



## BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

- ◆ Jacob Devlin  
Ming-Wei Chang  
Kenton Lee  
Kristina Toutanova  
Google AI Language
- ◆ PDF 링크  
: <https://arxiv.org/pdf/1810.04805.pdf>

정봉수



연구 배경

---

downstream task method : 1. feature-based 2. fine-tuning  
현재 기술들은 pretrained-language model 의 힘을 제약시킨다.  
특히나 fine-tuning에서  
단방향 문제를 MLM 문제를 사용해서 해결했다.

# Introduction

BERT Introduction 에서 논문의 저자는 Open GPT , ELMo 보다 자신들의 모델이 Bidirectional Encoding 을 잘한다고 설명 하고 있습니다.

그 방법으로 소개 된 것이 MLM(Masked Language Model) 과 NSP(Next Sentence Prediction) 입니다.

In this paper, we improve the fine-tuning based approaches by proposing BERT: **B**idirectional **E**ncoder **R**epresentations from **T**ransformers. BERT alleviates the previously mentioned unidirectionality constraint by using a “masked language model” (MLM) pre-training objective, inspired by the Cloze task (Taylor, 1953). The masked language model randomly masks some of the tokens from the input, and the objective is to predict the original vocabulary id of the masked

word based only on its context. Unlike left-to-right language model pre-training, the MLM objective enables the representation to fuse the left and the right context, which allows us to pre-train a deep bidirectional Transformer. In addition to the masked language model, we also use a “next sentence prediction” task that jointly pre-trains text-pair representations. The contributions of our paper are as follows:

# MODEL

Bert Architecture 을 도식화 한 것으로 크게 BERT 모델은

## 1. Pre-training

1-1 MLM

1-2 NSP

## 2. Fine-Tuning 나뉘어집니다.

BERT 모델은 TASK 별로 새로운 Architecture을 생성하지 않고 단일 모델로 여러개의 TASK를 처리 할 수 있고 11개 분야에서 state-of-the-art 를 달성 했습니다.

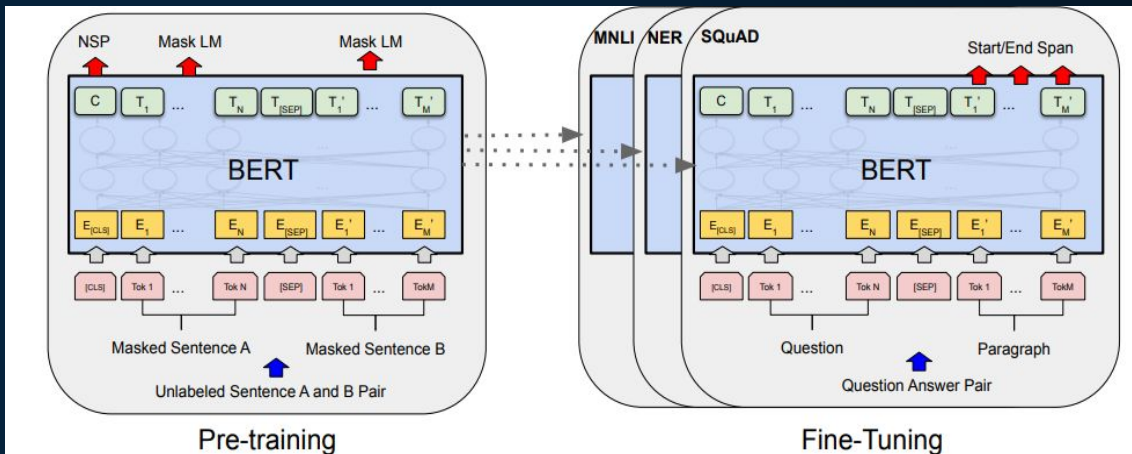


Figure 1: Overall pre-training and fine-tuning procedures for BERT. Apart from output layers, the same architectures are used in both pre-training and fine-tuning. The same pre-trained model parameters are used to initialize models for different down-stream tasks. During fine-tuning, all parameters are fine-tuned. [CLS] is a special symbol added in front of every input example, and [SEP] is a special separator token (e.g. separating questions/answers).

# BERT

---

## 1. Pre-training

pre-training 동안에는 unlabeled  
한 데이터를 학습합니다.

## 2. Fine-Tuning

pre-training 으로 학습되어진  
파라미터들을 초기 parameter  
값으로 initialized 한다.

## 3 BERT

We introduce BERT and its detailed implementation in this section. There are two steps in our framework: *pre-training* and *fine-tuning*. During pre-training, the model is trained on unlabeled data over different pre-training tasks. For fine-tuning, the BERT model is first initialized with the pre-trained parameters, and all of the parameters are fine-tuned using labeled data from the downstream tasks. Each downstream task has separate fine-tuned models, even though they are initialized with the same pre-trained parameters. The question-answering example in Figure 1 will serve as a running example for this section.

A distinctive feature of BERT is its unified architecture across different tasks. There is mini-

# BERT MODEL ARCHITECTURE

BERT 모델은 기본적으로 Transformer의 Encoder 부분을 여러 단계 쌓은 모형이다.

BERT(base) 모델 :  $L = 12, H=768, A=12$   
BERT(large) 모델 :  $L = 24, H=1024$   
,  $A=16$  으로 이루어져 있다.

$L$  은 Transformer blocks 갯수  
 $H$  는 hidden size  
 $A$  는 Attention head 의 갯수이다.

**Model Architecture** BERT's model architecture is a multi-layer bidirectional Transformer encoder based on the original implementation described in Vaswani et al. (2017) and released in the `tensor2tensor` library.<sup>1</sup> Because the use of Transformers has become common and our implementation is almost identical to the original, we will omit an exhaustive background description of the model architecture and refer readers to Vaswani et al. (2017) as well as excellent guides such as "The Annotated Transformer."<sup>2</sup>

In this work, we denote the number of layers (i.e., Transformer blocks) as  $L$ , the hidden size as  $H$ , and the number of self-attention heads as  $A$ .<sup>3</sup> We primarily report results on two model sizes: **BERT<sub>BASE</sub>** ( $L=12, H=768, A=12$ , Total Parameters=110M) and **BERT<sub>LARGE</sub>** ( $L=24, H=1024, A=16$ , Total Parameters=340M).

BERT<sub>BASE</sub> was chosen to have the same model size as OpenAI GPT for comparison purposes. Critically, however, the BERT Transformer uses bidirectional self-attention, while the GPT Transformer uses constrained self-attention where every token can only attend to context to its left.<sup>4</sup>



# BERT INPUT REPRESENTATION

BERT의 Input은 단일 sentence가 들어갈 수도있고, (sentence A , sentence B)가 하나로 묶여서 들어 갈 수 있다.

Bert모델은 WordPiece 라는 단어 분리 알고리즘을 사용해서 30,000 영어 토큰 사전을 생성했다. (tokenizer도 위의 임베딩 모델을 사용한다.

sentence의 시작에는 항상 [CLS] 라는 토큰이 붙고 , sentence A sentence B를 하나로 묶어서 Input에 넣을때는 항상 ([CLS] sentence A [SEP] sentence B [SEP]) 이렇게 넣어준다. ex)2 set

([CLS] sentence A [SEP]) ex) 1 set

**Input/Output Representations** To make BERT handle a variety of down-stream tasks, our input representation is able to unambiguously represent both a single sentence and a pair of sentences (e.g.,  $\langle \text{Question, Answer} \rangle$ ) in one token sequence. Throughout this work, a “sentence” can be an arbitrary span of contiguous text, rather than an actual linguistic sentence. A “sequence” refers to the input token sequence to BERT, which may be a single sentence or two sentences packed together.

We use WordPiece embeddings (Wu et al., 2016) with a 30,000 token vocabulary. The first token of every sequence is always a special classification token ([CLS]). The final hidden state corresponding to this token is used as the aggregate sequence representation for classification tasks. Sentence pairs are packed together into a single sequence. We differentiate the sentences in two ways. First, we separate them with a special token ([SEP]). Second, we add a learned embedding to every token indicating whether it belongs to sentence A or sentence B. As shown in Figure 1, we denote input embedding as  $E$ , the final hidden vector of the special [CLS] token as  $C \in \mathbb{R}^H$ , and the final hidden vector for the  $i^{\text{th}}$  input token as  $T_i \in \mathbb{R}^H$ .

For a given token, its input representation is constructed by summing the corresponding token, segment, and position embeddings. A visualization of this construction can be seen in Figure 2.

# BERT PRE-TRAINING(MLM)

deep bidirectional representation 을  
얻기 위해서 MLM을 사용하지만  
fine-tuning에서는 [MASK] 토큰이  
등장 하지 않기 때문에 [MASK] 토큰을  
사용한 학습이 좋은 결과를 만들지  
못할 수도 있다 그것을 막기 위해서

전체 데이터의 15 프로를 [MASK]  
토큰을  
사용하고 거기에 80프로는 [MASK]토큰  
10프로는 랜덤 토큰, 나머지 10프로는  
원래의 단어를 그대로 둔다.

MLM에서는 문장에서 MASK로 여겨지  
는 15프로의 토큰을 맞추는 것을  
목표로 한다.

In order to train a deep bidirectional representation, we simply mask some percentage of the input tokens at random, and then predict those masked tokens. We refer to this procedure as a “masked LM” (MLM), although it is often referred to as a *Cloze* task in the literature (Taylor, 1953). In this case, the final hidden vectors corresponding to the mask tokens are fed into an output softmax over the vocabulary, as in a standard LM. In all of our experiments, we mask 15% of all WordPiece tokens in each sequence at random. In contrast to denoising auto-encoders (Vincent et al., 2008), we only predict the masked words rather than reconstructing the entire input.

Although this allows us to obtain a bidirectional pre-trained model, a downside is that we are creating a mismatch between pre-training and fine-tuning, since the [MASK] token does not appear during fine-tuning. To mitigate this, we do not always replace “masked” words with the actual [MASK] token. The training data generator chooses 15% of the token positions at random for prediction. If the  $i$ -th token is chosen, we replace the  $i$ -th token with (1) the [MASK] token 80% of the time (2) a random token 10% of the time (3) the unchanged  $i$ -th token 10% of the time. Then,  $T_i$  will be used to predict the original token with cross entropy loss. We compare variations of this procedure in Appendix C.2.



# BERT PRE-TRAINING(NSP)

MLM 으로 *bidirection representation*은 어느 정도 처리 했지고, 문장 간에 어떠한 관계가 있는지는 NSP(Next sentence Prediction)을 통해 획득하게 된다.

처음에 설명하기를 (sent A , sent B)와 같이 인풋을 설정 한다고 했다, 여기에 sent B 에 올 문장을 50프로는 실제 문장을 위치하고 , 나머지 50프로는 실제 문장이 아닌 랜덤한 문장을 배치 한다.

학습은 전체 문장이 이어진 문장이냐 아니면 다음에 올문장이 아니냐로 loss 를 얻어 낸다.

## Task #2: Next Sentence Prediction (NSP)

Many important downstream tasks such as Question Answering (QA) and Natural Language Inference (NLI) are based on understanding the *relationship* between two sentences, which is not directly captured by language modeling. In order to train a model that understands sentence relationships, we pre-train for a binarized *next sentence prediction* task that can be trivially generated from any monolingual corpus. Specifically, when choosing the sentences A and B for each pre-training example, 50% of the time B is the actual next sentence that follows A (labeled as `IsNext`), and 50% of the time it is a random sentence from the corpus (labeled as `NotNext`). As we show

# BERT INPUT Figure

최종 input =  
TokenEmbedding(input) +  
Segment Embedding(0 or 1) +  
Position Embedding()

저자는 문장 길이는 최대 512 토큰으로  
제한했다.

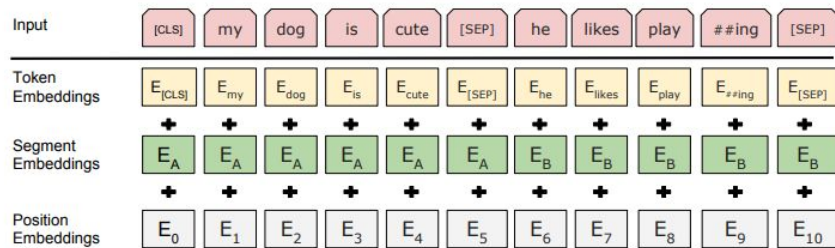


Figure 2: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings.

# OUTPUT

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	Average -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
BERT <sub>BASE</sub>	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
BERT <sub>LARGE</sub>	<b>86.7/85.9</b>	<b>72.1</b>	<b>92.7</b>	<b>94.9</b>	<b>60.5</b>	<b>86.5</b>	<b>89.3</b>	<b>70.1</b>	<b>82.1</b>

GLUE Test results

# OUTPUT

위쪽이 SQuAD 1.1 result

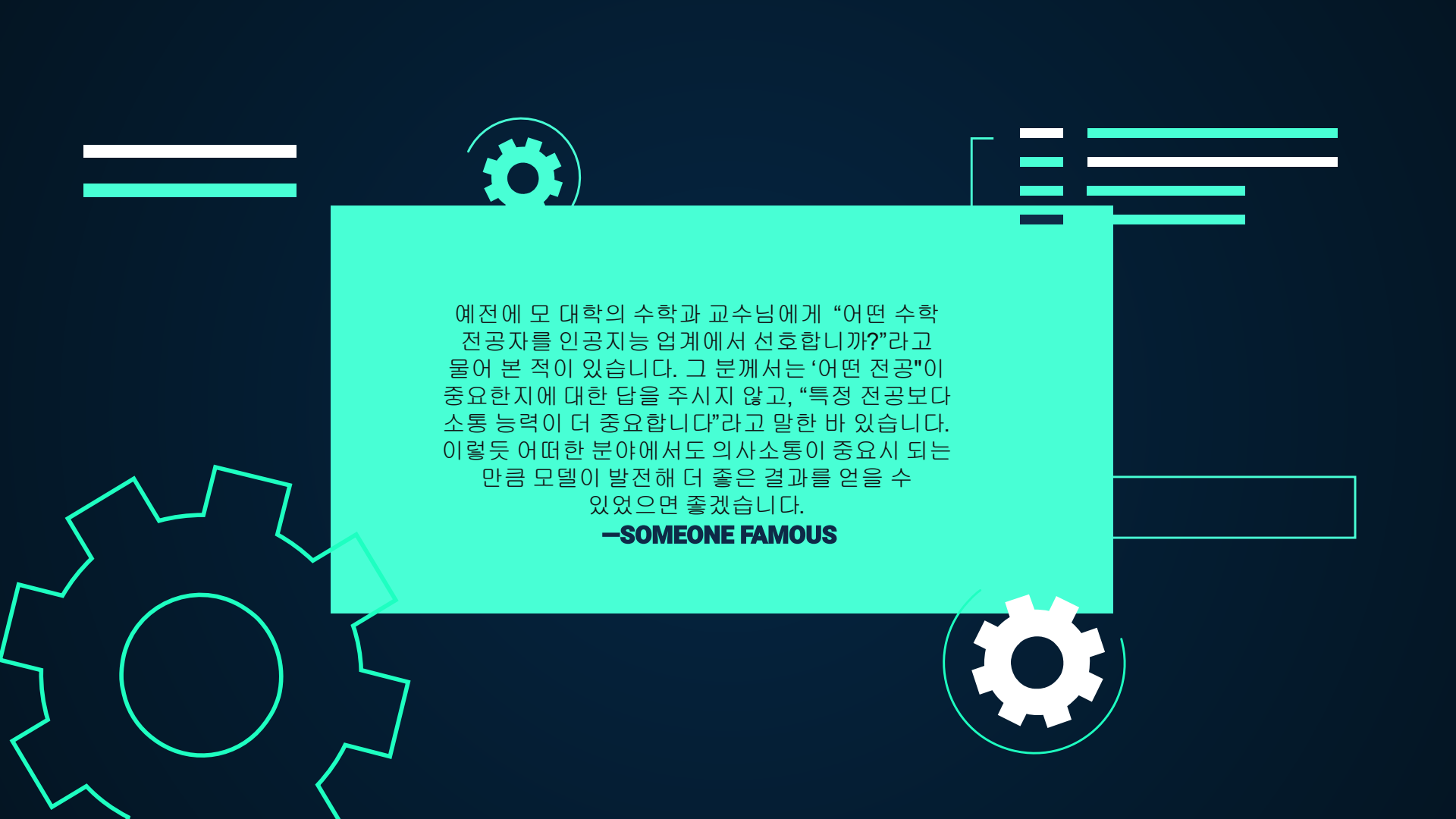
아랫쪽이 SQuAD 2.0 result

System	Dev		Test	
	EM	F1	EM	F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	-	-	82.3	91.2
#1 Ensemble - nlnet	-	-	86.0	91.7
#2 Ensemble - QANet	-	-	84.5	90.5
Published				
BiDAF+ELMo (Single)	-	85.6	-	85.8
R.M. Reader (Ensemble)	81.2	87.9	82.3	88.5
Ours				
BERT <sub>BASE</sub> (Single)	80.8	88.5	-	-
BERT <sub>LARGE</sub> (Single)	84.1	90.9	-	-
BERT <sub>LARGE</sub> (Ensemble)	85.8	91.8	-	-
BERT <sub>LARGE</sub> (Sgl.+TriviaQA)	<b>84.2</b>	<b>91.1</b>	<b>85.1</b>	<b>91.8</b>
BERT <sub>LARGE</sub> (Ens.+TriviaQA)	<b>86.2</b>	<b>92.2</b>	<b>87.4</b>	<b>93.2</b>

Table 2: SQuAD 1.1 results. The BERT ensemble is 7x systems which use different pre-training checkpoints and fine-tuning seeds.

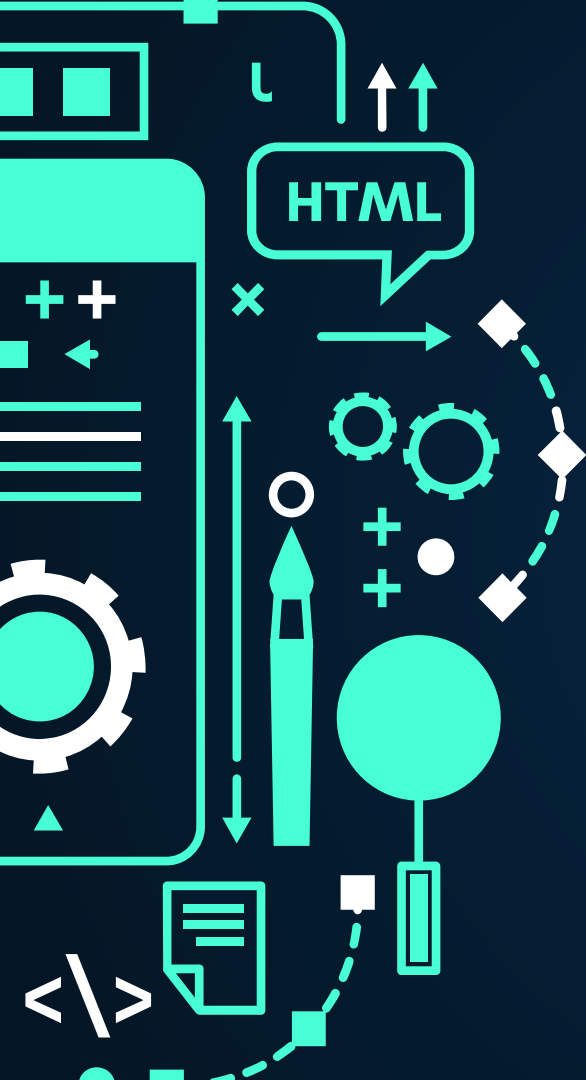
System	Dev		Test	
	EM	F1	EM	F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	86.3	89.0	86.9	89.5
#1 Single - MIR-MRC (F-Net)	-	-	74.8	78.0
#2 Single - nlnet	-	-	74.2	77.1
Published				
unet (Ensemble)	-	-	71.4	74.9
SLQA+ (Single)	-	-	71.4	74.4
Ours				
BERT <sub>LARGE</sub> (Single)	78.7	81.9	80.0	83.1





예전에 모 대학의 수학과 교수님에게 “어떤 수학 전공자를 인공지능 업계에서 선호합니까?”라고 물어 본 적이 있습니다. 그 분께서는 ‘어떤 전공’이 중요한지에 대한 답을 주시지 않고, “특정 전공보다 소통 능력이 더 중요합니다”라고 말한 바 있습니다. 이렇듯 어떠한 분야에서도 의사소통이 중요시 되는 만큼 모델이 발전해 더 좋은 결과를 얻을 수 있었으면 좋겠습니다.

**—SOMEONE FAMOUS**



# THANKS!

Does anyone have any question?

[hmy831004@google.com](mailto:hmy831004@google.com)  
010 2687 6629



# RESOURCES

- Effective approaches to attention-based neural machine translation  
(Minh-Thang Luong, Hieu Pham, Christopher D. Manning)
- D. Bahdanau, K. Cho, and  
Y. Bengio. 2015. Neural machine translation by  
jointly learning to align and translate. In ICLR.  
(Bahdanau et al.2015 )
- BLEU: a method for automatic evaluation of machine translation  
(Papineni, K.; Roukos, S.; Ward, T.; Zhu, W. J. (2002))
- Re-evaluating the Role of BLEU in Machine Translation Research  
(Callison-Burch, C., Osborne, M. and Koehn, P. (2006))
- Attention Mechanism  
(<https://wikidocs.net/22893>)