying attention heads against an LSTM model without attention mechanisms.

Our research question delves into how BERT's attention mechanism enhances its ability to accurately interpret and respond to questions across a diverse array of subjects compared to LSTM's sequential data processing, particularly in tasks that require deep understanding of context and nuanced language interpretation. We assess the impact of attention on handling complex queries when context is provided, specifically in the context of extractive question answering. Our experiments reveal that an increase in the number of attention heads significantly enhances model performance up to a threshold, beyond which the benefits diminish and may even degrade due to overfitting. This study addresses the critical need for efficient model fine-tuning strategies in natural language processing and provides insights into the effective deployment of attention mechanisms in language models for enhanced question-answering.

1 Introduction

Question-answering systems transitioned from early rule-based models to machine learning-based systems, and then to Recurrent Neural Network (RNN) approaches like LSTM and GRU. RNNs, while significant, faced challenges with long-term dependencies and slower sequential processing. This led to the adoption of transformer-based models like BERT and GPT, which enhanced domain-specific QA through fine-tuning on specific domain. Recent advancements in Natural Language Processing (NLP) have significantly enhanced the capability of machines to understand and respond to human language, making NLP crucial in various applications ranging from automated customer service to intelligent personal assistants. One of the most challenging tasks in this field is question answering, which requires a model not only to understand the natural language text but also to provide accurate answers from a given context. An exemplar of its necessity is in the domain of healthcare, where patients frequently seek specific information from vast databases of medical literature. Efficiently and accurately navigating these queries is essential for improving user experience and ensuring that reliable information is readily accessible.

This paper investigates the impact of attention mechanism and determining the optimal number of attention heads within the Transformer-based BERT model, aimed at optimizing its performance for the question answering task on the dataset. The dataset presents a unique challenge as it includes questions that are purposely unanswerable based on the given context, thus requiring the model to not only find correct answers but also identify when no valid answer exists. By comparing the results of a fine-tuned BERT model with various configurations of attention heads to those of an LSTM model without attention mechanisms, this study seeks to elucidate the role of attention in processing complex queries and improving answer accuracy. We explore the balance between model complexity and computational efficiency to provide insights into effective strategies for enhancing models in complex NLP tasks.

2 Background

In the field of question answering (QA), advanced neural architectures have gradually replaced rule-based models. Early attempts at quality assurance systems mainly depended on manually created rules and pattern matching, which took a lot of manual labor and domain expertise. These systems struggled with the complexity and variability of natural language, but they were good at responding to questions in specific domains. Following the advent of machine learning, researchers turned to statistical models such as RNNs (Recurrent Neural Networks) and Hidden Markov Models (HMMs). RNNs offered a dynamic framework for handling sequence data and made it easier for users to learn word dependencies in questions and contexts [@Jurafsky2000]. Despite advancements, these models continued to struggle with long-term dependencies and frequently lost context over longer texts—a critical component for accurate question answering.

The advent of Transformer models marked a turning point in NLP, especially for QA tasks. Introduced by Vaswani et al. in their seminal paper "Attention Is All You Need" [@Vaswani2017], Transformer models, eschewing recurrent layers for self-attention mechanisms, could process entire sequences of words simultaneously, providing a global understanding of the context. This ability to capture complex dependencies across a sequence of words enabled the models to achieve remarkable success on various NLP benchmarks. Subsequently, BERT (Bidirectional Encoder Representations from Transformers) and its derivatives further advanced the field by pre-training on vast corpora of text and fine-tuning specific tasks, including QA, achieving unprecedented levels of performance and truly revolutionizing the approach to automated question-answering systems [@Devlin2018BERT].

The pursuit of improving QA (Quality Assurance) systems extends well beyond academia and has significant real-world implications across multiple sectors. For example, in customer service, automated QA systems can reduce response times and improve the accuracy of support by providing answers to frequently asked questions. In the medical field, QA systems can quickly sort through vast amounts of research to offer healthcare professionals evidence-based answers that can greatly impact life-saving decisions. However, the effectiveness of these systems depends on their ability to not only retrieve information but also comprehend and reason about the context. This challenge is further compounded by the introduction of the SQuAD-v2 dataset, which includes unanswerable questions, requiring models to discern when a question cannot be answered given the text. Achieving this level of understanding was previously unattainable with simpler RNN architectures.

Transformer-based models, such as BERT, have broadened the scope of possibilities in machine learning, yet they introduce new challenges. A critical factor influencing the effectiveness of these models is the fine-tuning of attention heads. These heads determine the interaction among segments of the input sequence. Insufficient attention heads may not fully exploit the model's potential, whereas an excess may lead to overfitting and computational inefficiencies. Thus, it is essential to optimize the number of attention heads to balance performance with computational economy. This study, therefore, concentrates on the hyperparameter optimization of attention heads to maximize efficacy while minimizing unnecessary computational expenses.

We build upon the foundational BERT model developed by Devlin et al [1] and the LSTM advancements in sequence understanding, focusing on their applications in the nuanced domain of unanswerable queries presented in SQuAD v2 as introduced by Rajpurkar et al. [2]

3 Methodology

3.1 Dataset Overview - Dataset Link

The SQuAD (Stanford Question Answering Dataset) v2 is a benchmark dataset for machine reading comprehension tasks in natural language processing. Specifically, SQuAD v2 is designed for extractive question answering, where the task is to extract the answer to a given question from a provided passage of text. Dataset is designed to pose a significant challenge for machine comprehension models by including unanswerable questions. It builds upon the first version by adding approximately 50,000 unanswerable questions to the existing 100,000 question-answer pairs from SQuAD v1.1. These questions are created

to look similar to answerable ones but do not have a correct answer within the provided passages.

The structure of the SQuAD v2.0 dataset includes
"Title" : The title of the Wikipedia article from which the context is extracted.
"paragraphs" : Each title has one or multiple paragraph entries.
"context" : A paragraph from the Wikipedia article that contains the information necessary to answer the accompanying questions.
"qas" : Field containing question-answer entries.

Each question-answer entry has the following fields:

- "id" : Unique identifier for each question in the dataset.
- "Context": A paragraph from the Wikipedia article that contains the information necessary to answer the accompanying questions.
- "Question": A question that pertains to the given context. The question may or may not have an answer based on the context provided.
- "Answer": This field contains the answer to a question if one is available. It includes the text of the answer and the indices indicating the position of the answer within the context paragraph.
- "is_impossible": A flag indicating whether a question can be answered with the given context. If true, it means the question does not have an answer in the context.

In the SQuAD v2.0 dataset, each question is structured to elicit a specific answer from the provided context. If the question is answerable, the dataset contains an answer entry comprising the text span and its corresponding starting character index within the context. However, for questions deemed unanswerable, an empty "answers" list is provided. The questions within SQuAD v2.0 are meticulously crafted to challenge the comprehension and analytical abilities of both humans and machine learning models. These questions encompass various types, including fact-based inquiries seeking precise information from the text, definition queries prompting explanations of terms or concepts mentioned within the context, and reasoning challenges necessitating an understanding of causality, implications, or other nuanced relationships within the provided information. Through this diverse array of question types, SQuAD v2.0 effectively evaluates the capability of models to comprehend and analyze textual content across a range of cognitive tasks.

Contributors to the SQuAD v2.0 questions include crowdworkers hired through platforms like Amazon Mechanical Turk. These contributors were shown Wikipedia paragraphs and asked to create questions in their own words. For the unanswerable questions, they were provided with a paragraph on a certain topic and asked to write questions that were plausible but did not have answers in the given paragraph, often requiring a different context to answer.

SQuAD v2.0 thus presents a dual challenge for NLP models: not only must they find answers to direct questions where an answer exists, but they must also identify when no suitable answer is present, testing both the retrieval and comprehension capabilities of the models.

3.2 Data Preprocessing

3.2.1 LSTM

Initially, textual components like context, questions, and answers are tokenized, breaking them into discrete units such as words or subwords. This tokenization ensures uniformity across sequences. Subsequently, padding ensures all sequences are of equal length by appending a special token to shorter sequences. Word embeddings are then utilized to convert tokenized words into dense vector representations, effectively capturing semantic relationships. These embeddings aid the model in understanding the contextual meaning of the text.

Following this, input-output pairs are established for LSTM training. Each input sequence comprises a passage-question pair, while the output sequence contains the corresponding answer span or a marker indicating the absence of an answer. Batching optimizes training efficiency by grouping these pairs for simultaneous processing. Tokenized words and answer spans are then converted into numerical representations suitable for LSTM input. Finally, the dataset is partitioned into training,

validation, and test sets to accurately assess model performance and prevent overfitting.

3.2.2 BERT

Firstly, tokenization is applied to segment the input context and question into tokens using the same tokenizer employed during the pre-training of the BERT model. Special tokens such as [CLS] to mark the beginning and [SEP] to denote separation are added, along with [CLS] tokens inserted at the beginning of the answer span. Secondly, padding is implemented to standardize the length of tokenized sequences, with shorter sequences padded using a special token ([PAD]) and longer sequences possibly truncated to meet the model's maximum input sequence length. Attention masks are then generated to differentiate between actual words and padding tokens, facilitating the model's focus on relevant tokens during training and inference. Additionally, segment IDs are assigned to distinguish input sequences from different segments, enabling the model to understand the context-question relationship.

Furthermore, the preprocessing pipeline includes the labeling of answer spans within the tokenized context, identifying the start and end positions of the answer text, and creating label vectors to signify the answer span. The preprocessed data is formatted according to the BERT model's input requirements, comprising tokenized input sequences, attention masks, segment IDs, and answer span labels. Finally, the dataset is divided into training, validation, and testing subsets to facilitate model training and evaluation.

3.3 Algorithm Implemetation and Fine Tuning

3.3.1 LSTM Experimentation

In our research, we explored the effectiveness of the Bi-LSTM network in the context of question-answering tasks, specifically using the SQuAD 2.0 dataset. We developed an LSTM-based model augmented with dropout layers to mitigate overfitting, crucial for managing the model's complexity and enhancing its generalization capabilities. The model architecture consisted of an embedding layer, a bidirectional LSTM, and a linear layer to predict the start and end positions of answers within the context passages.

To facilitate training, we utilized a custom collation function for batching, which dynamically padded sequences to the longest sequence in each batch, ensuring efficient batch processing without fixed sequence length constraints. The training process involved optimizing the model using the Adam optimizer with an initial learning rate set through empirical tuning. Loss was computed separately for the start and end positions of the answers and averaged to update the model weights. Additionally, we employed a learning rate scheduler to reduce the learning rate when the training plateaued, optimizing the training phase for better convergence. Throughout the experiments, the model's performance was quantitatively assessed by tracking the loss and accuracy, providing insights into the effectiveness of LSTMs in handling the complexities of question-answering tasks. This experimentation is useful to study the performance of the model where the attention mechanism is not present.

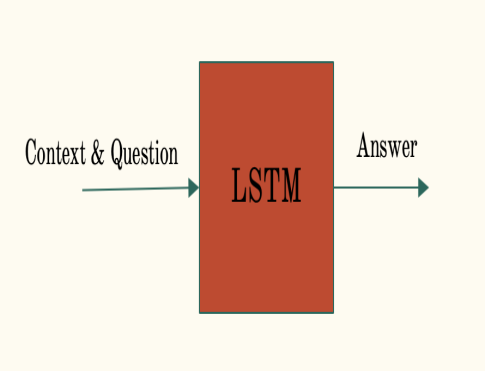

Figure 1: LSTM Pipeline

In our experimentation with Bi-LSTM for question answering, we initiated the process with comprehensive data preprocessing as mentioned in section 3.2.1, a vital step for preparing the dataset for effective model training. Following this, we implemented a Bi-LSTM model, chosen for its capacity to study the model which doesn't contain an attention mechanism. We then proceeded to the model training phase, where our LSTM model was exposed to various question-answer pairings, teaching it to predict answers based on the input context and question. After training over 20 epochs, we thoroughly evaluated the model.

3.4 BERT Expermination

The BERT-base model employs a transformer architecture with 12 layers, 768 hidden units, and 12

attention heads, utilizing pre-training on a large corpus of text to generate contextualized embeddings for diverse NLP tasks. We followed a strict protocol in our study to apply a pre-trained BERT-base-cased model which we fetched from hugging face and explored its inherent abilities to answer questions without changing the attention heads. Our approach consisted of a set of organized phases that started with data normalization. To guarantee consistency in input data, we refined textual content by removing articles, punctuation, and unnecessary whitespace. Our evaluation metrics relied on this normalization in two ways: first, to determine the exact match score—which measures the exact correspondence between the model's predictions and the ground truth—and second, to determine the F1 score, which measures the balance between precision and recall.

We systematically tracked the model's learning progress by recording loss and accuracy metrics, which included both F1 scores and exact match rates across 3 epochs. The model underwent rigorous training and validation cycles, where its performance on predicting answer spans was closely monitored and optimized. Crucially, model checkpoints were saved after each epoch, allowing us to preserve the state of the model at each stage of training. This careful monitoring and preservation of training stages provided valuable insights into the model's development and ensured the reproducibility of our results.

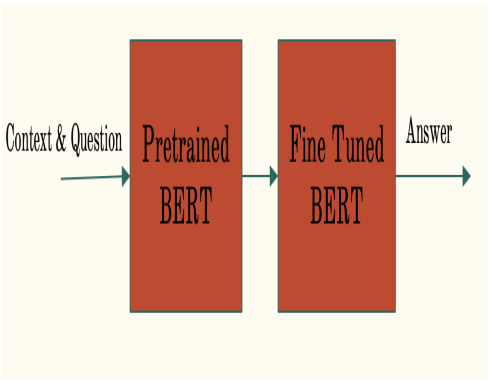

Figure 2: BERT Experimentation Flow

We have attained an accuracy of 72.65% using F1 metric and 57.6% using exact match metrics.

3.5 Attention Heads Experimentation

In our pursuit to study the impact of varying the attention heads on model performance and to check if there could be any other optimal no of attention heads in the architecture of BERT for question answering, we conducted a series of experiments utilizing Optuna, a hyperparameter optimization framework. The objective was to discern how varying the attention heads impacts the models' performance and to find the optimal number of attention heads that would yield the highest F1 score and EM score. By trialing different configurations, ranging from a narrower focus of 4 to an expansive 16 attention heads, we trained and evaluated each version of the model under identical conditions. The number of attention heads in models like BERT needs to evenly divide the hidden size to ensure that each head's attention computations have consistent dimensions. This alignment allows the model to parallelize learning across the heads efficiently and simplifies the overall architecture, promoting both computational efficiency and easier model maintenance. The experimentation was grounded in the hypothesis that a specific arrangement of attention heads could strike a balance between model complexity and performance. After rigorous testing and validation over multiple epochs, the results were aggregated, guiding us toward an evidence-based decision on the ideal number of attention heads.

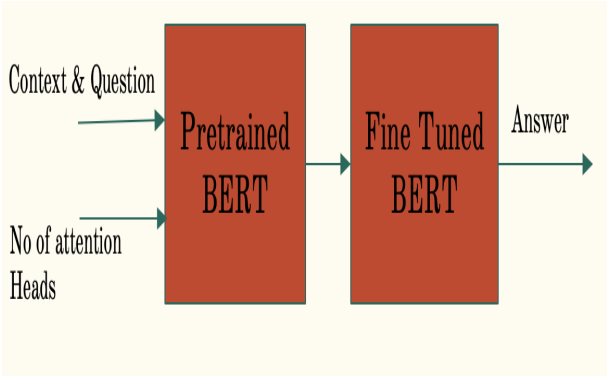

Figure 3: Attention Heads Experimentation

It is clear from the results that the least successful models in terms of F1 and Exact Match metrics are the LSTM model and its lack of attention mechanisms. This suggests that attention mechanisms are useful for answering questions because they allow a model to focus on different parts of the input sequence when making predictions.

Both evaluation metrics consistently improve as the BERT-based model's number of attention heads rises from 4 to 12. Moving from 12 to 16 attention heads causes this improvement to plateau and slightly decline, suggesting that there may be an ideal number of attention heads where the model

performs at its best before beginning to decline, possibly as a result of overfitting or the complexity of the model becoming too complex and not capturing nuances and context.

Model	Attention Heads	F1 score
LSTM	0	0
pretrained BERT (base)	4	1
pretrained BERT (base)	8	1
pretrained BERT (base)	12	1
pretrained BERT (base)	16	1

4 Evaluation

We evaluate our model's performance using Exact Match (EM) and F1 score to measure precision and recall in answer prediction, and BLEU score to assess the quality of generated answers compared to the ground truth.

5 Results

We have attained an accuracy of 53% on the LSTM Model and 70% accuracy on the BERT Model.

6 Conclusion Future Scope

Our project focused on BERT and LSTM models for text-based question answering. Future research could explore applying transfer learning and few-shot learning to enhance model adaptability to niche domains and low-resource languages. This approach, beyond our current scope, aims to improve NLP models' effectiveness in specialized areas without extensive labeled datasets, expanding their applicability and performance.

Sangpil: Current reference style is incorrect, please use bibliography with .bib file

7 Referencces

1. Devlin, J., et al. (2018). BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv.
2.Rajpurkar, P., et al. (2016). SQuAD: 100,000+ questions for machine comprehension of text. EMNLP.