Master's Thesis/Master's Research Project Report

A Study on Transformer-based Machine Comprehension with Curriculum Learning

BUI MINH QUAN

Supervisor Prof.NGUYEN LE MINH

Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology
(Information Science)

August, 2020

Abstract

In the early 1990s, the whole society had no idea about what is the Internet was or what it could do. Just several years later, it had exploded onto computers all over the world. It was a new evolutionary step for society. This also led to the explosion of information. Day by day, increasing amounts of data, humans have to process huge amounts of information every day. To save time and effort, support system such as information extraction and response system that can help humans select information from a large number of documents is essential. Since the 1960s, scientists have been interested in building question answering (QA) system for assisting people in finding information as well as questions that need to be answered. Typically, two early QA systems during this time period were BASEBALL [1] and LUNAR [2]. Both QA systems were very successful in their own domain. LUNAR has proved the usability of the QA system by answering exactly 90% of the questions in its domain posed by people did not have any experiment on the system. Therefore, people do not need to read and understand too many sources of information to come to a conclusion. They can rely on QA system to get answers.

QA system is divided into two types: open and close domain system. Open-domain is the system that can answer questions on many different topics in society. However, the amount of data for this system must be huge and varied on many topics. Not only that, but the growth of social networks has also increased a large number of fake data. This makes finding and processing information more challenging than ever. The close domain is a system used to respond to specific topics, so building data is quite natural because we can quickly check the accuracy of the data.

Thanks to the stable development of hardware, the recent success of deep learning (DL) is unbelievable. From categorizing objects in images and speech recognition to captioning images, understanding visual scenes, summarizing videos, translate language, paint, even produce pictures, speech, sounds, music and also in QA task. This is evidenced by the results of recent state-of-the-art deep learning architecture on open domain datasets beyond human performance.

In this study, We focus on two main things: developing a QA system for improving the health of the society by answering questions regarding nutrition and exercise, testing and slightly improving the performance of current state-of-the-art deep learning architectures on complex structure dataset by curriculum learning [3]. The results show that although these architectures

can surpass human performance in easily structured datasets, they perform poorly performance in complex structured datasets.

Keywords: Deep Learning, Question Answering, Performance, Hardware, Health, Nutrition, Exercise.

Acknowledgement

First of all, I would like to say thank you to my main supervisor Professor NGUYEN LE MINH, who not only gave me an opportunity to study at JAIST but also create the most favourable conditions for me to be able to complete this thesis. I would like to appreciate my brother BUI MINH NHAT for financially helping during my most difficult times. I would also like to acknowledge Professor Tojo Satoshi for his comments on my research proposal, and showing me the exciting things about school life in JAIST.

Second, I want to express my gratitude for everything that members in NGUYEN-sensei's laboratory have helped me achieve here. Specially, Mr. NGUYEN HA THANH and Dr. TRAN DUC VU for their comments, techinical supports and discuss about my research.

Lastly, thank you, mom and dad, for always treating me as the best child in the world. Thank you for encouraging me and helping me through the master program. You are the best parents in this universe for sure.

Author
BUI MINH QUAN

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Statement

Question answering (QA) is a computer science discipline within the fields of information retrieval and natural language processing (NLP), which is concerned with building systems that automatically answer questions raised by humans in natural language. QA can be found in many areas of NLP, including Natural language database systems, spoken dialogue systems, etc. There are two main types of questions in the QA task: factoid, which provides concise facts, and non-factoid is a big term that covers all questions answering topics beyond factoid question-answering. One of the most important factor that promotes the development of QA is the dataset. Historically, large open-domain has driven fields forward. One of the most famous datasets for QA task right now is The Standford Question Answering Dataset (SQuAD) [4] with more than 100,000 examples (triple of context, question, and answer) for factoid QA. In the other hand, The Whyset [5] dataset, which was used for why QA with 850 Japanese why-question and their top-20 answers passage 17,000 question-passage pairs for non-factoid. Thanks to these data sets and modern hardware, the development of machine learning models have been phenomenal in recent years. Despite the outstanding developments, there are still many issues in machine comprehension (MC) task.

- The main challenge of QA, and also NLP task is ambiguity. For example, existing a lot of words has different meanings depending on the context, leading to ambiguity on lexical, syntactic, and semantic levels.
- The second is synonymy, and we can express one meaning can express on different ways base on a specific context. For instance, a bat can describe the animal and baseball bat, but they cannot interchangeable in all context.

- Currently research, they usually focus on a specific type of question. For example, some research focus on factoid questions, some focus on why and yes-no questions.
- On Social media, there are a lot of fake and lack information articles on various topic. This the big challenge, because fake data might fool deep learning architecture and the prediction from the model would be wrong.

One of the approaches that we can deal with ambiguity is how we embed the data. Word embedding: word to vector shows that we can use a vector to present words in a way that captures semantic or meaning-related between words. One thing to realize is that one of the most popular pre-trained word vectors is Glove [6], which provides context-free embedding (static), cannot solve the ambiguity. But with text embedding such as ELMO [7], which use LSTM to give contextualized embeddings (dynamic), and also BERT [8] can deal with some cases of ambiguity. For instance, take these two sentences "I like apple" and "I like apple macbook". Word embedding will give the same embedding for the word "apple", but BERT will give a different one on context. For fake news problem, current research has released some DL model for fake news detection, and we can use this model to filter the fake articles. For why, how, yes-no question, we can extract the longer answer in the context, or we can generate the answer base on the context. Although many methods have been developed, they still cannot solve the problem thoroughly.

1.2 Objectives

The main purpose of our research is to create a QA system for the health domain called FitQA system. As the first step, we want to test the performance of current state-of-the-art deep learning model. Then we prepare FitQA with various types of questions. For further development, we will enhance the system with the capability of expert knowledge and reasoning.

1.3 Originality

There currently lacks researches achieving high-performance QA systems for specific domains, for instance, health, which can deal with complex data with expert knowledge and reasoning. Just for answer candidate extraction, current state-of-the-art DL architectures is extremely low with 47.3% F1

score on our complex structure dataset, despite achieving 93.16% F1 score on SQuAD factoid dataset. While our focus is health care systems, our approach could be adapted to other domains with similar characteristics (data complexities required expert knowledge and reasoning).

1.4 Thesis Outline

The specific content of this thesis is described as below:

- **Chapter 2: Related Works and Background :** We introduce an overview of related datasets, which help us come up with an idea to build FitQA, and some basic knowledge about Deep Learning networks.
- **Chapter 3: Methodology:** We show the detailed information of FitQA, state-of-the-art DL architectures and the training strategy that we applied in our experiments.
- **Chapter 4: Experiments:** In this chapter, experimental settings, evaluation metrics, tuning hyper-parameter and results are described.
- **Chapter 5: Discussion and Future Works:** We give not only several important information about limitation of current state-of-the-art DL architectures but also several possible methods for future work for our task.

Chapter 2

Background and Related Datasets

2.1 Related Datasets

Dataset is the most important part of machine learning systems. With a quality data set, the machine learning model can learn the important information contained in the data and can operate effectively. We will introduce some of the most popular datasets for QA system as following.

2.1.1 SQuAD dataset

Standford Question-Answering Dataset (SQuAD) [4] is a dataset for the QA system developed by Standford University. SQuAD contains more than 100,000 questions raised on Wikipedia articles. The answer for every single question is a small paragraph in the passage. Three steps build sQuAD: searching and filtering articles, crowdsourcing question-answers base on these articles, and conducting more answers.

- **Searching and filtering articles:** They used Project Nayuki's [1]. Wikipedia to obtain the most 10,000 quality articles from English Wikipedia. After that, they randomly picked 536 articles on various topic and extracted 23,215 paragraphs from these articles. Finally, they separated these paragraphs into training, development, test set to the ratio 8:1:1.
- **Crowdsourcing:** Firstly, they collect the answers by Daemo [9] and

[1] https://www.nayuki.io/page/computing-wikipedias-internal-pageranks

Amazon Mechanical Turk [2]. For every single paragraph, crowdworkers had to raise and answer about five questions base on paragraph's content. Moreover, crowdworkers must raise the questions by themself, the paraphrase is not allowed.

- **Additional Answers:** For enhancing the quality of SQuAD, they asked crowdworker to give more answers for each question in developemt and test set by selecting the shortest answer as could as possible in the passage.

Figure 2.1 can illustrate the structure of question-answer pairs and passage after being processed of SQuAD dataset.

SQuAD is a quality dataset, and many researchers had focused on developing DL models based on this data set. The results of these DL models almost passed human performance. This is the reason made Standford university upgraded SQuAD to the second version called SQuAD v2 [10]. In this version, except for the data from version 1, they added about 50,000 questions could not be answered into this dataset. This improvement made the performance of DL models at that time dropped 20%. The detailed of these adversarial examples are described as Figure 2.2.

2.1.2 NewsQA dataset

NewsQa [11] is a dataset developed by Microsoft. With more than 10,000 news articles from Cable News Network (CNN) [3], they created more than 100,000 question-answering pairs with an answer is contained within its article. They made NewsQA as follows:

- **Article Curation:** They retrieved a huge number of articles (90,266 articles) from CNN by the system of Hermann [12]. Then randomly picked 12,744 articles and separated them into training, development and test set to ratio 9:0.5:0.5.

- **Question Collection:** They want NewsQA to look more natural and challenge, then the questioners do not have a full article to refer and raise questions. With the only headline and some summary information, questioners had to raise questions without word overlap with summary information and at least three questions per article.

[2]https://www.mturk.com/
[3]https://edition.cnn.com/

In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravity. The main forms of precipitation include drizzle, rain, sleet, snow, graupel and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals within a cloud. Short, intense periods of rain in scattered locations are called "showers".

What causes precipitation to fall?
gravity

What is another main form of precipitation besides drizzle, rain, snow, sleet and hail?
graupel

Where do water droplets collide with ice crystals to form precipitation?
within a cloud

Figure 2.1: Question-answer pairs for a passage in SQuAD dataset

Reasoning	Description	Example	Percentage
Negation	Negation word inserted or removed.	Sentence: "*Several hospital pharmacies have decided to outsource high risk preparations . . .* " Question: "*What types of pharmacy functions have* ***never*** *been outsourced?*"	9%
Antonym	Antonym used.	S: "*the extinction of the dinosaurs. . . allowed the tropical rainforest to spread out across the continent.*" Q: "*The extinction of what led to the* ***decline*** *of rainforests?*"	20%
Entity Swap	Entity, number, or date replaced with other entity, number, or date.	S: "*These values are much greater than the 9–88 cm as projected . . . in its Third Assessment Report.*" Q: "*What was the projection of sea level increases in the* ***fourth assessment report****?*"	21%
Mutual Exclusion	Word or phrase is mutually exclusive with something for which an answer is present.	S: "*BSkyB. . . waiv[ed] the charge for subscribers whose package included two or more premium channels.*" Q: "*What service did BSkyB* ***give away for free unconditionally****?*"	15%
Impossible Condition	Asks for condition that is not satisfied by anything in the paragraph.	S: "*Union forces left Jacksonville and confronted a Confederate Army at the Battle of Olustee. . . Union forces then retreated to Jacksonville and held the city for the remainder of the war.*" Q: "*After what battle did Union forces leave Jacksonville* ***for good****?*"	4%
Other Neutral	Other cases where the paragraph does not imply any answer.	S: "*Schuenemann et al. concluded in 2011 that the Black Death. . . was caused by a variant of Y. pestis. . .* " Q: "*Who* ***discovered*** *Y. pestis?*"	24%
Answerable	Question is answerable (i.e. dataset noise).		7%

Figure 2.2: Types of 100 negative examples from development set of SQuAD 2.0

Reasoning	Example	Proportion (%) *NewsQA*	*SQuAD*
Word Matching	Q: **When were** the **findings published**? S: Both sets of research **findings were published Thursday**...	32.7	39.8
Paraphrasing	Q: **Who** is the **struggle between** in Rwanda? S: The **struggle pits ethnic Tutsis**, supported by Rwanda, **against ethnic Hutu**, backed by Congo.	27.0	34.3
Inference	Q: **Who** drew **inspiration** from **presidents**? S: **Rudy Ruiz** says the lives of US **presidents** can make them **positive role models** for students.	13.2	8.6
Synthesis	Q: **Where** is **Brittanee Drexel** from? S: The mother of a 17-year-old **Rochester**, **New York** high school student ... says she did not give her daughter permission to go on the trip. **Brittanee** Marie **Drexel**'s mom says...	20.7	11.9
Ambiguous/Insufficient	Q: **Whose mother** is **moving** to the White House? S: ... **Barack Obama's mother-in-law**, Marian Robinson, will **join** the Obamas at the **family's private quarters** at 1600 Pennsylvania Avenue. [Michelle is never mentioned]	6.4	5.4

Figure 2.3: Quantity Comparison of the Reasoning mechanisms between NewsQA and SQuAD

- **Answer Collection:** The Answerers are provided with full information about the articles. Their task is to answer the questions that are raised by Questioners. For every question, the answer is acknowledged if two Answerers choose it.
- **Validation:** To be sure to make a quality dataset. They had one more crowdsourcing, who is provided full articles, questions and answers. Their mission is to choose the best answer or remove all inappropriate answers.

They evaluate the complexity of NewsQA using the following criteria:

1. **Word Matching**: The duplication between words in questions and context. The more keywords in the question appear in the context mean this question is easy to extract.
2. **Paraphrasing**: Having at least one sentence look similar to the question.
3. **Inference:** This type is not easy to extract from the context. It must be inferred from information scattered throughout the context.
4. **Synthesis:** The answer is synthesized from a lot of information in the context.

Reasoning	**Lexical variation (synonym)**
Frequency	Major correspondences between the question and the answer sentence are synonyms. 41% in Wiki documents, 39% in web documents.
Examples	Q What is solid CO2 commonly called? S The frozen solid form of CO2, known as **dry ice** ... Q Who wrote the novel The Eagle Has landed? S The Eagle Has Landed is a book by British writer **Jack Higgins**
Reasoning	**Lexical variation and world knowledge**
Frequency	Major correspondences between the question and the document require common sense or external knowledge. 17% in Wiki documents, 17% in web documents.
Examples	Q What is the first name of Madame Bovary in Flaubert's 1856 novel? S Madame Bovary (1856) is the French writer Gustave Flaubert's debut novel. The story focuses on a doctor's wife, **Emma** Bovary Q Who was the female member of the 1980's pop music duo, Eurythmics? S Eurythmics were a British music duo consisting of members **Annie Lennox** and David A. Stewart.
Reasoning	**Syntactic Variation**
Frequency	After the question is paraphrased into declarative form, its syntactic dependency structure does not match that of the answer sentence 69% in Wiki documents, 65% in web documents.
Examples	Q In which country did the Battle of El Alamein take place? S The 1942 Battle of El Alamein in **Egypt** was actually two pivotal battles of World War II Q Whom was Ronald Reagan referring to when he uttered the famous phrase evil empire in a 1983 speech? S The phrase evil empire was first applied to the **Soviet Union** in 1983 by U.S. President Ronald Reagan.
Reasoning	**Multiple sentences**
Frequency	Requires reasoning over multiple sentences. 40% in Wiki documents, 35% in web documents.
Examples	Q Name the Greek Mythological hero who killed the gorgon Medusa. S **Perseus** asks god to aid him. So the goddess Athena and Hermes helps him out to kill Medusa. Q Who starred in and directed the 1993 film A Bronx Tale? S **Robert De Niro** To Make His Broadway Directorial Debut With A Bronx Tale: The Musical. The actor starred and directed the 1993 film.
Reasoning	**Lists, Table**
Frequency	Answer found in tables or lists 7% in web documents.
Examples	Q In Moh's Scale of hardness, Talc is at number 1, but what is number 2? Q What is the collective name for a group of hawks or falcons?

Figure 2.4: Reasoning Analysis of TriviaQa

5. **Ambiguous/Insufficient:** The answer is not in the context, or there is no way to answer this kind of question.

We can check the number of these types and the comparison between NewsQa and SQuAD is showed in Figure 2.3

2.1.3 Trivia dataset

Trivia dataset contains 650,000 question-answer-evidence triples. From these triples, Trivia dataset has about 95.000 question-answer pairs. Data collection process:

- **Question-answer pairs collection:** They crawled from 14 websites which contain trivia and quiz. After that, filtering short questions because these questions are too easy to answer.
- **Evidence Collection:** Evidence is collected by two method:
    1. Assuming question as a query and using Bing[4] to obtain top 10 websites contain the possible answer for the question.
    2. Applying TAGME [13] to find important words link question to Wikipedia pages and crawl the whole content as evidence documents.
- Finally, filtering and removing irrelevant articles.

The reasoning used to answer the questions from Trivia dataset is shown as Figure 2.4

2.2 Background Knowledge

The Transformer [14] is a deep DL architecture introduced in 2017. Like recurrent neural network (RNNs), Transformer are developed to handle sequential data, such as natural language, for tasks such as translation and QA. The special improvement thing of Transformer is that it does not require that the data be processed in order. There are also other advantages that make the Transformer successful in recent years. Thanks to these achievements, we have applied transformers to our experiments and the following we describe in detail the Transformer forming ideas and break Transformer down to see how it works.

[4]https://www.bing.com/

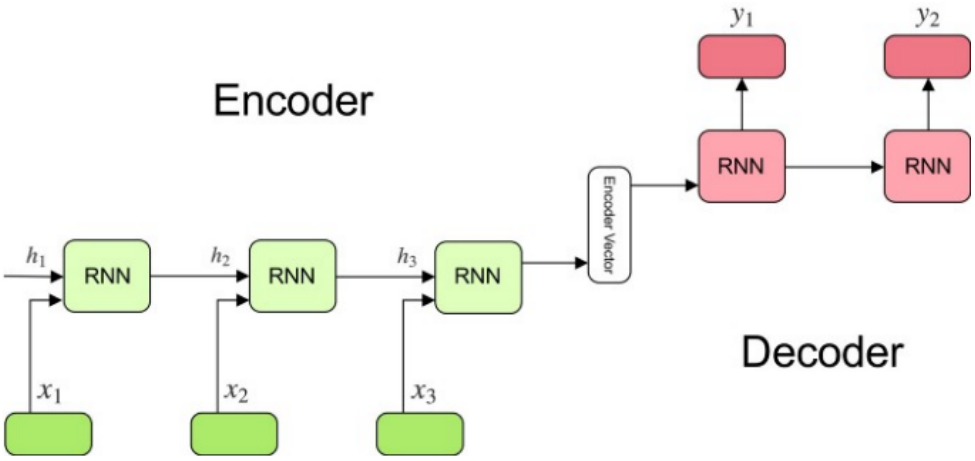

Figure 2.5: Illustration of Sequence-to-Sequence model

2.2.1 Sequence-to-sequence model

Sequence-to-sequence (S2S) [15] learning is introduced by Ilya Sutskever in 2014. This model focus on map the input and output whether they do not have the same length. For example, the input sentence "How old are you?" has 4 word and the output sentence "年はお幾つですか？" has 8 charaters. The model contain 3 pieces: encoder, context (encoder) vector and decoder. We can check the high-level look of S2S through Figure 2.5 [5]

1. **Encoder**

   - A list of some common recurrent units such as Long short-term memory (LSTM) [16] or Gated recurrent units (GRUS) [17] for better result. Where every single cell take a piece of the input sentence and obtain the information of that piece and feed it foward to the next cell.
   - In QA task, the input sentence is the list of tokens from the question.
   - The formula below show us how to calculate the hidden state:

   $$h_t = f(W^{(hh)}h_{t-1} + W^{hx}x_t) \tag{2.1}$$

   where:

   – W^{hh} is weight of recurrent cell.

[5] https://towardsdatascience.com/understanding-encoder-decoder-sequence-to-sequence-model-679e04af4346

- W^x is weight of input cell.
- h_t is current state.
- h_{t-1} is previous state.
- x_t is current input state.

2. Context Vector

   - The output vector of the last cell in encoder is context vector, which represent for the whole sentence.
   - This context vector feed into the first cell of decoder.

3. **Decoder:**

   - The same with the Encoder, decoder has a list of recurrent units to predict the final output y_t at current state t.
   - In QA task, each cell of decoder takes output vector of previous state and the word is extracted from the corresponding answer.
   - the current hidden state is calculated as formual below:

$$h_t = f(W^{hh} h_{t-1}) \tag{2.2}$$

   - And the output of the current state is calculated as following:

$$y_t = softmax(W^{hy} h_t) \tag{2.3}$$

     where:
     - $W^h y$ is weight at output state.
     - softmax is take a vector as input and give a probability as output.

2.2.2 Attention Mechanism

The big problem of S2S model is when it has to handle to the long sentence. The model can not keep the information of very first work in the sentence. It means that the context vector as the input of the decoder may miss some information. This causes the system to make a false prediction.

The attention mechanism [18] is introduced in 2014. This method helps the S2S model hold the information in the context vector better. As content in Figure 2.6, before passing to the decoder, the context vector seems to be linked to all the words in the input. So the decoder can have better information for predicting the result.

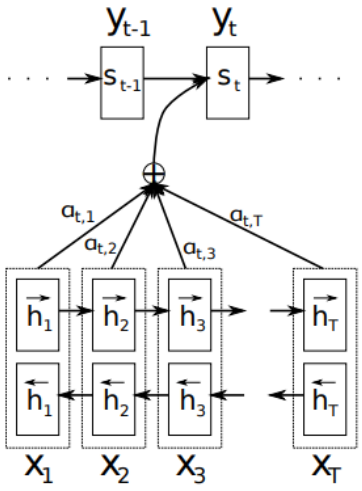

Figure 2.6: Bidirectional recurrent network with Attention Mechanism by Bahdanau et al., 2015

Demonstration Firstly, we assume that the input is a sentence x has n words and the output is a sentence y has m words.

$$\mathbf{x} = [x_1, x_2, x_3, ..., x_n] \tag{2.4}$$

$$\mathbf{y} = [y_1, y_2, y_3, ..., y_n] \tag{2.5}$$

Second, for the model S2S, we use bidirectional recurrent network such as LSTM, GRU, etc. Figure 2.6 show that this bidirectional recurrent network has 2 hidden state vector $\overrightarrow{\mathrm{h}}_t$ and $\overleftarrow{\mathrm{h}}_t$. The most simple way to keep information from two vector is concatenate them together. The ideal of this method is to keep information of the next word and previous word at the current state.

$$h_i = [\overrightarrow{\mathrm{h}}_i; \overleftarrow{\mathrm{h}}_i], i \in [1, ..., n] \tag{2.6}$$

The context vector is obtained by:

$$c_t = \sum_{i=1}^{n} \alpha_{t,i} h_i, t \in [1, ..., m] \tag{2.7}$$

$$\alpha_{t,i} = align(y_t, x_i) = \frac{exp(score(s_{t-1}, h_i))}{\sum_{i'=1}^{n} exp(score(s_{t-1}, h_{i'}))} \tag{2.8}$$

where *align* is alignment model calculate the compatibility between position of input i and the position of output t. And the *score* function:

$$score(s_t, h_i) = v_a^\top tanh(W_a[s_t; h_i]) \tag{2.9}$$

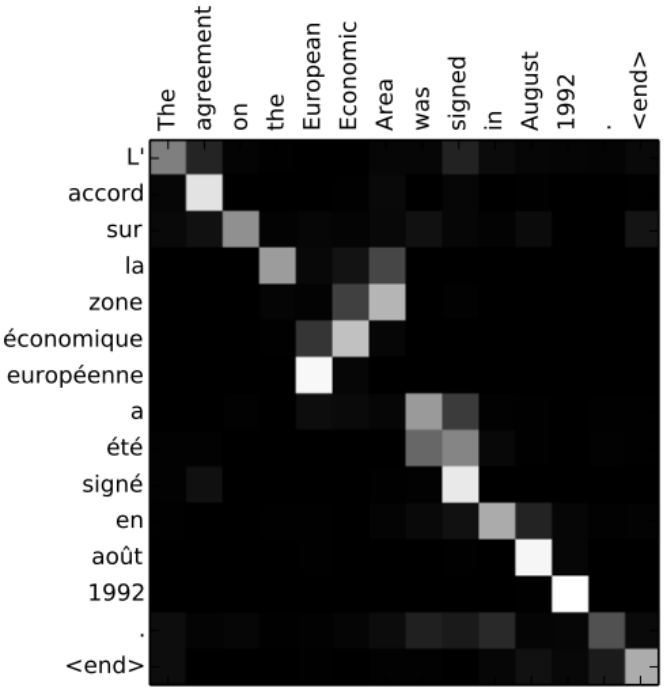

Figure 2.7: Sample is obtained by alignment model

where v_a and W_a is learnt from alignment model and tanh is non-linear activation function.

Through Figure 2.7 [18], We can see the close connection between input and output when attention is applied for machine translation task.

2.2.3 Transformer

Transformer is DL architecture that avoiding recurrence and not basing completely on attention mechanism to obtain context vector. It completely bases on self-attention to calculate representation of its input and output without using RNNs or Convolution Neural Networks (CNN) [19]. We describe in detailed Transformer as below.

Overview of Transformer

The main parts of Transformer are a stack of encoder and decoder. Stack of encoder contains layers of the encoder, and every single layer encoder has two layer within it. First one is self-attention layer, and the second one is a feed-forward neural network (FFNN). The decoder also has the same structure. However, between self-attention and FFNN decoder has an additional layer

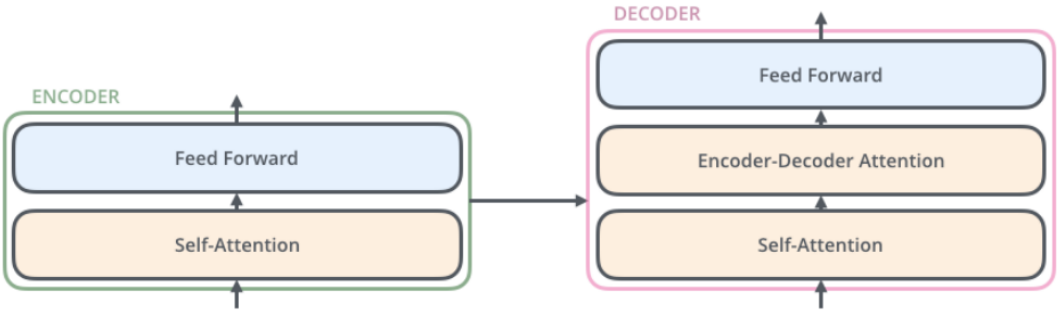

Figure 2.8: High-level look of Transformer model

that is encoder-decoder attention. This layer takes the information from encoder and help decoder memorizes relevant pieces of information from the input sentence. Figure 2.8 [6] can illustrate the flow of transformer model from input to the output.

Self-Attention

Self-Attention mechanism is a method to help the encoder to check all the other words of the input for figure out relevant parts in the sentence. This method can improve the current word has better embedding. For example, we have a sentence:

```
Mary can not go to work because she catches a cold.
```

Without information from other words, "she" here can not have any relation to Mary. This is the reason that self-attention is needed for embedding words form sentence.

To calculate self-attention for each word in the given sentence, we can refer to this formula:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.10}$$

where:

- Q: Query matrix.
- K: Key matrix.
- V: Value matrix.

[6] http://jalammar.github.io/illustrated-transformer/

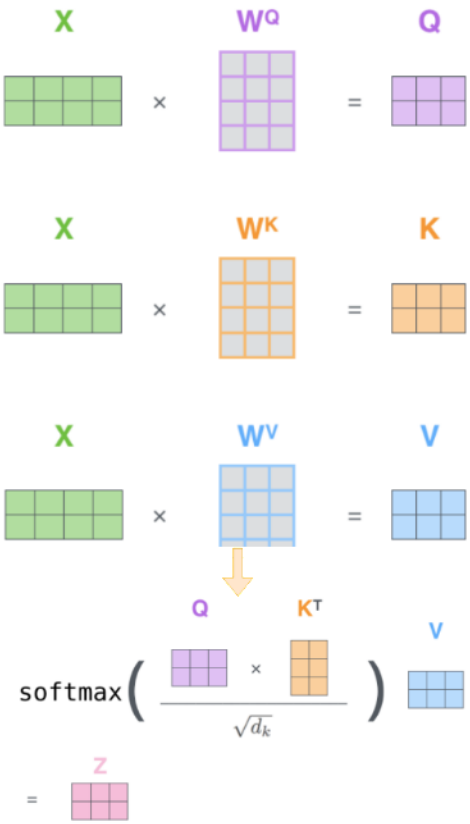

Figure 2.9: Process of calculating self-attention score

- softmax: generalized logistic function.
- d_k: the dimension of three vector Q, K.

To obtain Q, K and V vector, we sequential multiply the embedding matrix (X) to matric W^Q ,W^K and W^V, which are pre-trained matrices. Since we have done with matrices, we can apply the formula (2.10) to calculate the attention score for the current position. The process of calculating attention score is showed as Figure 2.9 [7]

Multi-Headed Attention

As Figure 2.9, this is just one head from many h head (h = 8 in Ashish Vaswani paper [14]) . It means that with the input sentence, we calculate eight times with eight different sets of W^Q ,W^K and W^V. This following

[7]http://jalammar.github.io/illustrated-transformer/

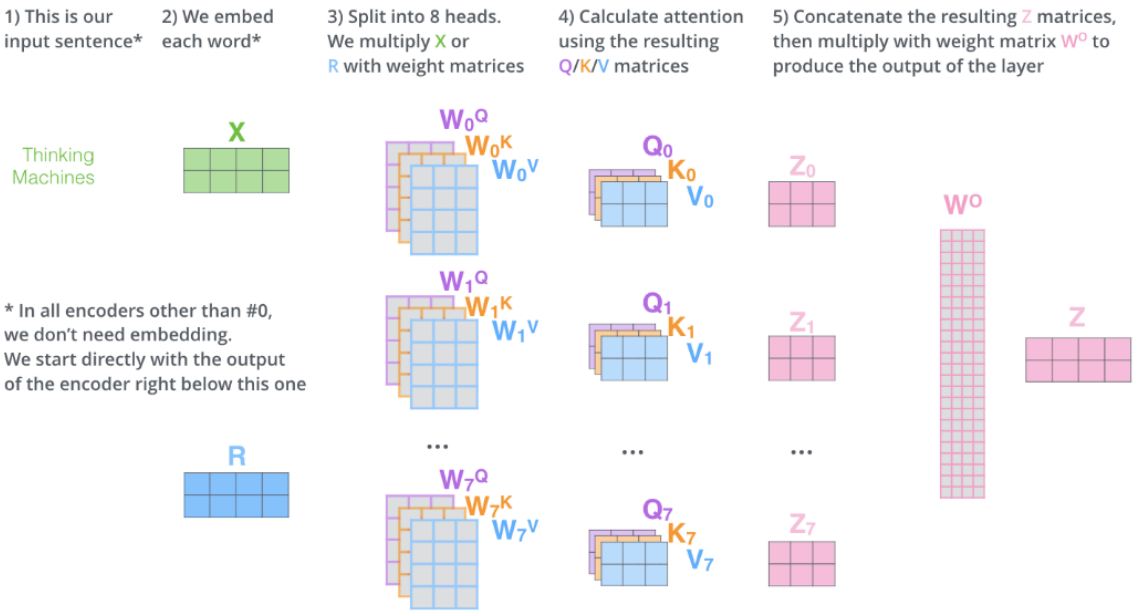

Figure 2.10: Multi-head attention illustration

formula calculates Multi-head attention:

$$\begin{aligned} MultiHead(Q, K, V) &= Concat(head_1, ..., head_h)W^O \\ head_i &= Attention(W_i^Q, KW_i^K, VW_i^V) \end{aligned} \tag{2.11}$$

where W^O is a weight matrix that was trained parallel with the model. We can have a high-level look of attention layer of Transformer through Figure 2.10 [8].

Decoder

The output of the final layer of encoder stack is matrix K and V. These matrices will be used by encoder-decoder attention layer within the decoder. The same to encoder stack, decoder stack also have six layers, multi-headed attention is also applied. The different between encoders and decoders is the encoder-decoder attention layer. This layer works the same way as multi-headed attention. However, instead of creating K and V matrix, they take K and V matrix from the encoders.

[8] http://jalammar.github.io/illustrated-transformer/

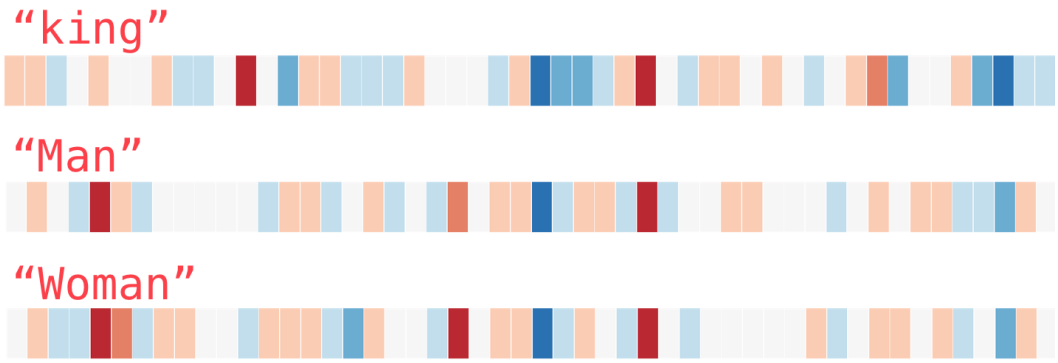

Figure 2.11: The similarity between words in GLoVe

Final layers

The output of the final decoder layer from decoder stack is a tensor of floats. We feed it into FFNN, and the output of this layer is a vector with dimension equal to the number of words that model learn from the dataset. In the other hand, it is vocabulary size. Every single element corresponding to the score of the word in our vocabulary. The final layer is the softmax layer, which converts elements score to probability. The word with the highest probability is taken as the output of the model.

2.2.4 Word Embedding Recap

As background knowledge, before feeding into DL model, we need the embedding layer to convert text into a number. Some word embedding such as Word2Vec [20] or GloVE [6] show that we can represent a word as a vector of number. For example, the vector representation for the word ”King” compare to ”man” and ”woman” as Figure 2.11 can illustrate that these vector has some relation between them. But the weakness of these pre-trained embedding is they present the same vector for one word in every context. For example, we have two sentences:

```
He is the king
```

and

```
This chess piece is king
```

The word ”king” in the two sentences is totally different. One is a human, and one is from chess and GloVe or Word2Vec present this word all the same. To overcome this weakness, every word has to have a different vector in a

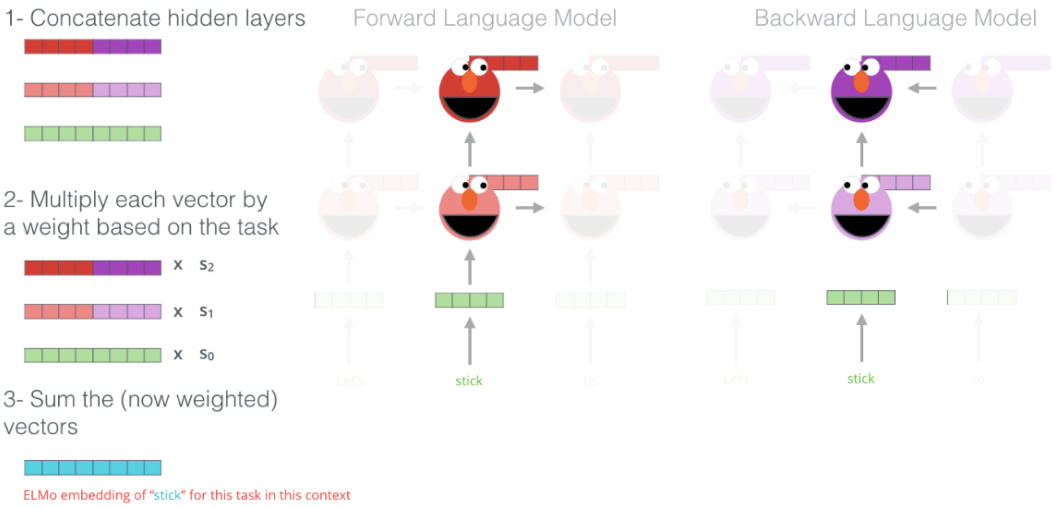

Figure 2.12: Example of ELMO for creating embedding for a word

different context. A deep contextual word representation called ELMO [7] is introduced in 2018, not like GloVe or Word2vec, ELMO creates an embedding for every single word in the sentence by using bi-directional LSTM to look for the whole sentence and allocate the vector embedding for this word. Base on model and attention mechanism, the word embedding from ELMO hold not only the information of the next word but also the information from the previous word. We can see how ELMO give an embedding for "stick" as the Figure 2.12 [9]

2.2.5 Bidirectional Encoder Representations from Transformers (BERT)

BERT basically is separated into two part: pre-training and fine-tuning. When pre-training, the architecture is trained on a huge number of training data which do not contain the label. Fine-tuning, it means BERT takes all the parameters which is trained in a various number of training examples, then these parameters are trained again with training examples which contain label from the specific task. This approach is a significant method to improve the performance of smaller tasks which have a limitation of training data.

[9] http://jalammar.github.io/illustrated-bert/

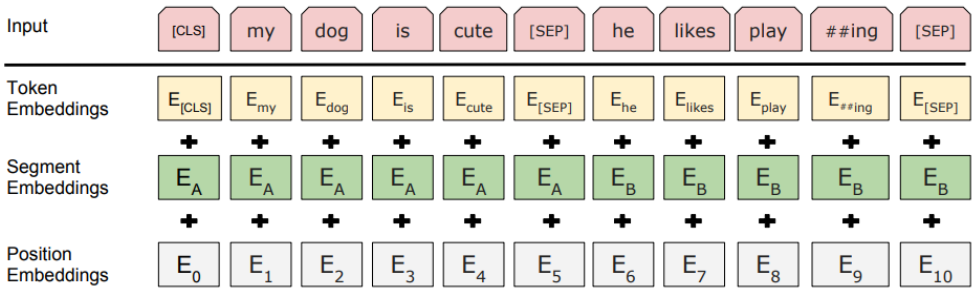

Figure 2.13: The progress of vectorizing input sentence

BERT Embedding

The architecture of BERT base on Transformer. BERT also contains a stack of encoders and decoders. The main purpose of encoder is to convert the input into a vector representation, and the decoder is to predict the result specific task. Since the pre-trained part is to trains and generates vector representation for the input text, the decoder only is enough. Before feeding into BERT, we have to do some pre-processing step:

1. **Tokenization**: The very first step is to split the input sentence into words, remove noise from data such as "/, * , etc". from the dataset.

2. **Token embedding:** In there work, they use WordPiece [21] to convert token into vector. Moreover, they add token [CLS] stands for classification as the first token of the sentence, and [SEP] token as the ending token of the sentence.

3. **Segment Embeddings**: To distinguish which sentence that words belong to, they use the additional embedding to each word to identify whether A or B is the sentence which contains that word.

4. **Positional Embedding:** The last embedding is used to indicate word position in the sentence.

5. **Final representation:** After conducting three embeddings, the representation for a word from the sentence is a vector by summing three embedding vectors. The input sentence representation is showed as Figure 2.13 [8].

Masked Language Modeling (MLM)

This approach making BERT becomes special in NLP community. Instead of embedding from right-to-left or left-to-right, BERT randomly converts word token into [MASK] token with probability equal to 15%. The mission of the model is to predict what is the [MASK] token corresponding to which word. The problem is [MASK] token does not apply in the fine-tuning process but the pre-training process. This reason makes some incompatible from pre-training to fine-tuning process. The solution for this issue is with tokens are replaced by [MASK]:

1. 80% from 15% tokens are converted into [MASK] token.
2. 10% from 15% tokens are converted into any word.
3. 10% from 15% tokens stay the same.

Next sentence Prediction

BERT not only uses [MASK] token to improve the quality of embedding but also uses next sentence prediction (NSP) to enhance the performance of tasks which need reading comprehension such as QA task. During the pre-training process, the model takes pairs of sentences A and B as input and try to predict whether B is the next sentence of A.

As the above information, We have mentioned about [SEP] token. During the pre-training process, BERT splits two sentences by [SEP] token and feed into the model as follow:

- The probability of the next sentence is the right next sentence is 50%.
- The probability of the next sentence is a random sentence in our dataset is 50%.

BERT has to predict whether the second sentence is the next sentence of the first sentence. For example, the input of next sentence and not next sentence are shown as Figure 2.14 [10]

BERT using MLM and NSP at the same time. The reason for this method is to reduce the loss during the training process.

[10]https://towardsml.com/

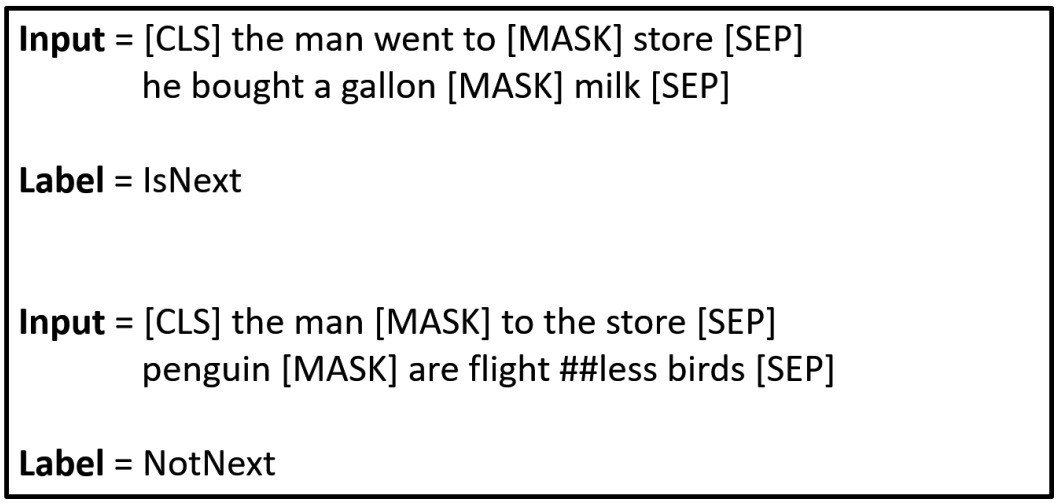

Figure 2.14: Example of Next sentence Prediction

How to use BERT for QA Task

The QA task basically is that QA system takes the question and relevant article and give an answer as the output of the system. We fine-tune by letting BERT takes not only question and article but also the starting and ending position of the answer which contains within the article. To predict the answer, BERT has 2 vector S and E (as Start/End Span in Figure 2.15 [8]) stand for start position and ending position. The probability of picking these position is explored by applying softmax function to all the words in the article, and the following formula calculates the probability:

$$P_i = \frac{e^{S \cdot T_i}}{\sum_j e^{S \cdot T_j}} \tag{2.12}$$

where T_i is a vector representation of position being calculated, T_j is the other vector representation position in the article. After obtaining the set of candidate span, the score for each pair of starting and ending point is computed by $S \cdot T_i + E \cdot T_j$ and the pair has highest one under condition j >= i is chosen as the output of the model.

2.2.6 ALBERT

The success of BERT shows that the pre-training step on huge dataset has significant improvement on almost NLP tasks like QA or translation. However, there is a weakness still exist in BERT, the serious of them is typically

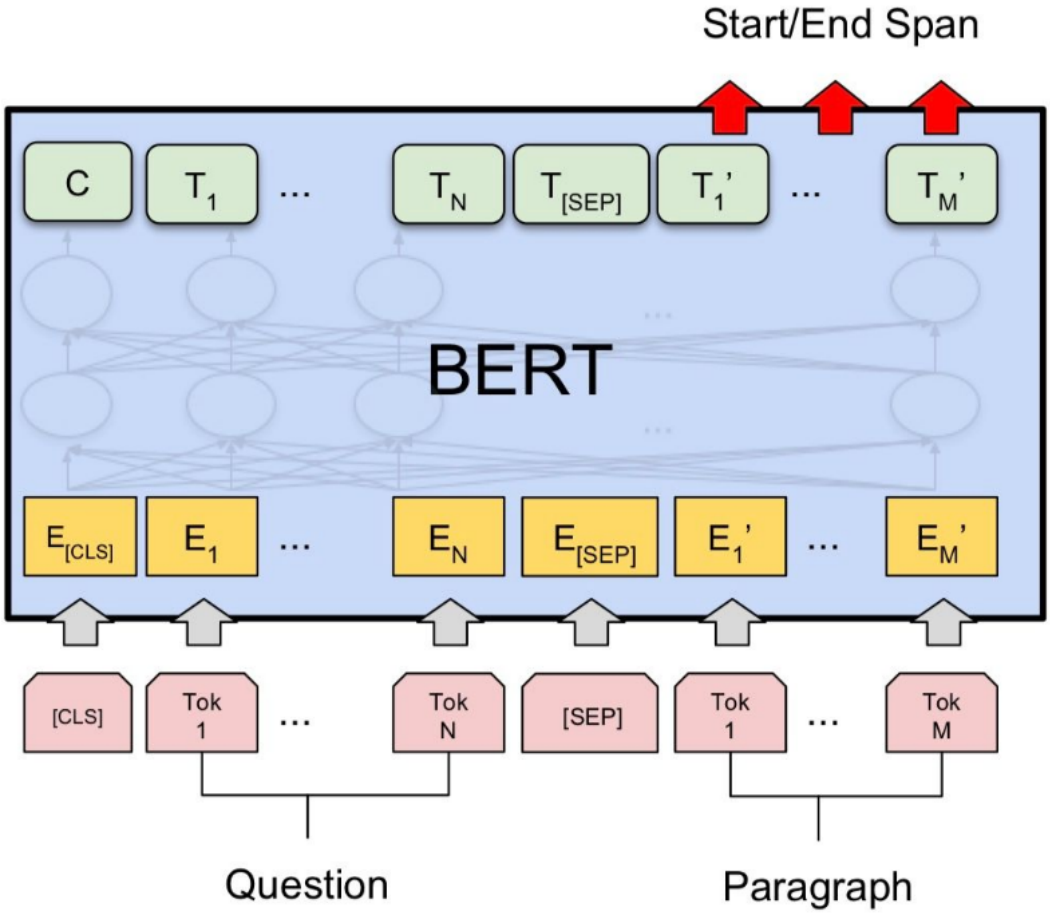

Figure 2.15: Illustrion of BERT fine-tuning QA task

Model	Parameters	Layers	Hidden	Embedding	Parameter-sharing
$BERT_{base}$	108M	12	768	768	False
$BERT_{large}$	334M	24	1024	1024	False
$ALBERT_{base}$	12M	12	768	128	True
$ALBERT_{large}$	18M	24	1024	128	True
$ALBERT_{base}$	60M	24	2048	128	True
$ALBERT_{base}$	235M	12	4096	128	True

Table 2.1: Comparison between BERT and ALBERT

running time, and the number of parameters needs to train. One of the models with many training parameters is GPT-2 [22] with more than 1.5 billion parameters. In 2020, OpenAI release GPT-3 [23], the currently hugest model with more than 175 billion parameters need to train. With a large number of parameter, a lot of GPUs or TPUs can not tackle this kind of models like this because of memory limitation. To handle this problem, Google releases A Lite BERT (ALBERT) [24], an improved model of BERT, which can train fewer parameters without reducing its performance compare to BERT. We will discuss the main cores of ALBERT make it special as following.

Factorized Embedding Parameterization

Some improvement models like XLNet [25] or RoBERTa [26] still use WordPiece as word embedding for their models. We assume that WordPiece has embedding size equal to E, hidden layer size is H (E = H in BERT), and the vocabulary size of WordPiece is 30,000. If we want to train the model deeper, we have to increase the H size and because E = H, so the size of embedding also increases, equivalent to 30,000E.

The main ideal of ALBERT is to reduce the parameters need to train. Instead of setting E = H, ALBERT has constant E = 128 (as Table 2.1 [24]), the number of parameters will be significant reduced by this method. For example, we assume that $H_B = 768$ $E_B = 768$ stand for hidden and embedding size for BERT and $H_A = 768$ $E_A = 128$ stand for hidden and embedding size for ALBERT. We can have a size comparison of there parameter as follow:

$$P_B = 30,000 H_B E_B = 23040000 \tag{2.13}$$

$$P_A = 30,000 E_A + E_A H_A = 3938304 \tag{2.14}$$

As we can see from the above example, BERT has more than 19101696 parameter need to train compare to ALBERT.

Cross-layer Parameter Sharing

There are several work relate to parameter sharing such as Universal Transfomer [27] or Deep Equilibrium Models (DQE) [28] for transformer architecture. The main ideal is to share the parameter through specify layer such as feed forward network or attention layer parameters. In ALBERT, they share all the parameter in all layers.

With two central cores, the number of parameters reduces almost 18 times compared to BERT.

2.2.7 RoBERTa

RoBERTa [26] is the transformer architecture base on BERT and having several improvements make it out-performance of BERT in some tasks.

Training Data

Comparing to BERT which only train on BOOKCORPUS [29] dataset (16GB of text), RoBERTa train with 160GB of text in various of datasets such as:

- The same with BERT, **BOOKCORPUS** is also used to train RoBERTa.
- **CC-News** (76GB of text), which is obtained from CommonCrawl News [30] dataset, is a dataset with more than 60M articles.
- **OpenWebText** [31] with more than 38GB text which is crawled on Reddit [11].
- The last dataset is used to train RoBERTa is **STORIES** [32].

Dynamic Masking

As we mentioned above in section 3.2.1, BERT uses [MASK] token once when pre-processing data. The issue is when we train the model with a lot of epochs, there may be cases that have the same mask position in the same sentence.

To tackle this problem, RoBERTa duplicates training data by ten times and they expect that the [MASK] token will be in different position. This adjustment can help performance of RoBERTa increases by 2 % compare to BERT on SQuAD 2.0 [10].

[11] Reddit.com

Model	SQuAD 1.1/2.0	MNLI-m	SST-2	RACE
Our reimplementation (with NSP loss)				
SEGMENT-PAIR	90.4/78.7	84.0	92.9	64.2
SENTENCE-PAIR	88.7/76.2	82.9	92.1	63.0
Our reimplementation (without NSP loss)				
FULL-SENTENCES	90.4/79.1	84.7	92.5	64.8
DOC-SENTENCES	90.6/79.7	84.7	92.7	65.6
$BERT_{BASE}$	88.5/76.3	84.3	92.8	64.3
$XLNet_{BASE}$(K = 7)	-/81.3	85.8	92.7	66.1
$XLNet_{BASE}$(K = 6)	-/81.0	85.6	93.4	66.7

Table 2.2: Performance comparison between training approaches on SQuAD, MNLI-m, STT-2 and RACE dataset

Different Training Objective

There are several research concern about NSP from BERT where it is necessary [33] [25]. To explore this problem, RoBERTa is trained in different ways such as:

- **Segment pair and NSP**: The same way BERT, input of the model is a pair of processed tokens. Each element of pair can have more than one sentence.
- **Sentence pair and NSP**: Instead of multiple sentences like Segment pair type, the input is just exactly one sentence for each element of pair.
- **Full sentence without NSP:** The input is not a pair, it is a list of sentences which are randomly picked or consecutive sentences from different articles.
- **Doc sentence without NSP**: The same way with Full sentence type but sentences are in the same article.

After obtaining result from these setting above as Table 2.2, they conclude that the input with multiple sentences is better and NSP loss is not necessary in for downstream tasks.

Chapter 3

Methodology

In this chapter, we introduce FitQA, the dataset we build for heath domain, the recap and new era of embedding, several state-of-the-art architectures base on Transformer and Curriculum learning, a training strategy for DL architectures, to enhance the performance of these architectures. We show the illustration of the whole system as Figure 3.1.

3.1 FitQA

FitQA is a QA dataset for health domain, and it contains all most 700 question-answer pairs. The answer for each question may be an entity, span, sentence or sentences in the context. We built FitQA by these steps below:

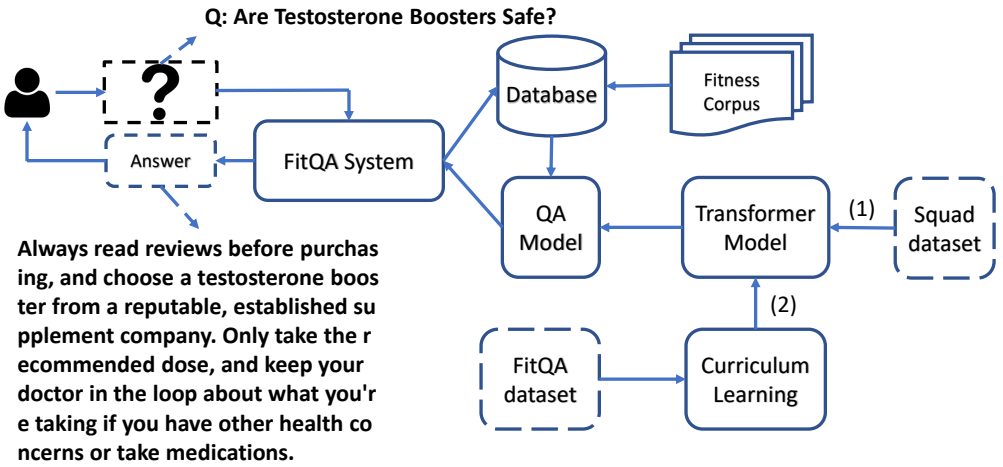

Figure 3.1: Illustration of FitQA System

- **Article and Question Collection:** First, we search for websites which contain various articles relate to nutrition and training. After taking time for searching, we figure out that bodybuilding [1] and t-nation [2] website contain a huge number of articles related to these topics. We crawl all the articles from this site in February 2020. In the other hand, t-nation owns a forums [3] for everyone to discuss a health topic, and it has a thousand question from different users. We crawled about 10,000 questions from this forums

- **Articles Selection:** From two sets of articles and questions, we apply elastic search [4] to look for what is the most articles that users usually concern and look for information. As a result, we can obtain about 200 articles from a set of 10,000 questions.

- **Context Creation:** Some articles have really long content and cover several different contexts. We decide to split the article by its section, and one section is on context.

- **Question and Answer Curation:** FitQA is built for health. Then

[1]https://www.bodybuilding.com/en-JP/index
[2]https://www.t-nation.com/
[3]https://forums.t-nation.com/
[4]https://www.elastic.co/

# Words	Proportion	Example
1-10	17.7%	**Question**: What is the main ingredient in most fat burners? **Answer**: Caffein
11-20	22.7%	**Question:** Does caffeine really stimulate fat loss? **Answer:** These work, and they do increase energy expenditure which leads to fat loss.
21-30	12%	**Question:** How can i control hunger? **Answer:** I drink sparkling water throughout the day. Not only does it keep you hydrated, but it also reduces hunger and sugar cravings.
31-40	10%	**Question:** Is pepper good for heart? **Answer:** The November 2013 issue of Cell Biochemistry and Biophysics reported that black pepper has beneficial effects on blood pressure. It also reduces inflammation, which is a big factor in heart disease.
41-50	12.8%	**Question:** Can I build muscle on keto diet? **Answer:** It's possible, but not easy, likely, or ideal. I could walk from my house (in Georgia) all the way to San Diego, but it would be much faster if I took a flight. Hypertrophy while on keto is kind of like that.
>51	24.8%	**Question:** Can carrot reduce the risk of cancer? **Answer:** They found that the more often men ate carrots, and the greater the amount of carrots eaten, the less likely they were to get prostate cancer. They even came up with some definitive numbers: For every 10 grams of carrots consumed each day, men reduced their risk of developing prostate cancer by 5%.

Table 3.1: Statistics Length and QA pairs From FitQA

we want the information must be detailed in answer. We realize that questions raise by users from t-nation forum are hard for DL models to extract the answer from the context. These kind of questions are not clear, it needs to be deduced or world knowledge to answer. So we built the dataset by raising the question around the context that the answer can be an entity, a short span in context, a sentence or maybe sentences.

FitQA almost has the same format with SQuAD 2.0 dataset, the different and challenges that make FitQA different from SQuAD 2.0 are as below:

- The average length of context in FitQA is double of that in SQuAD 2.0.
- The average length of answers in FitQA is ten times longer than SQuAD 2.0.

The state-of-the-art models are overwhelming human performance, and the dataset must be harder and more challenges to be able to achieve some important achievements in machine comprehension task. There are some participants in this case, NewsQA [11] have more challenges than SQuAD [4] by having less word matching examples(7.1%), more paraphrasing example(7.3%) and more synthesis and inference examples(13.4%). On the other hand, TriviaQA has 69% questions that have different syntactic structure, and 41% of them have lexical different. Moreover, the information needed to answer the question is scattered over multiple sentences. Base on these ideals, we increase the complexity of FitQA by the diversity in answer length, we show statistics length and several examples from FitQA in Table 3.1. To test the performance of state-of-the-art models, we create the test set by picking 100 examples with varied length, and the rest is for the training set.

3.2 Curriculum Learning

We examine some previous work and propose a useful method for handling the long article and also answer. Curriculum learning [3] is a learning strategy in machine learning, we let the DL model learn with easy examples first and then gradually handles harder cases. Several works have shown that this problem can be overcome by using this learning strategy. As a result of Cao Liu [34] in his natural answer generation task, curriculum learning can increase his model performance by 6.8% and 8.7% in the accuracy for easy and hard questions.

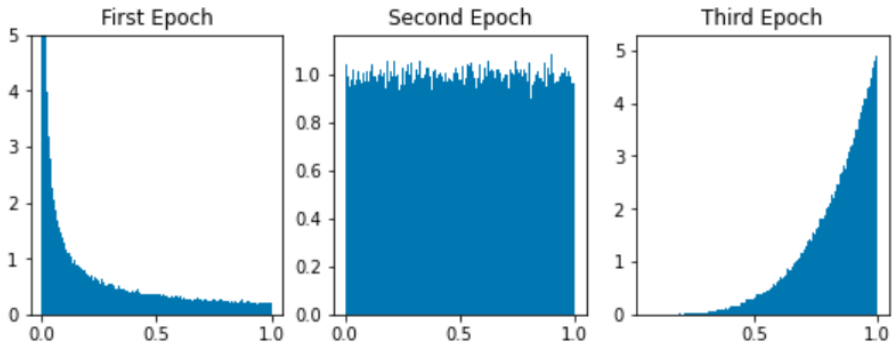

Figure 3.2: Probability of picking example through epoch

In our question-answering task, we defined the complexity by the length of the answer. We assume that an example containing a short answer is easy, and an example having a long answer is difficult. We want models to learn from easy to difficult sample and also from the difficult to easy sample. We give every single example a score equal to its answer length. After that, we sort the list of examples in ascending order by scores to pop the sample by its index easily. The formula and pseudo-code of curriculum learning(Algorithm 1) to calculate the probability of picking an example $Index$ from n examples as below:

$$index_{ij} = \lfloor n x_{ij}^{t_i} \rfloor \tag{3.1}$$

$$x_{ij} \sim U[0, 1) \tag{3.2}$$

where t_i is the temperature of epoch i^{th} obtained by:

$$t_i = \gamma^{\alpha_i} \tag{3.3}$$

$$\alpha_i = \frac{2(1-i)}{N_{epochs} - 1} + 1 \tag{3.4}$$

where γ is a temperature base, N_{epochs} is the total number of training epoch. The temperature base γ reflects how high the probability of picking a long or short example through the epoch.

We illustrate the probability of picking example curriculum learning as Figure 3.2. In the first epoch, we can easily see that the possibility of picking the examples with the short answer is high and it is pretty low with cases have a long answer. In the second epoch, the probability of picking example is the uniform distribution. In the last epoch, the probability of picking the examples with the long answer is hugely higher than the short answer examples.

Result: Sample by length
Input:
The list of n_{sample} samples is sorted in ascending order:SL;
γ is temperature base ;
N_{epochs} is total number of epoch;
Output:;
The list of n_{sample} samples is arranged by curriculum learning:OL;
Function: curriculum_learning(SL,γ,N_{epochs}):
OL $\leftarrow$ {}
for *i = 1 to* N_{epochs} **do**
    $temp_{Li}$ = SL
    $\alpha_i = \frac{2(1-i)}{N_{epochs}-1} + 1$
    $t_i = \gamma^{\alpha_i}$
    for *j = 0 to* n_{sample} **do**
        x_{ij} = random(0,1)
        $index_{ij}$ = length($temp_{Li}$)
        OL.push($temp_{Li}[index_{ij}]$) $temp_{Li}$.pop($index_{ij}$)
    end
end

Algorithm 1: Curriculum learning pseudo-code.

In the next chapter, we will describe in detailed how we apply curriculum learning to these state-of-the-art transformer model, assessments and analysis of results.

Chapter 4

Experiments and Results

4.1 Experiment Settings

From SQuAD and NewsQA leaderboard, there are some approaches perform better performance, but all of them are built base on the pre-trained model. To test FitQA for the machine comprehension task, we compare the performance of four common pre-trained deep learning models: bidirectional encoder representations from Transformers (BERT), two versions of a Lite BERT (ALBERT), and RoBERTa. We describe details of all the pre-trained models as below:

1. bert-base-uncased : 12 layers, 768 hidden, 12 heads, 110M parameters, and trained on low-cased English text.

2. albert-base-v1: 12 layers, 768 hidden, 128 embedding, 12 heads, 11M parameter.

3. albert-base-v2: 12 layers, 4096-hidden, 128 embedding, 64-heads, 223M parameters.

4. roberta-base:12-layers, 768 hidden, 12-heads, 125M parameters RoBERTa using the BERT-base architecture.

We conduct experiments on SQuAD, FitQA. Performance on these datasets is measured by exact match(EM) and per answer token-based F1 score, which was published by Rajpurkar [4]. The detailed settings are described as below:

1. **FitQA uniform:** Using four pre-trained models to test the performance on FitQA with the probability of picking an example is the uniform distribution.

Model	FitQA uniform		SQuAD 2.0+FitQA uniform	
	EM(%)	***F1(%)***	***EM(%)***	***F1(%)***
albert-base v1	11.1	34.9	16.9	42.5
albert-base v2	12.1	37.8	15.3	42.5
bert-base-uncase	6.9	31.1	17.8	45.0
roberta-base	5.6	25.1	14.1	42.7

Table 4.1: F1 and Exact Match(EM) scores on FitQA with MAL=30 and without curriculum learning

Model	SQuAD 2.0+FitQA $\gamma = 2$		SQuAD 2.0+FitQA $\gamma = 3$		SQuAD 2.0+FitQA $\gamma = 5$	
	EM(%)	***F1(%)***	***EM(%)***	***F1(%)***	***EM(%)***	***F1(%)***
albert-base v1	15.3	43.1	16.7	44.0	15.0	42.1
albert-base v2	14.3	43.5	14.4	43.2	17.0	**45.6**
bert-base-uncase	17.8	45.0	17.0	44.2	16.7	**46.1**
roberta-base	14.3	**45.5**	14.7	42.3	14.4	43.8

Table 4.2: F1 and Exact Match(EM) scores on FitQA with MAL=30

2. **SQuAD 2.0 + FitQA uniform:**: We use four pre-trained models fine-tune SQuAD 2.0 first, then fine-tune FitQA with the probability of picking example is the uniform distribution.

3. **SQuAD 2.0 + FitQA γ=2,3,5:** We use 4 pre-trained models fine-tune SQuAD 2.0 with uniform distribution first, then sequentially applying curriculum learning with γ=2,3,5 on FitQA training set. Finally fine-tuning on this processed training set and test the performance.

We repeat these experiments two time with *maximum answer length (MAL)* sequentially equal to 30 and 60. The illustration of the whole process of applying curriculum learning to FitQA to obtain processed data which use to train transformer architectures.

4.2 Experimental Results

According to information from SQuAD 2.0 leaderboard, the best performance that Albert single can archive is 88.592% EM score and 91.286% F1 score. However, in section 3.1, we showed that the length of some examples in FitQA are extremely long and diverse. This is the reason makes four state-of-the-art models can not work well on FitQA. The result in Table 4.1, albert-base-v2

Model	FitQA uniform		SQuAD 2.0+FitQA uniform	
	EM(%)	***F1(%)***	***EM(%)***	***F1(%)***
albert-base v1	8.8	38.6	18.3	51.3
albert-base v2	17.0	47.3	21.2	54.2
bert-base-uncase	6.9	31.1	17.0	52.6
roberta-base	4.9	30.9	19.0	52.7

Table 4.3: F1 and Exact Match(EM) scores on FitQA with MAL=60

Model	SQuAD 2.0+FitQA $\gamma = 2$		SQuAD 2.0+FitQA $\gamma = 3$		SQuAD 2.0+FitQA $\gamma = 5$	
	EM(%)	***F1(%)***	***EM(%)***	***F1(%)***	***EM(%)***	***F1(%)***
albert-base v1	15.7	51.3	16.7	50.8	17.0	51.4
albert-base v2	16.7	51.5	19.3	52.0	18.3	53.0
bert-base-uncase	15.4	51.4	20.33	53.9	17.7	52.1
roberta-base	18.0	53.3	20.3	53.7	19.0	**55.3**

Table 4.4: F1 and Exact Match(EM) scores on FitQA with MAL=60

has the best result but only 37.8% F1 score and 12.1% on EM. SQuAD 2.0 is the most similar dataset to FitQA. To maximize the performance, we firstly train all models on SQuAD 2.0 and fine-tune FitQA. After training on SQuAD and fine-tune FitQA the performance increase 5.68% on EM and 8.9%F1 score on average. Next, we mute the shuffling feature, then apply curriculum learning to the training set. We start with temperature base $\gamma = 2$ and MAL=30. As the results in Table 4.1, curriculum learning made average F1 score from 43.2% to 44.3%. Especially, it can increase the performance of roberta-base by 2.8%. With $\gamma = 3$, there is no significant improvement.

We increase γ base γ to 5, and we can get the best results with 46.1% and 45.6% on bert-base-uncase and albert-base-v2.

Next, we want the model to face the harder challenge by increasing MAL to 60. As we can see in Table 4.2, The models without fine-tune SQuAD 2.0 has performance increases by 4.75% on F1 score, and 0.5% on EM score compare to MAL = 30. After using a model which fine-tuned SQuAD, we can see the significant increment of performance of these models on FitQA. The performance increases 15.7% on F1 score and 9.5% on EM score. Next, we apply curriculum learning on these models. With γ equal to 2, there is no improvement. As the results in Table 4.2, we can see roberta-base with curriculum learning ($\gamma = 5$) has the highest performance model with 55.3% on F1 score. In the other hand, bert-base-uncased with curriculum learning

Length	bert-base-uncase $\gamma = 5$		albert-base-v2 $\gamma = 5$		bert-base-uncase uniform	
	EM(%)	***F1(%)***	***EM(%)***	***F1(%)***	***EM(%)***	***F1(%)***
1-10	41.7	62.4	47.9	58.5	50	63.4
11-20	37.0	**59.1**	40.7	**64.0**	38.8	57.0
21-30	20.5	**52.9**	10.3	46.3	20.5	50.9
31-40	0.0	43.4	0.0	**48.3**	0.0	44.8
41-50	0.0	**34.6**	0.0	**35.0**	0.0	30.4
51-60	0.0	**37.7**	0.0	**40.9**	0.0	34.2
61-70	0.0	34.7	0.0	32.4	0.0	37.7
>70	0.0	23.8	0.0	21.4	0.0	24.3

Table 4.5: F1 and Exact Match(EM) scores on best settings base on length with MAL=30

Length	albert-base-v2 uniform		roberta-base $\gamma = 5$		bert-base-uncase γ=3	
	EM(%)	***F1(%)***	***EM(%)***	***F1(%)***	***EM(%)***	***F1(%)***
1-10	35.4	49.4	47.9	**67.9**	29.1	47.5
11-20	37.0	63.9	25.9	62.2	44.4	59.9
21-30	46.2	67.2	35.9	61.5	30.8	65.0
31-40	30.3	61.2	15.1	53.4	21.1	57.5
41-50	0.0	44.5	0.0	**46.1**	0.0	**46.9**
51-60	0.0	56.6	0.0	**56.9**	0.0	**56.8**
61-70	0.0	47.2	0.0	41.8	0.0	**50.0**
>70	0.0	35.8	0.0	36.3	0.0	**41.6**

Table 4.6: F1 and Exact Match(EM) scores on best settings base on length with MAL=60

($\lambda = 3$) and albert-base-v2 without curriculum learning (fine-tune SQuAD version) are also have excellent performance. So we will take three models to analyze their results.

4.3 Result Analysis

With MAL=30, bert-base-uncased and albert-base v2 with temperature base $\gamma = 5$ seems to be the best for FitQA, so we analyze the results and compare them to bert-base-uncase without curriculum learning. As a result of Table 4.5, we show the accuracy of the answer group was mentioned in Table 3.1.

By comparing the results from these settings, we expect to determine that curriculum learning is useful for extracting more text or capture more related information to answer the question. In lengths from 0 to 10 words, we can see there is no significant change between all settings. Starting from 11 words, we can see that these results go beyond the uniform distribution setting. With temperature base $\gamma = 5$ from Table 4.5, we can see bert-base-uncase works well in lengths from 11 to 60 words, and the performance in this range increase 2.08% on average compare to uniform distribution bert-base-uncase.

Albert-base-v2 can also perform well in this range with 3.44% increase in total. TABLE 4.6 summarizes overall statistics of 3 best settings on FitQA with MAL=60. It is worth discussing these interesting facts revealed by the results of bert-base-uncased . The test in range 41 to more than 70 words found differences from bert-base-uncase compare to albert-base-v2 with 2.8% improvement on F1 score.

One limitation is found in these experiments. From TABLE 4.8, the most extended answer can be extracted is 50 words. It means for the examples have the answer more than 50 words, the EM score will be zero. Not only that, but it is also hard to extract long answer correctly from the context, and some answers are a subset of gold answers. This is the reason leads EM score to 0 in some evaluations. We show several examples to demonstrate for this limitation in Table 4.7. Models extract the answers is acceptable, but not enough in these cases.

As the results are shown in Table 4.5 and Table 4.6, we have succeeded in improving the length that the model can extract by applying curriculum learning on the training set. This success leads to an increase in F1 score. The problem is all the state-of-the-art models perform poorly under long answer form dataset. The best result that these models can get is just 21.2% on EM and 55.3% on F1 score.

Question	Gold Answer	Predicted Answer
Are Testosterone Boosters Safe?	Always read reviews before purchasing, and choose a testosterone booster from a reputable, established supplement company. Only take the recommended dose, and keep your doctor in the loop about what you're taking if you have other health concerns or take medications.	Always read reviews before purchasing, and choose a testosterone booster from a reputable, established supplement company.
what is the difference between brown fat and white fat?	Both types store energy, but white fat cells each contain only one droplet of fat, while brown fat contains lots of tiny droplets of fat. Brown fat also contains tons of the brownish cellular organelles known as mitochondria, which use the droplets of fat to create energy and, as a byproduct of creating energy, heat.	Both types store energy, but white fat cells each contain only one droplet of fat, while brown fat contains lots of tiny droplets of fat.

Table 4.7: 2 Examples for the most limitation of FitQA

Model	Total Tokens	Longest Answer	Shortest Answer
bert-base-uncase $\gamma = 5$,MAL=30	1462	28	1
albert-base-v2 $\gamma = 5$,MAL=30	1384	28	1
bert-base-uncase uniform,MAL=30	1271	28	1
albert-base-v2 uniform,MAL=60	2255	46	1
roberta-base $\gamma = 5$,MAL=60	1980	42	1
bert-base-uncase $\gamma = 3$,MAL=60	2330	50	1

Table 4.8: Total number of tokens, longest and shortest answer that models can extract from 100 examples of test set

Chapter 5

Conclusion

In our research, we provide the experiments on complex structure dataset FitQA with different state-of-the-are Transformer models. The results prove that curriculum learning can slightly improve performance of Transfomer model. Our experiments indicate that even state-of-the-art models which out-performance human performance on open domain datasets such as SQuAD, they still perform poorly on difficult questions like why, how and yes-no questions. For future work, we want to improve the quality and also quantity of FitQA, research for new methods to significantly improve the performance of DL models on FitQA and apply this QA system on practical application.

Bibliography

[1] B. F. Green Jr, A. K. Wolf, C. Chomsky, and K. Laughery, "Baseball: an automatic question-answerer," in *Papers presented at the May 9-11, 1961, western joint IRE-AIEE-ACM computer conference*, 1961, pp. 219–224.

[2] W. A. Woods and W. WA, "Lunar rocks in natural english: Explorations in natural language question answering." 1977.

[3] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 41–48.

[4] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100,000+ questions for machine comprehension of text," *arXiv preprint arXiv:1606.05250*, 2016.

[5] J.-H. Oh, K. Torisawa, C. Hashimoto, T. Kawada, S. De Saeger, Y. Wang *et al.*, "Why question answering using sentiment analysis and word classes," in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 2012, pp. 368–378.

[6] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[7] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[9] S. Gaikwad, D. Morina, R. Nistala, M. Agarwal, A. Cossette, R. Bhanu, S. Savage, V. Narwal, K. Rajpal, J. Regino *et al.*, "Daemo: A self-governed crowdsourcing marketplace," in *Adjunct proceedings of the 28th annual ACM symposium on user interface software & technology*, 2015, pp. 101–102.

[10] P. Rajpurkar, R. Jia, and P. Liang, "Know what you don't know: Unanswerable questions for squad," *arXiv preprint arXiv:1806.03822*, 2018.

[11] A. Trischler, T. Wang, X. Yuan, J. Harris, A. Sordoni, P. Bachman, and K. Suleman, "Newsqa: A machine comprehension dataset," *arXiv preprint arXiv:1611.09830*, 2016.

[12] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," in *Advances in neural information processing systems*, 2015, pp. 1693–1701.

[13] P. Ferragina and U. Scaiella, "Tagme: on-the-fly annotation of short text fragments (by wikipedia entities)," in *Proceedings of the 19th ACM international conference on Information and knowledge management*, 2010, pp. 1625–1628.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[15] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[17] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[18] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[20] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[21] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[22] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.

[23] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *arXiv preprint arXiv:2005.14165*, 2020.

[24] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," *arXiv preprint arXiv:1909.11942*, 2019.

[25] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," in *Advances in neural information processing systems*, 2019, pp. 5753–5763.

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

[27] M. Dehghani, S. Gouws, O. Vinyals, J. Uszkoreit, and Ł. Kaiser, "Universal transformers," *arXiv preprint arXiv:1807.03819*, 2018.

[28] S. Bai, J. Z. Kolter, and V. Koltun, "Deep equilibrium models," in *Advances in Neural Information Processing Systems*, 2019, pp. 690–701.

[29] Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler, "Aligning books and movies: Towards story-like visual explanations by watching movies and reading books," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 19–27.

[30] S. Nagel, "Cc-news," 2016.

[31] A. Gokaslan and V. Cohen, "Openwebtext corpus," 2019.

[32] T. H. Trinh and Q. V. Le, "A simple method for commonsense reasoning," *arXiv preprint arXiv:1806.02847*, 2018.

[33] G. Lample and A. Conneau, "Cross-lingual language model pretraining," *arXiv preprint arXiv:1901.07291*, 2019.

[34] C. Liu, S. He, K. Liu, J. Zhao *et al.*, "Curriculum learning for natural answer generation," 2018.