# Automatic Post Editing 최신 동향 정리

## 작성자: 박찬준

**Task 정의:** Automatic Post Editing이란 기계번역 시스템의 결과물을 교정하여 더 나은 번역문을 만들어내는 Task.

### 한눈에 보는 주요 특징

- WMT에서 매년 Shared Task를 열고 있다.
- 2019 WMT에서 **Unbabel** 이라는 회사가 우승을 차지 했으며 핵심 아이디어는 BERT 적용
- 국내에서 **포항공대**에서 많은 연구가 이루어지고 있으며 2019 WMT에서 근소한 차이로 2등 차지
- 학습데이터 구성은 **(SRC,MT,PT)**로 이루어져 있으며 각각 소스문장(원문), MT(번역문), PT(사후교정 결과)로 구성되어 있다. WMT에서 제공하는 데이터셋과 eSCAPE데이터를 많이 사용한다.
- 학습데이터 구조적 특성에 착안하여 최근 SRC(원문), MT(번역문)을 별도의 소스로 간주하는 **다중 소스 번역 문제(Multi Source translation Problem)**으로 간주함
- **Multi Source Transformer** 구조를 바탕으로 많은 연구가 이루어지고 있음
- 2018년까지는 각 입력과 교정문 사이의 의존성을 별도로 학습하고 이들을 더하여 최종 입출력 의존성을 얻었으나 2019년 부터는 **입력과 교정문 사이의 관계성을 고려하는 연구**가 진행됨

## 주요역사

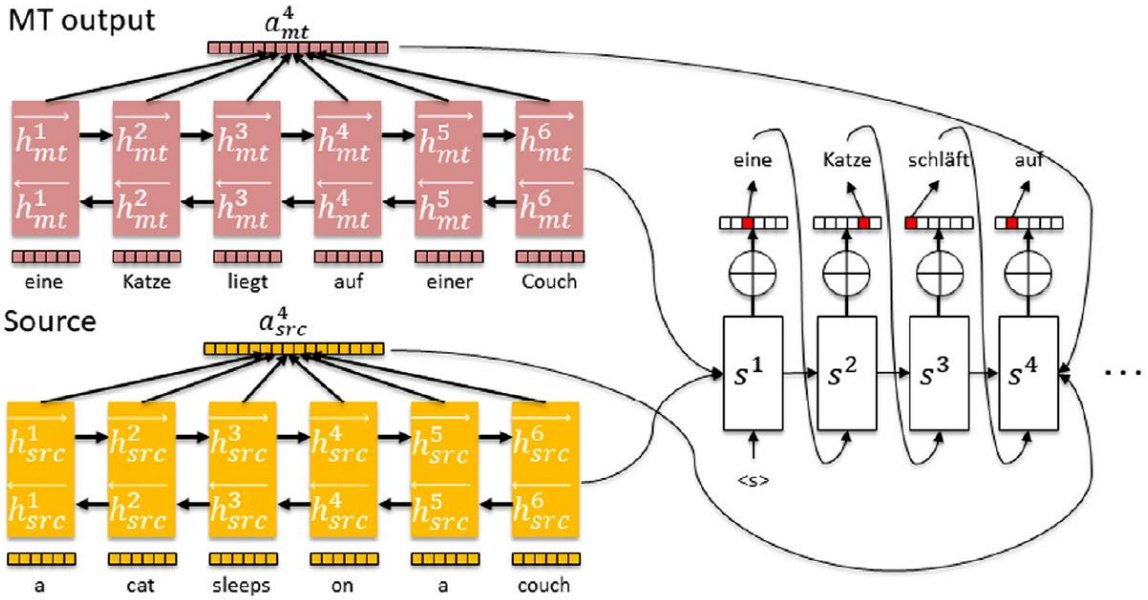
- 1994년 최초의 사후 교정 시스템이 제안되었으며 일본어를 영어로 번역한 문장에서 관사 선택의 문제를 해결하기 위한 연구가 진행됨.
- 2000년 기본적인 사후 교정 시스템의 설계가 제안됨
- 2007년 사후 교정을 번역문제로 바라보게 됨. Phrase based SMT를 적용한 연구가 진행됨
- 2015년 WMT의 공동 캠페인 과제로 선정됨. 이로 인해 형식 및 성능평가 방법(TER,BLEU)이 명확해짐
- 2017년 Conv to Conv를 이용한 모델이 우승
- 2018년 Multi Source Transformer를 이용한 모델이 우승 (국내: 3등)
- 2019년 BERT를 이용한 모델이 우승 (국내: 2등)

## 평가방법

TER(Translation Edit Rate, 번역 오류율)과 BLEU를 적용하며 TER이 낮을수록 좋다.

## 흐름으로 보는 관련연구

### WMT2016 : Multi Source translation Problem)으로 문제를 바라봄




기존 Encoder-Decoder 구조에 새로운 입력을 처리하기 위한 Encoder를 추가한 것

Attention을 각 입력에 별도로 적용하여 문맥정보를 가지고 옴

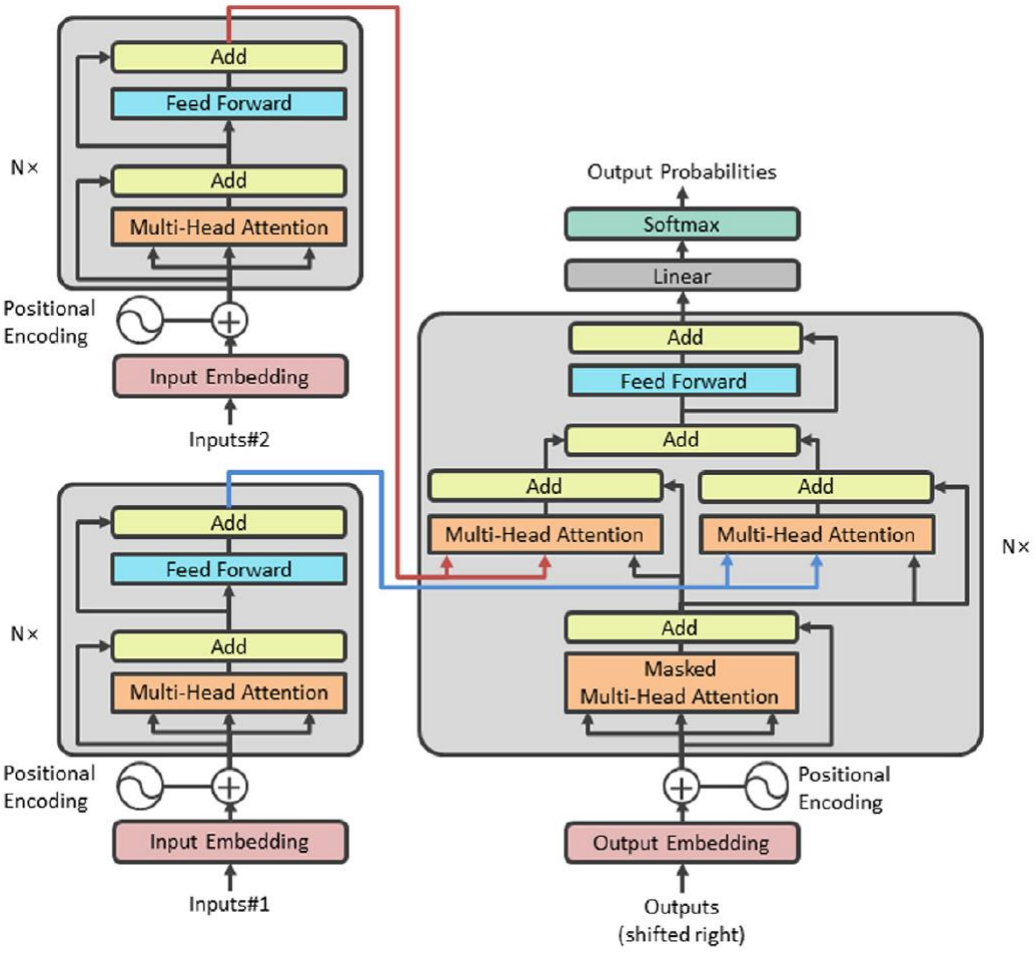
### WMT2017: Convolution Seq2Seq이 우승

### WMT2018: Transformer 기반 (모든 시스템)

다중 소스 번역 문제로 정의

1. 번역문과 원문을 하나의 입력으로 처리 (원문의 정보는 학습자질로 사용)
2. 번역문과 원문을 각각의 입력으로 처리

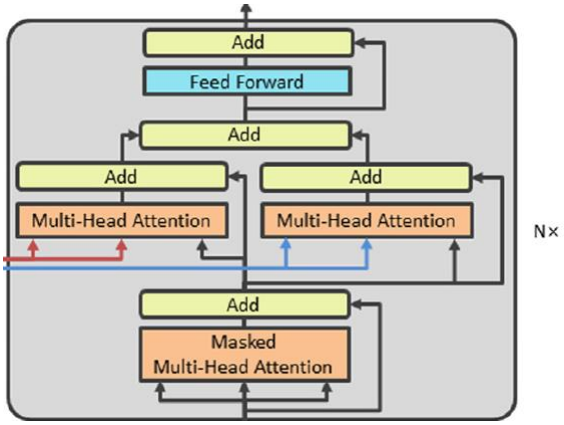
Transformer 같은 경우 기본적으로 하나의 Encoder를 사용하기에 1의 경우 모델의 구조를 변경하지 않고 그대로 사용 가능 하나 2의 경우 모델의 구조를 변경시켜야 한다. 즉 **다중 Encoder 구조**로 변경해야 한다.



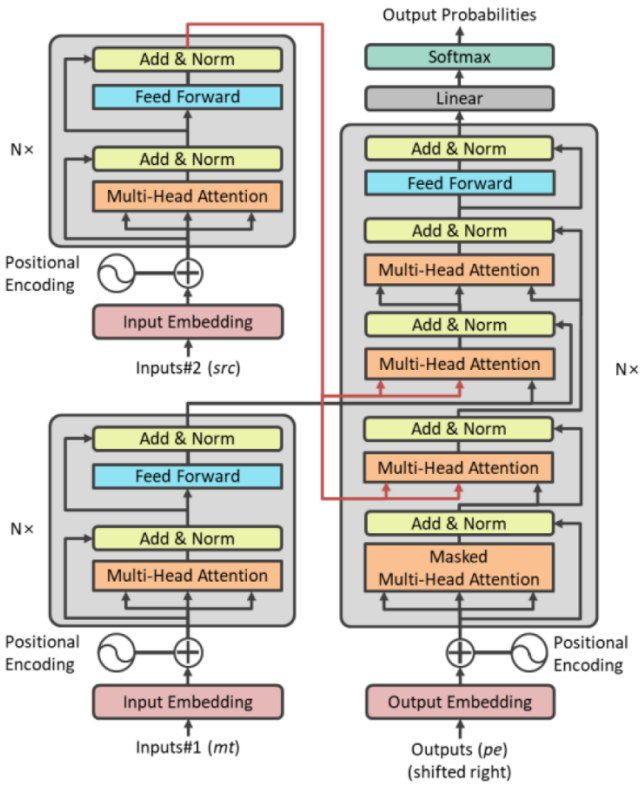

원문을 처리하는 Encoder와 번역문을 처리하는 Encoder를 **각각 구성**한다.

2개의 Encoder는 각각 번역문과 원문을 입력으로 받아 각 문장들의 자가 의존성을 학습하고 결과를 Decoder에 전달하게 됨.

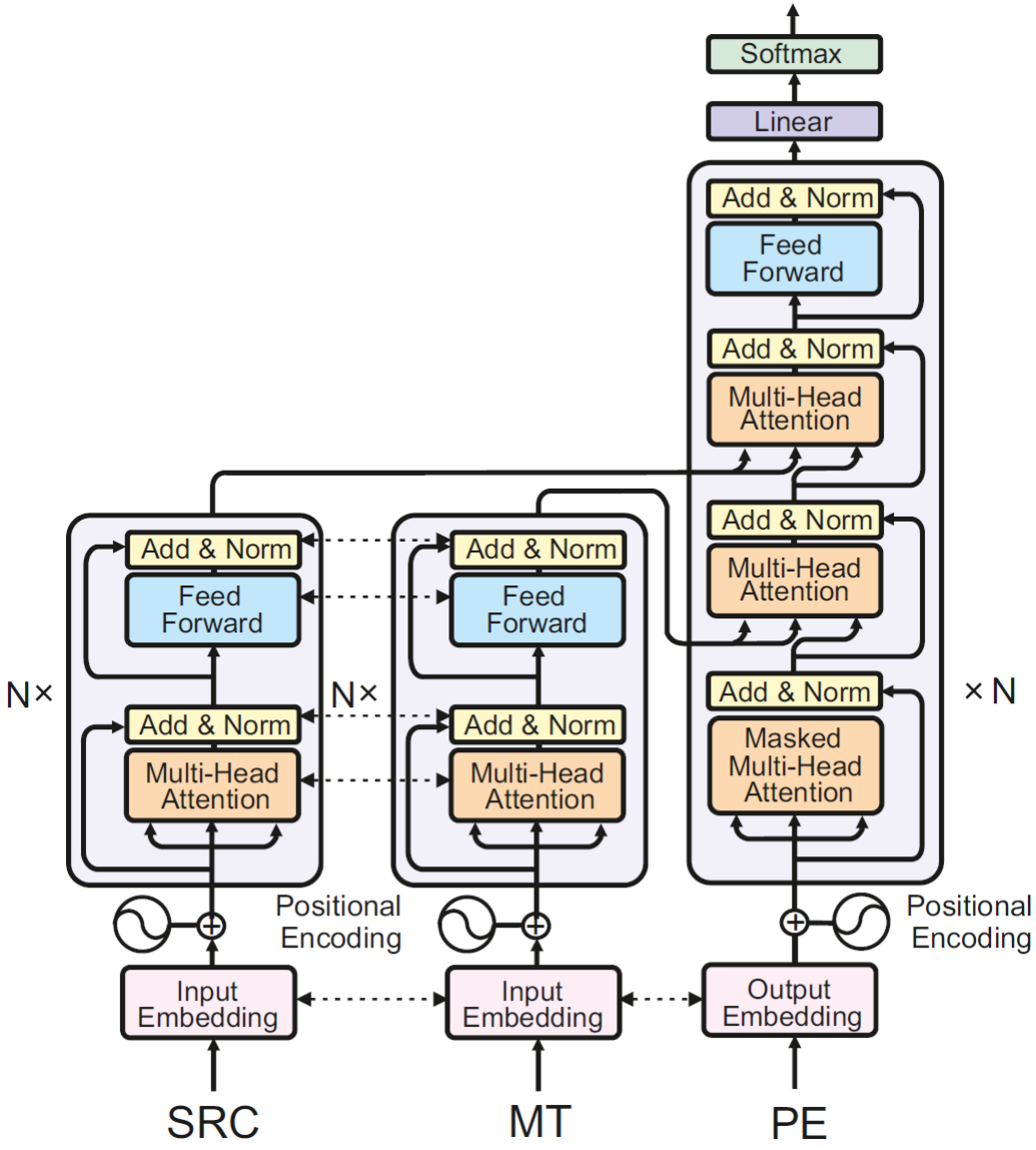
Decoder의 내부구조를 변경해야 하는데 Decoder 같은 경우 하나의 Encoder 출력만을 고려하여 작성되었기에 **각 Encoder 출력에 대해 별도로 Attention Layer를 구성**해야 한다. 또한 그 결과를 더하여 이후 Layer로 전달할 수 있도록 Sub Layer를 구성해야 한다.




결론적으로 Decoder에서 각 Encoder 출력에 대해 별도의 Attention을 수행하고 그 결과를 더하여 결과에 반영하는 구조이다.



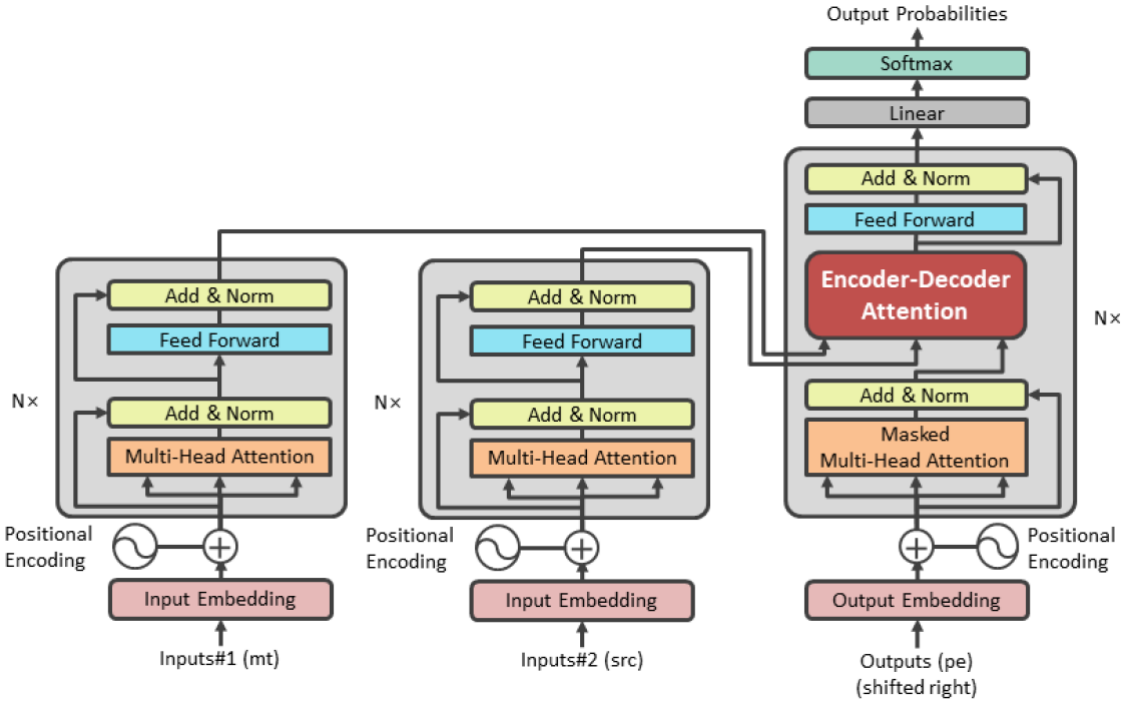

위와 같이 Sequential 하게 진행하는 것도 좋은 방법론이 될 수 있다.




WMT 2018에서 가장 우수한 성적을 보인 모델의 구조이며 Common Parameters 를 사용하는 것이 특징이다.

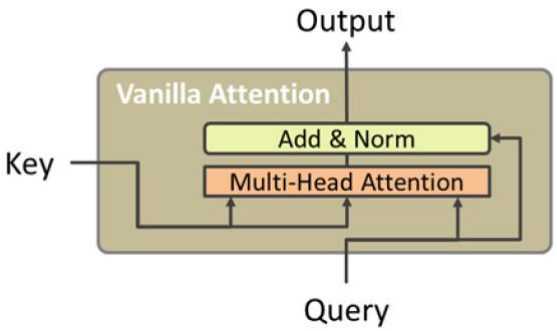
## 2019년 가장 최신 동향

## Decoder 구조에 대한 연구 in 포항공대




Encoder Decoder Attention 부분에 대한 다양한 연구를 진행함.

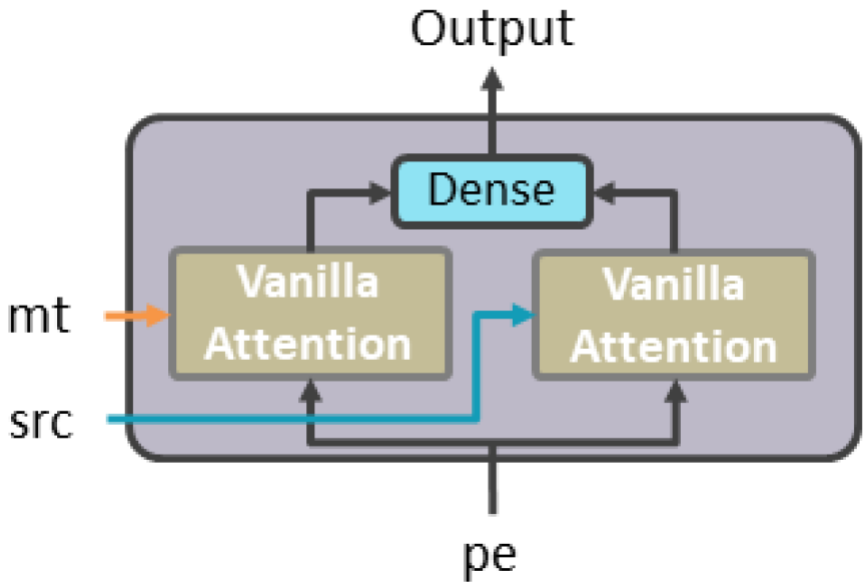
아래 그림은 기본적인 Transformer Vanilla Encoder Decoder Attention이다.



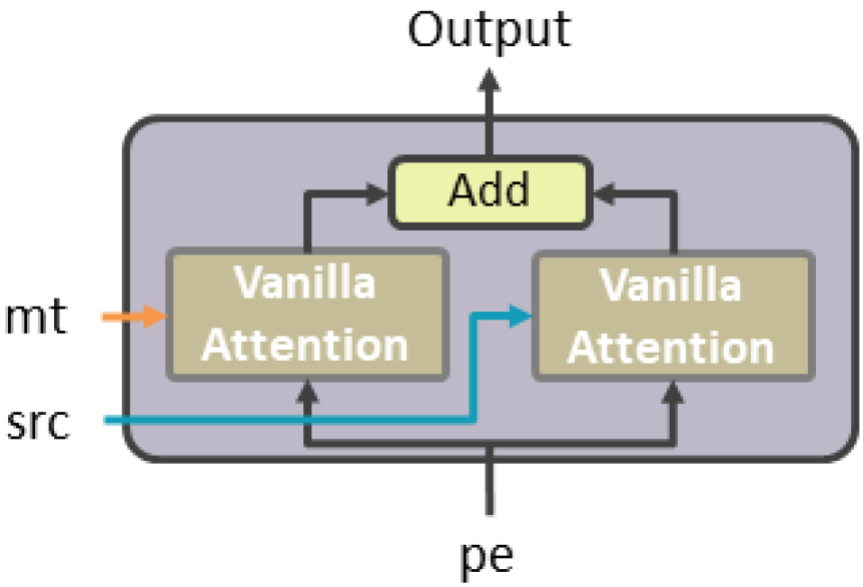

Key와 Value는 Encoder로부터 Query는 Decoder로부터 온다.

늘어난 Encoder에 따라 Decoder 입력과 Encoder 출력 간의 연결을 구성해야 한다.

## #Baseline

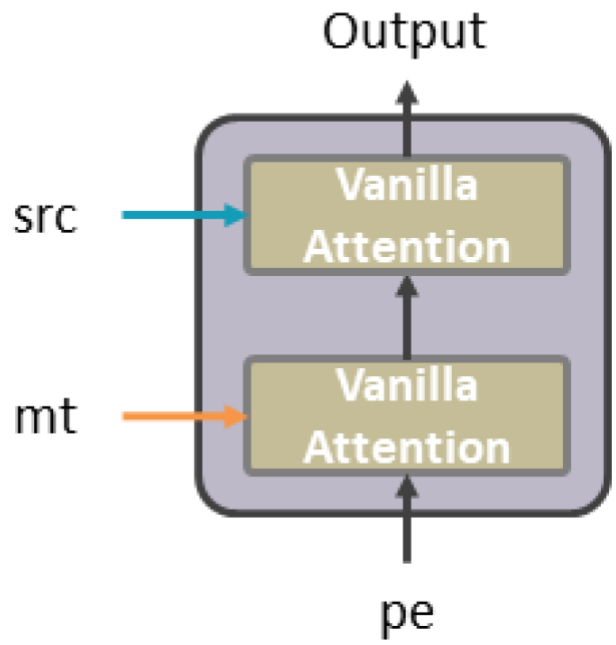



## #A

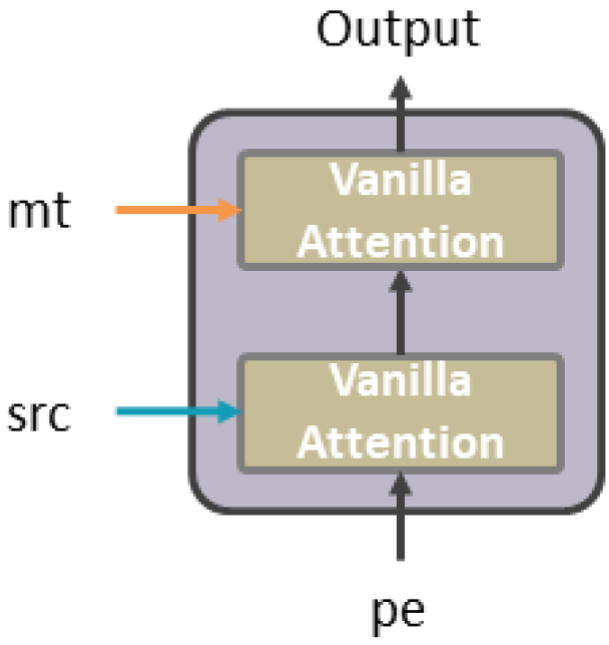



위 그림은 두 Attention의 결과를 **단순히 더하는 구조**.

**#B**

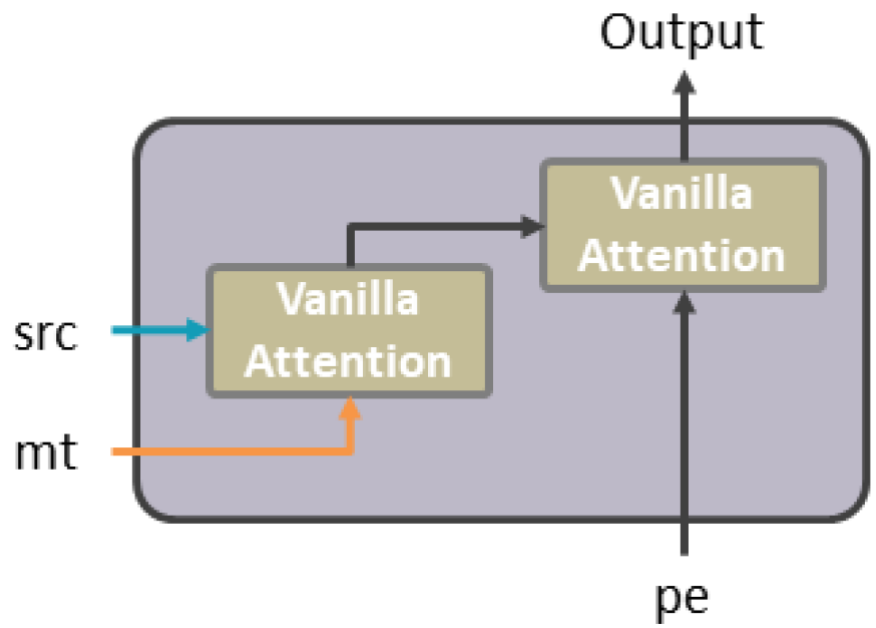



**#C**




B와 C는 각 Attention 결과를 순차적으로 참조하되 **원문과 번역문 중 어떤 쪽을 먼저 고려**할 것인지에 차이이다.

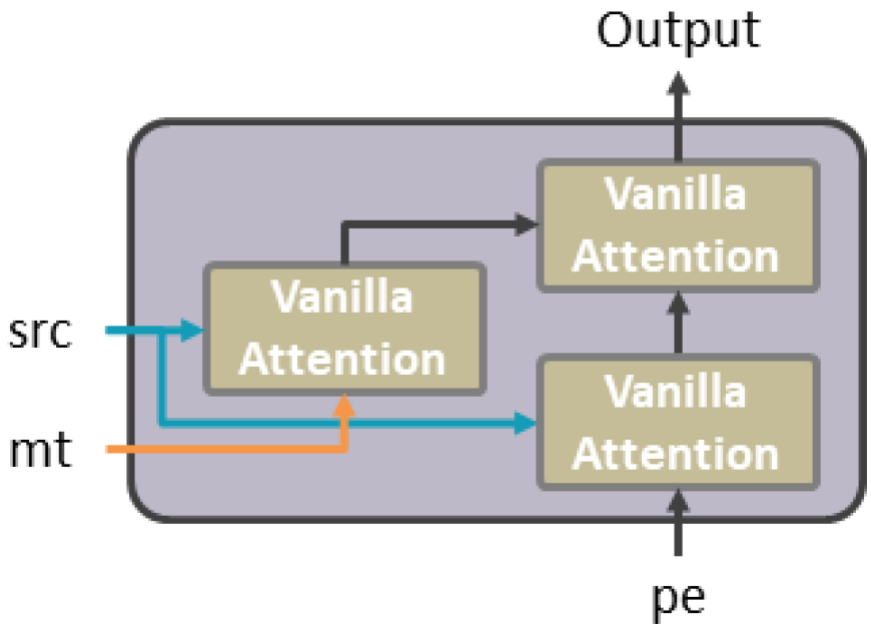
## #D




D는 번역문과 원문 사이의 Attention을 계산하여 전달하는 구조

**즉 번역문과 원문 사이의 관계성을 모델링 할 수 있다.**

## #E



**E는 D에 더해 src(원문)과의 Attention을 먼저 참조하도록 구성한 구조.**

| 모델 | dev | test('16) | test('17) |
|---|---|---|---|
| No-edit | 24.814 | 24.765 | 24.481 |
| Baseline | 19.390 | 19.705 | 19.883 |
| #A | 19.015 | 19.506 | 19.756 |
| #B | 19.192 | 19.469 | 19.847 |
| #C | 19.415 | 19.597 | 19.681 |
| #D | 19.122 | 19.220 | 19.670 |
| #E | 19.000 | 19.155 | 19.477 |

**실험결과 E가 가장 좋은 성능을 보였다. (TER 기준)**

**즉 원문과 번역문 사이의 관계를 고려하는 것이 성능향상에 중요한 요인이다.**

**원시문과 기계번역문간 효과적 관계모델링에 대한 연구 in 포항공대**

기존 Multi head Attention 수식

$$MultiHead(Q, K, V) = Concat(head_1, .., head_n)\, W^O$$
$$head_i = Attn(QW_i^Q, KW_i^K, VW_i^V)$$
$$Attn(Q, K, V) = softmax(QK^T/\sqrt{d_k})\, V$$

Multi Source Transformer 수식

$$C_{mt} = MultiHead(\mathrm{PE}, \mathrm{MT}, \mathrm{MT})$$
$$H = LayerNorm(\mathrm{PE} + C_{mt})$$
$$C_{src} = MultiHead(H, \mathrm{SRC}, \mathrm{SRC})$$

**PE: 교정문 SRC: 원문 MT: 번역문**

연구 기여 내용 정리

1. **원시문과 번역문 각각을 독립적으로 인코딩 했던 기존 연구와 다르게 번역문 인코딩 과정에서 원시문의 문맥정보를 포함하는 공동표현(Joint Representaion) 모델링**
2. **Decoder에서 공동표현과 독립적으로 인코딩 된 번역문을 함께 고려해 문맥벡터를 생성하는 결합 주의 집중(Combined Attention) 계층을 제안**

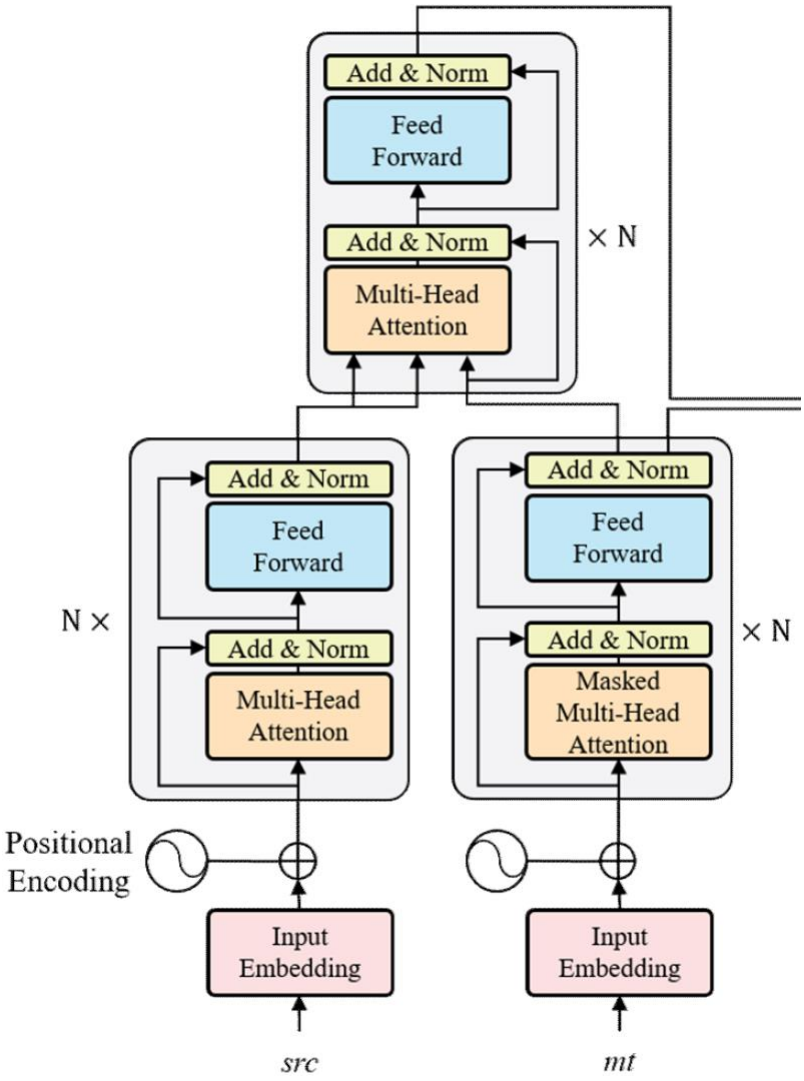
**공동표현(Joint Representation)이란?**

SRC와 MT의 독립적인 인코딩 표현을 생성한다

단 MT 인코딩 모듈은 기계번역 시스템의 디코딩 과정을 모방하기 위해 Masked Multi-Head Attention 적용

각각 인코딩 된 SRC와 MT는 별도의 인코딩 모듈을 통해 **공동 표현**을 생성하며 , Multi Head Attention 계층으로부터 각 번역 단어에 원시문장의 문맥정보가 포함된 인코딩 결과를 얻을 수 있다.

$$C_{joint} = MultiHead(\mathrm{MT}, \mathrm{SRC}, \mathrm{SRC})$$

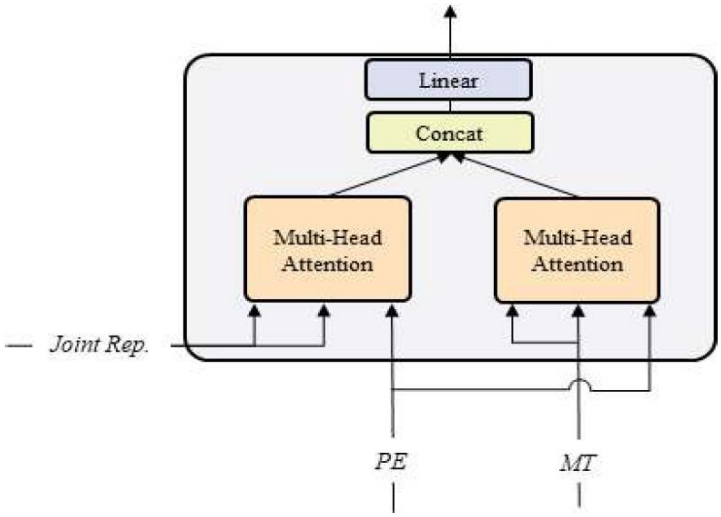


그림은 위와 같다.

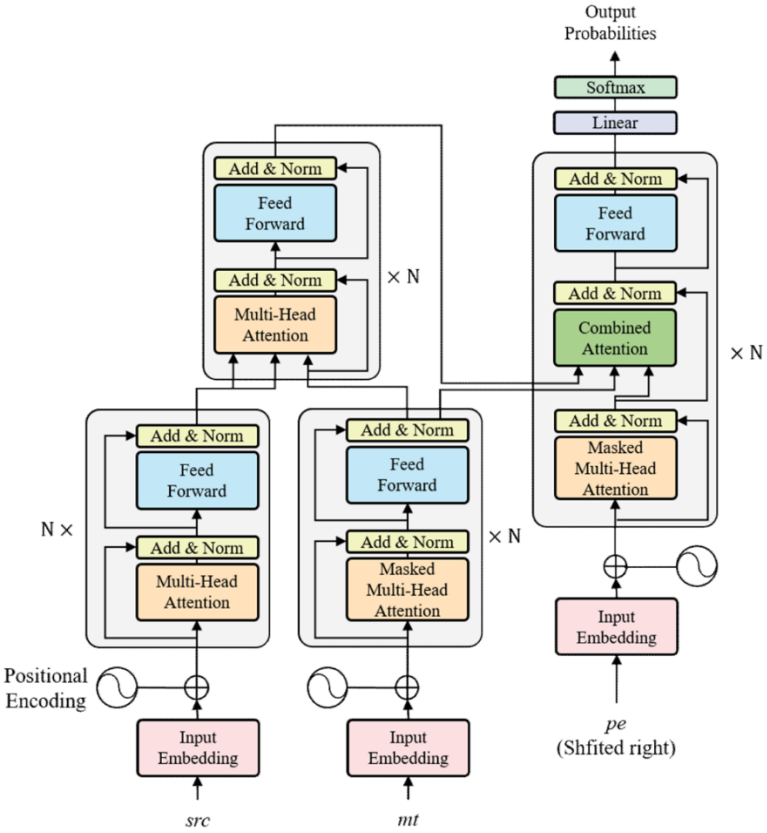
**Combined Multi Head Attention이란?**

Encoder의 두 출력(공동 표현과 독립적으로 인코딩 된 번역문 표현)을 함께 고려해 교정단어를 생성하기 위한 디코딩 모듈이다. 교정 단어 생성 시 중요도에 따라 두 출력에 다른 가중치가 부여되며, 가중 합을 통해 최종 결과를 얻는다.

$$C_{combied} = W_1 MultiHead(\mathrm{PE}, \mathrm{Joint}, \mathrm{Joint}) + W_2 MultiHead(\mathrm{PE}, \mathrm{MT}, \mathrm{MT})$$




**Combined Multi head Attention의 구조는 위와 같다.**




**전체 모델 구조는 위와 같다.**

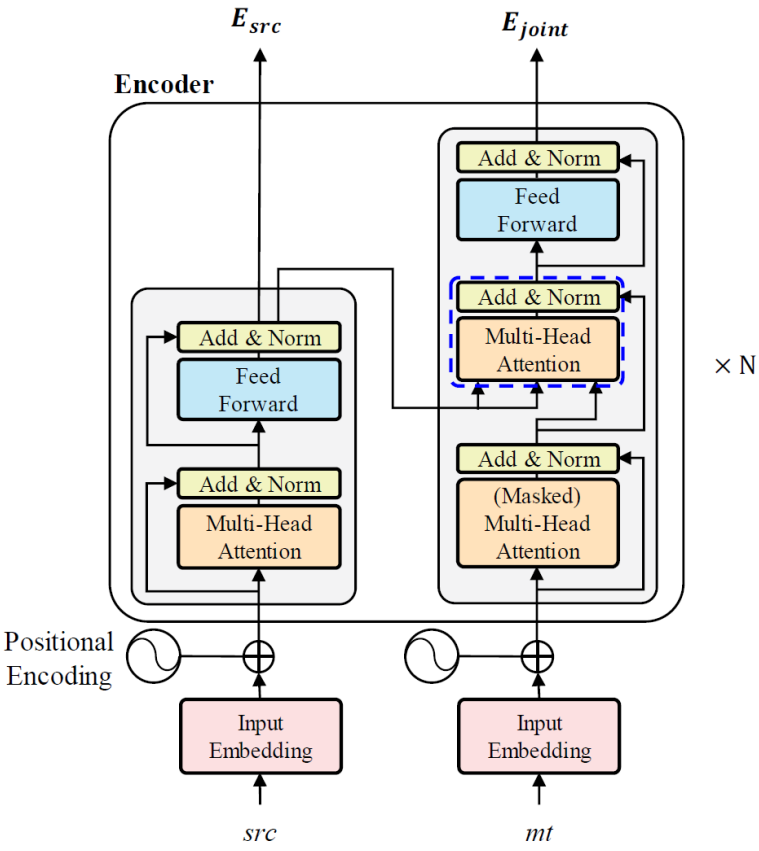
## WMT2019 2등 모델 분석

기존 WMT 2018의 문제점: SRC와 MT를 각각의 분리 된 Encoder를 적용하여 두 개의 값을 단순히 Sequential하게 처리하거나 단순 Concatenating을 진행함. 즉 두개의 관계를 파악하기 쉽지 않음.

크게 2가지를 주목해야

1. Joint Multi Source Encoder
2. Multi Source Attention Layer

먼저 Encoder를 살펴보자




Joint Representation이 핵심이다.

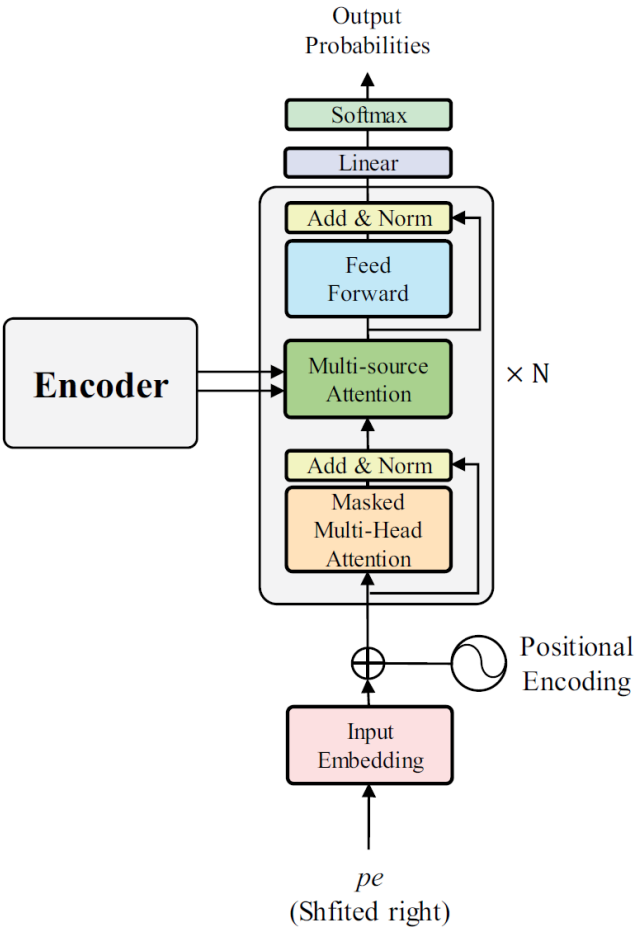
$$C_{src}^{i} = \text{MultiHead}(H_{mt}^{i}, H_{src}^{i}, H_{src}^{i})$$

$$H_{joint}^{i} = \text{LayerNorm}(H_{mt}^{i} + C_{src}^{i})$$

또한 주목할점은 Feeds into each Attention Layer the src embeddings from the same level

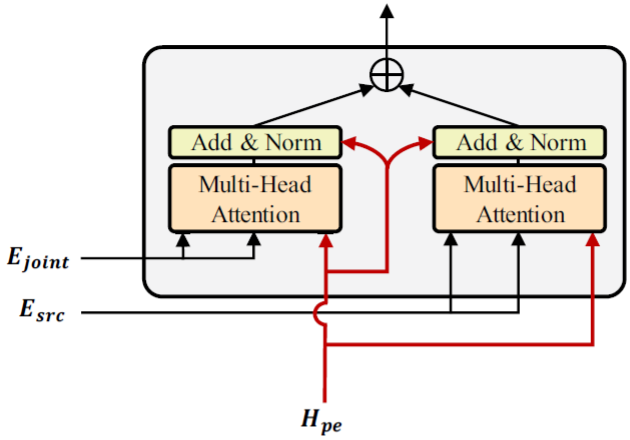
MT 인코딩 모듈은 기계번역 시스템의 디코딩 과정을 모방하기 위해 Masked Multi-Head Attention 적용

**그 다음 Decoder 를 살펴보자**

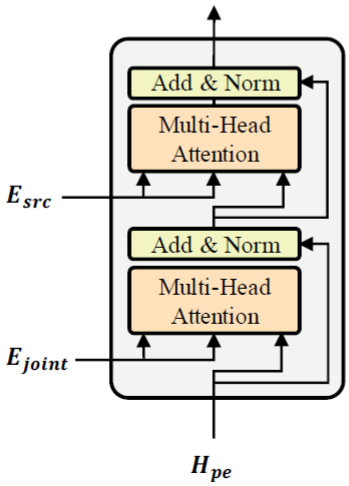


Multi Source Attention Layer 를 제안함. 2 가지 구조를 제안하였음

1. Multi Source Parallel Attention
2. Multi Source Sequential Attention




(a)




(b)

**Multi Source Parallel Attention ➔ Linear Combined 한 것 !**

$$H_{parallel} = H_1 + H_2$$

where

$$H_1 = \text{LayerNorm}(H_{pe} + C_{joint})$$
$$H_2 = \text{LayerNorm}(H_{pe} + C_{src})$$
$$C_{joint} = \text{MultiHead}(H_{pe}, E_{joint}, E_{joint})$$
$$C_{src} = \text{MultiHead}(H_{pe}, E_{src}, E_{src}).$$

**Multi Source Sequential Attention ➔ Sequentially Combine 한 것 !**

$$H_{seq} = \text{LayerNorm}(H' + C_{src})$$

where

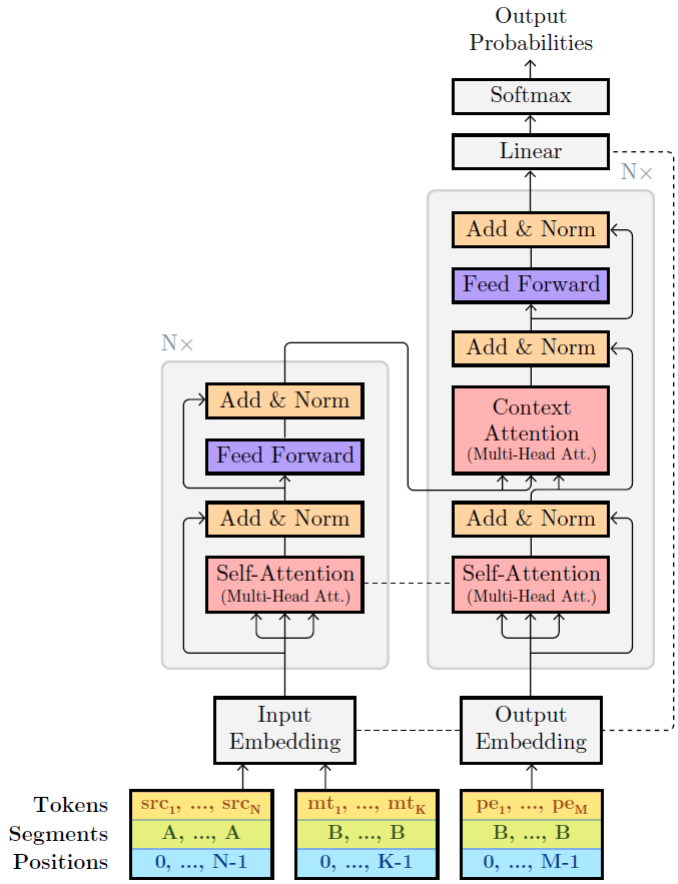
$$H' = \text{LayerNorm}(H_{pe} + C_{joint})$$
$$C_{src} = \text{MultiHead}(H', E_{src}, E_{src})$$
$$C_{joint} = \text{MultiHead}(H_{pe}, E_{joint}, E_{joint}).$$

실험결과는 아래와 같다.

| Systems | TER | BLEU |
|---|---|---|
| UNBABEL_Primary | **16.06** | 75.96 |
| POSTECH_Primary (top2Ens8) | 16.11 | **76.22** |
| POSTECH_Contrastive (var2Ens8) | 16.13 | 76.21 |
| USSAR-DFKI_Contrastive | 16.15 | 75.75 |
| POSTECH_Contrastive (top1Ens4) | 16.17 | 76.15 |
| Tebbifakhr et al. (2018) | 16.46 | 75.53 |
| Junczys-Dowmunt and Grundkiewicz (2018) | 16.50 | 75.44 |
| Shin and Lee (2018) | 16.70 | 75.14 |
| Baseline | 16.84 | 74.73 |

## WMT2019 1등 모델 분석

Bert Based Encoder-Decoder 구조이다 !




Single BERT Encoder 를 사용하며 SRC 와 MT 의 **Joint Representation** 을 제안함

Multilingual BERT 를 Pretrain 모델로 사용하였음.

**SRC 와 MT 를 [SEP]로 연결**

**Conservativeness Penalty** 를 사용하였음. ➔

그냥 휴리스틱하게, 학습데이터의 교정률이 적기 때문에 이런 특징반영을 위해, src, mt 에 등장하지 않은 단어들에 대해 penalty 를 주는 방법인듯 함.

따라서, 각 pe 단어를 생성하는 output layer 단에서의 벡터의 각 차원은 각 단어의 생성확률을 가지는 값이고, 각 차원 값에 위에서 src, mt 에 등장하지 않은 단어에 해당하는 차원에 penalty 값을 부여하는 방법

그 값은 하이퍼파라미터릭 하게 주는거 같고. 근데 너무 특정 학습데이터의 분포에 dependent 한 방법...

**최종 결과**

| System | ↓Ter | ↑BLEU |
|---|---|---|
| **Ours (Unbabel)** | **16.06**⋆ | 75.96 |
| POSTECH | 16.11⋆ | **76.22** |
| USSAR DFKI | 16.15⋆ | 75.75 |
| FBK | 16.37⋆ | 75.71 |
| UdS MTL | 16.77 | 75.03 |
| IC USFD | 16.78 | 74.88 |
| Baseline | 16.84 | 74.73 |
| ADAP DCU | 17.07 | 74.30 |

**결론**

- 아이디어 자체는 포항공대 모델이 압도적으로 좋다.
- Unbabel 모델은 BERT 를 적용한 것일 뿐.....
- 내년에 Pretrain ➔ Fine Tuning 기조가 유지될 것으로 판단됨
- XLM, MASS 와 같은 인코더 디코더 Pretrain 하는 구조를 적용해보면 좋을 듯

참고자료:

[1]MS-UEdin Submission to the WMT2018 APE Shared Task Dual-Source Transformer for Automatic PostEditing

[2]Multi Encoder Transformer Network for APE (신재훈,이종혁)

[3]Transformer 기반 번역문 사후 교정 (신재훈, 김영길, 이종혁)

[4]Transformer-based Automatic Post-Editing Model with Joint Encoder and Multi Source Attention of Decoder (이원기, 신재훈, 이종혁)

[5]Unbabel's Submission to the WMT2019 APE Shared Task BERT-Based Encoder-Decoder for Automatic Post Editing

[6]다중 인코더 Transformer 기반 번역문 자동 사후 교정 모델의 디코더 구조 연구 (신재훈,이원기, 김영길, 이종혁)

[7]원시문과 기계번역문 간 효과적 관계모델링을 통한 Transformer··기반 번역문 자동 사후 교정 (이원기, 신재훈 , 김영길, 이종혁)