

# Multi-Task Deep Neural Networks for Natural Language Understanding

Xiaodong Liu, Pengcheng He, Weizhu Chen, Jianfeng Gao

ACL 2019

김유리

## Index

Introduction

MT-DNN

GLUE Tasks

Experiments

Appendix

# Introduction

## Introduction

MT-DNN?

**MT-DNN(Multi-Task Deep Neural Network) 제안**

ACL 2019

Microsoft Dynamics 365 AI

## Introduction

### MT-DNN 핵심

BERT에 multi-task learning(GLUE의 9개 task)을 수행하여 성능 개선

- 다양한 Task의 Supervised Dataset을 모두 사용하여 대용량 데이터로 학습
- Multi-task Learning을 통해 특정 Task에 Overfitting 되지 않도록 Regularization

다음 Task들에 대해 SOTA (BERT 보다 높은 성능)

- 8개의 GLUE Task
- SNLI와 SciTail로 Domain Adaptation
  - Fine Tuning 데이터가 적을 때도 Classification 정확도가 꽤 높음

## Introduction

MT-DNN 핵심

즉, 여러가지 Task의 많은 데이터를 이용해 general representation을 표현해서  
새로운 Task와 Domain에 적용할 때 도움을 주자

## Introduction

MT-DNN 핵심

즉, 여러가지 Task의 많은 데이터를 이용해 general representation을 표현해서  
새로운 Task와 Domain에 Language Representation?

## Introduction

### Language Representation

#### Language Representation

: 자연어 이해를 위해 word나 sentence를 vector로 표현 하는 것

## Introduction

### Language Representation

NLU에서 word나 sentence의 Vector Representation을 생성하는 2가지 방식

- Multi-Task Learning
  - 여러 Task의 Labeled Dataset을 활용해 1개의 모델을 Supervised Learning
  - 가정 : 어떤 Task의 학습 효과가 다른 Task의 성능에 영향을 줄 거야!
  - Ex. 스케이트를 잘 타는 사람이 스키를 잘 탈 수 있어
- Language Model Pre-training
  - Unlabeled Dataset 활용
  - 문장에서 특정한 단어를 맞추는 방식으로 Unsupervised Learning
  - Ex. ELMo, BERT, …

## Introduction

### Language Representation

NLU에서 word나 sentence의 Vector Representation을 생성하는 2가지 방식

- Multi-Task Learning

- 여러 Task의 Labeled Dataset을 활용해 1개의 모델을 Supervised Learning
- 가정 : 어떤 Task의 학습 효과가 다른 Task의 성능에 영향을 줄 거야!

**Multi-Task Learning의 기본 Idea!**

학습한 내용이 Task간에 서로 영향을 주게 되니까(가정)

여러 Task를 한꺼번에 학습하면 LM을 더 잘 학습 시킬 수 있지 않을까?

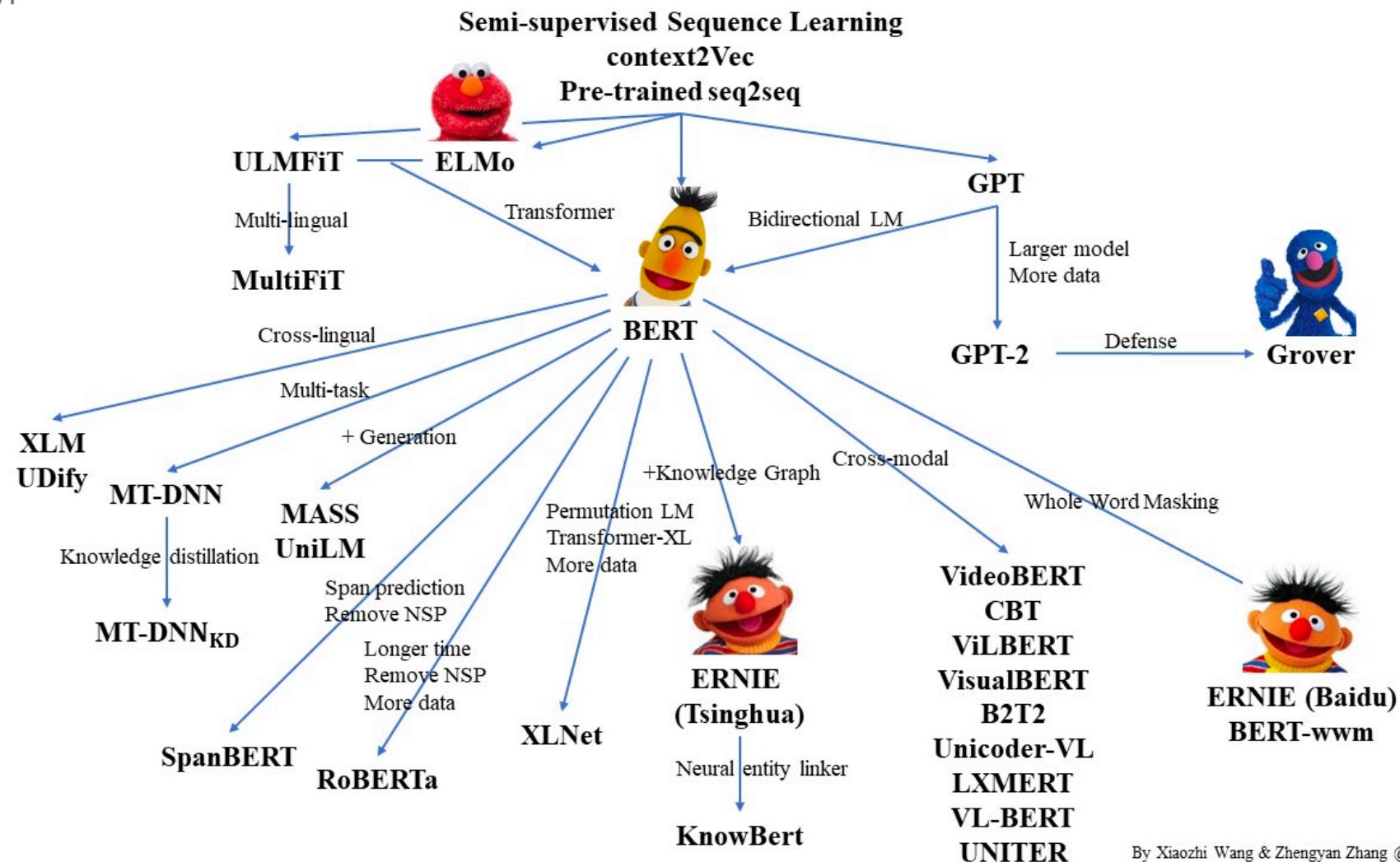
- Language Model Pre-training

- Unlabeled Dataset 활용
- 문장에서 특정한 단어를 맞추는 방식으로 Unsupervised Learning
- Ex. ELMo, BERT, ...

# Introduction

## Pre-trained LM

# Pre-trained Language Model

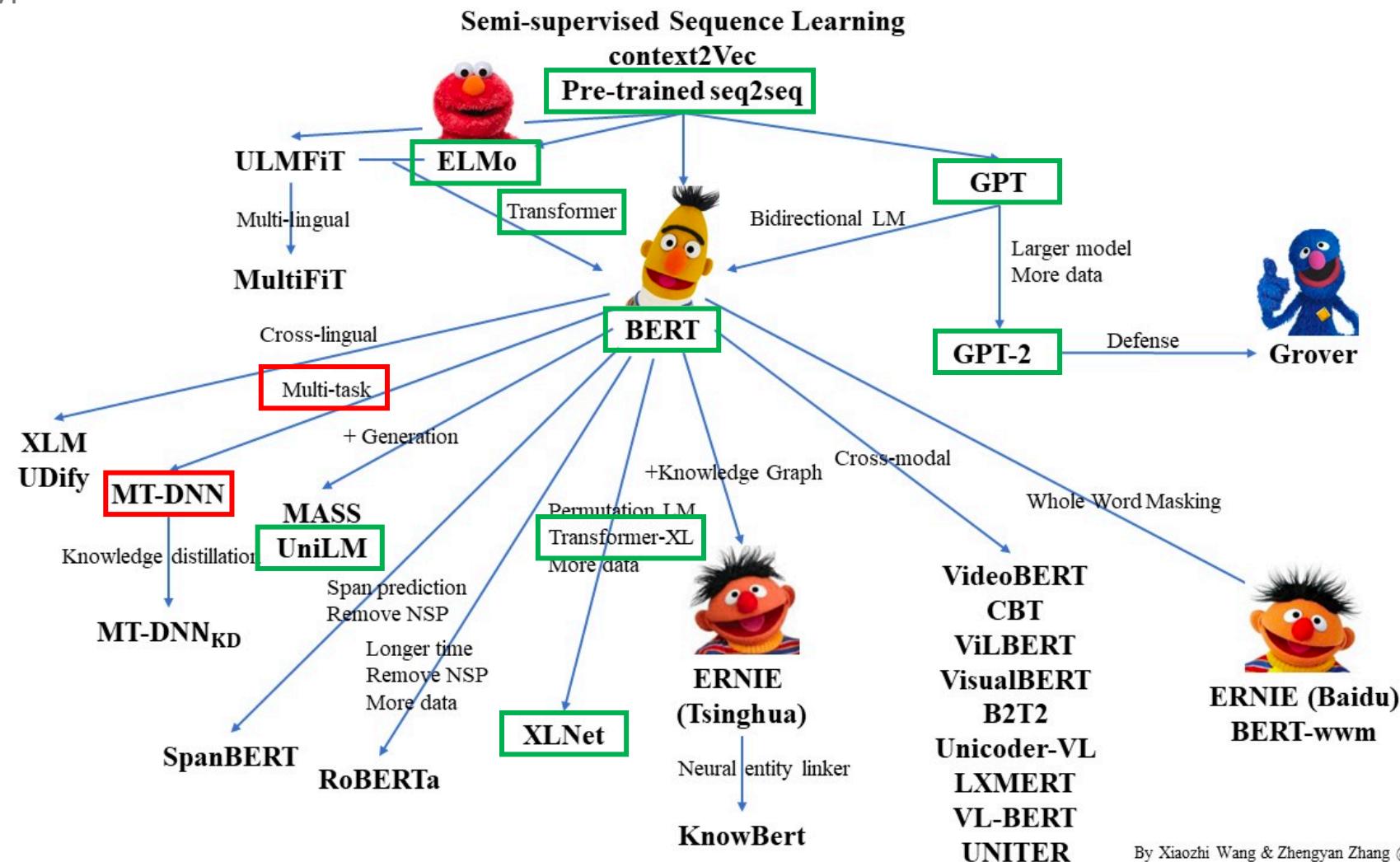


By Xiaozhi Wang &amp; Zhengyan Zhang @THUNLP

# Introduction

## Pre-trained LM

# Pre-trained Language Model



By Xiaozhi Wang &amp; Zhengyan Zhang @THUNLP

## Introduction

### MT-DNN 핵심 (반복)

BERT에 multi-task learning(GLUE의 9개 task)을 수행하여 성능 개선

- 다양한 Task의 Supervised Dataset을 모두 사용하여 대용량 데이터로 학습
- Multi-task Learning을 통해 특정 Task에 Overfitting 되지 않도록 Regularization

다음 Task들에 대해 SOTA (BERT 보다 높은 성능)

- 8개의 GLUE Task
- SNLI와 SciTail로 Domain Adaptation
  - Fine Tuning 데이터가 적을 때도 Classification 정확도가 꽤 높음

## Introduction

### MT-DNN 핵심 (반복)

BERT에 multi-task learning(GLUE의 9개 task)을 수행하여 성능 개선

- 다양한 Task의 Supervised Dataset을 모두 사용하여 대용량 데이터로 학습
  - Multi-task Learning 간단하게 보고 넘어가자 tting 되지 않도록 Regularization
  - BERT
  - Multi-Task Learning
  - GLUE
- 다음 보다 높은 성능)
- 8개의 GLUE Task
  - SNLI와 SciTail로 Domain Adaptation
    - Fine Tuning 데이터가 적을 때도 Classification 정확도가 꽤 높음

## Introduction

### BERT

#### BERT

- Book Corpus & Wikipedia Data 활용
- Masked Word Prediction & Next Sentence Prediction 방식으로 학습

## Introduction

### BERT

#### BERT

- **Masked Word Prediction**
  - 문장 안에서 특정 단어를 Masking
  - 다른 주변 단어를 활용하여 해당 Masked Word 예측하는 방식으로 학습
  - Ex. My dog is [Mask] → My dog is 7Bok
- **Next Sentence Prediction**
  - 2개의 문장이 주어졌을 때, 두 문장이 연결된 문장인지 아닌지 Classification하는 방식으로 학습
  - Ex. Input : the man went to the store [SEP] he bought a gallon of milk → IsNext

## Introduction

### Multi-Task Learning

#### Multi-Task Learning

: Supervised Task를 1개의 모델을 통해서 학습

- Multi-Task Learning의 이점
  - 대량의 Supervised Dataset을 활용한 학습이 가능
  - 모델이 특정한 Task에 Overfitting되지 않도록 Regularization 효과를 줄 수 있음

## Introduction

### Multi-Task Learning

#### Multi-Task Learning

: Supervised Task를 1개의 모델을 통해서 학습

- Multi-Task Learning의 이점
  - 대량의 Supervised Dataset을 활용한 학습이 가능

Data가 적을 때도 있는데 한번에 여러가지 Task의 Dataset을 모아서  
쓰다 보니, 비교적 많은 Dataset을 이용하게 됨

## Introduction

### GLUE

#### GLUE

: General Language Understanding Evaluation

- 여러가지 Task를 주고 그 Task에 대한 평가를 할 수 있음
- 리더보드를 통해 평가 결과를 랭킹 매길 수 있는 플랫폼
- 본 논문에서 GLUE의 9가지 Task를 이용

## GLUE Tasks

## GLUE Tasks

### GLUE 9개 Tasks

- **CoLA** : 문장이 문법적으로 맞는지 분류 (True/False)
- **SST-2** : 영화 리뷰 문장의 감정 분류 (Positive/Negative)
- **MNLI** : 문장 간 의미적 관계를 3가지로 분류 (Entailment, Contradiction, Neutral)
- **RTE** : 문장 간 의미적 관계를 3가지로 분류 (Entailment, Contradiction, Neutral)
- **WNLI** : 문장 간 의미적 관계를 3가지로 분류 (Entailment, Contradiction, Neutral)
- **QQP** : 문장 간 의미가 같음 여부를 분류 (True/False)
- **MRPC** : 문장 간 의미가 같음 여부를 분류 (True/False)
- **STS-B** : 문장 간의 의미적 유사도 점수로 예측
- **QNLI** : 지문과 질문 중 한 문장이 쌍으로 주어졌을 때, 해당 지문 문장에 질문의 답이 있는지 여부를 분류 (True/False)

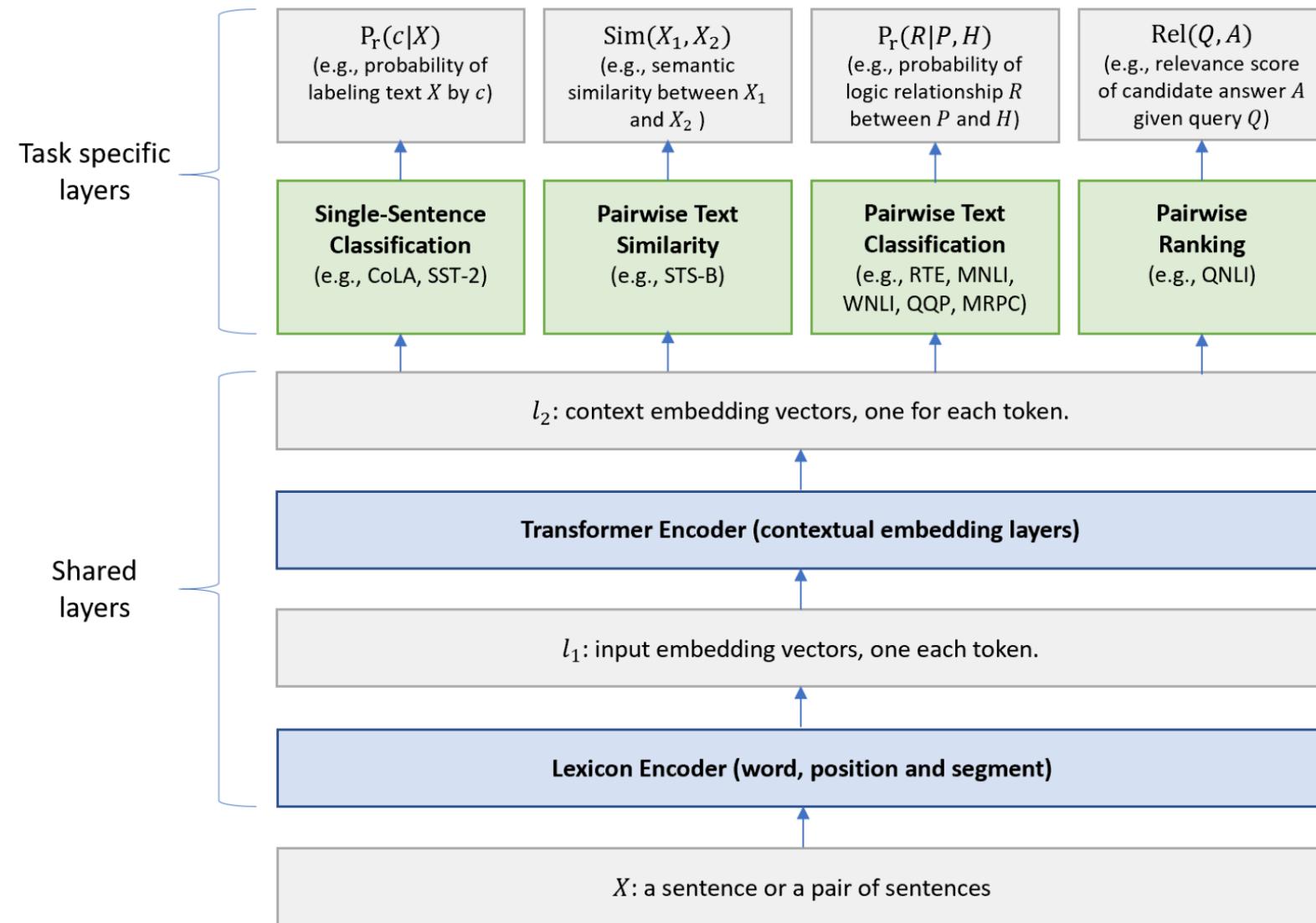
## GLUE Tasks

본 논문에서는 GLUE 9개 Task들을 크게 4가지 Task로 분류하여 사용

Single Sentence Classification	Text Similarity	Pairwise Text Classification	Relevance Ranking
하나의 문장이 Input으로 주어졌을 때 Class를 분류하는 Task	문장 쌍이 주어졌을 때, 점수를 예측하는 Regression Task	문장 쌍이 주어졌을 때, 문장의 관계를 분류하는 Task	질문 문장과 지문이 주어졌을 때, 지문 중 정답이 있는 문장을 Ranking을 통해 찾는 Task
CoLA SST-2	STS-B	RTE MNLI WNLI QQP MRPC	QNLI

# MT-DNN

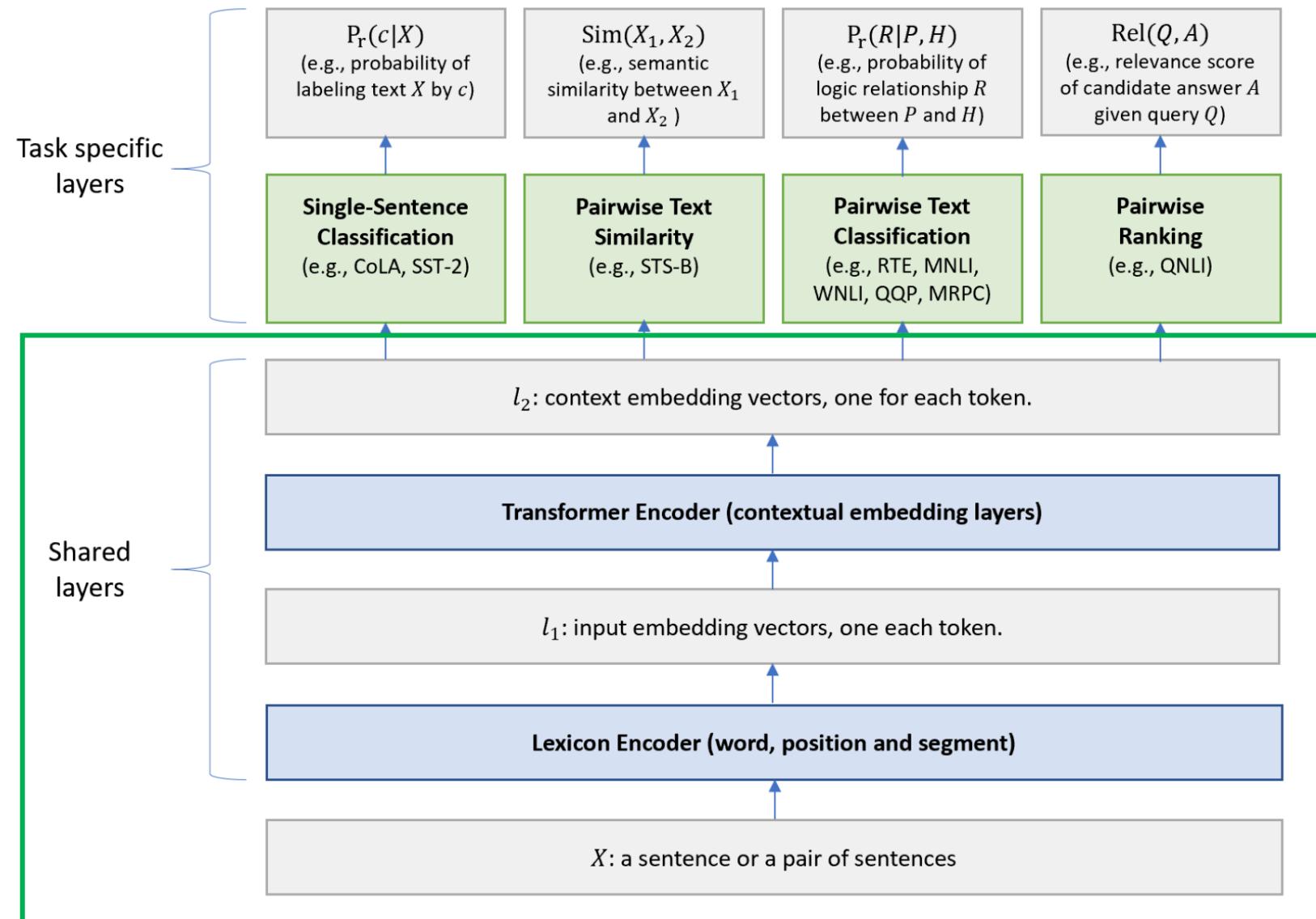
# MT-DNN Architecture



# MT-DNN

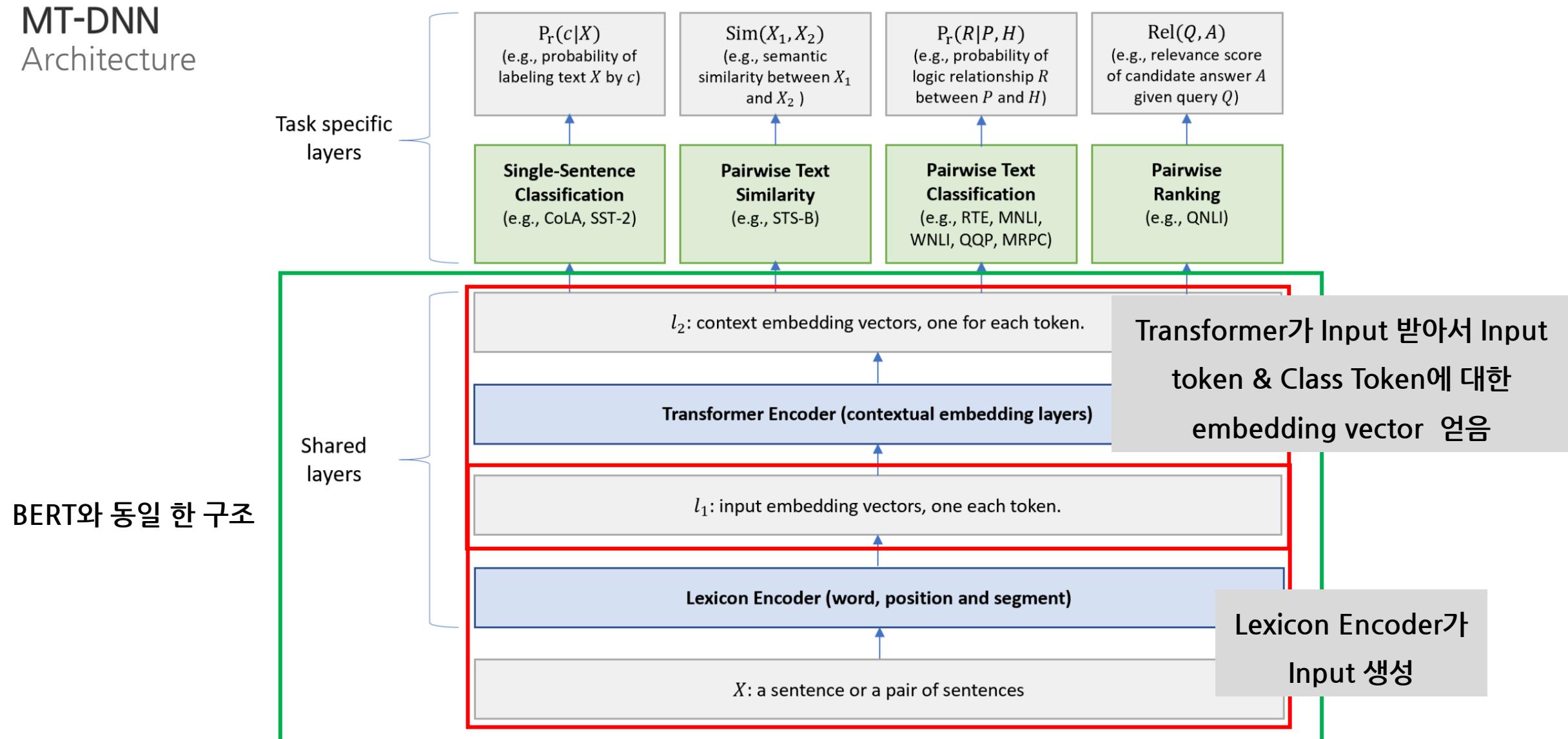
## Architecture

BERT와 동일 한 구조



# MT-DNN

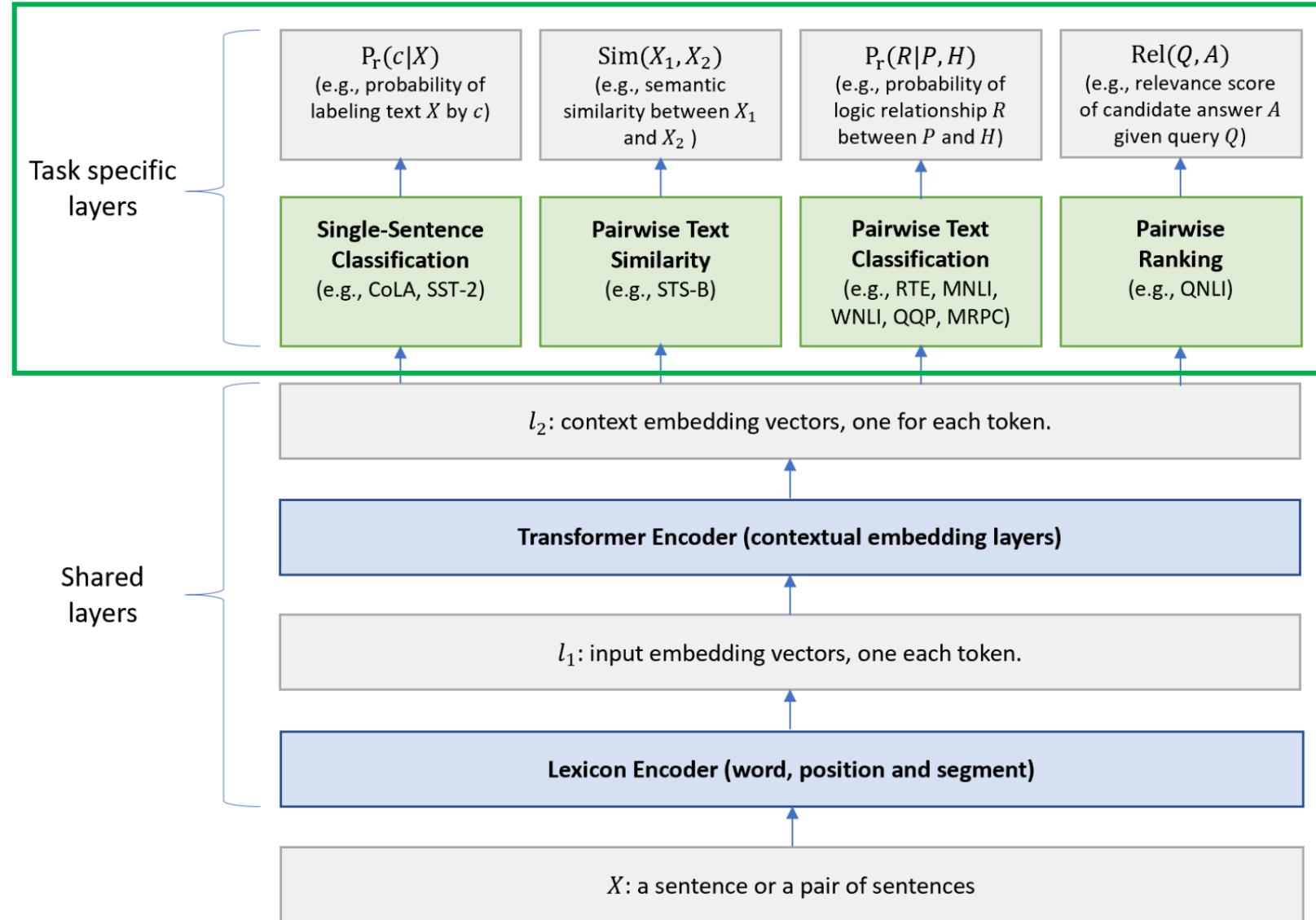
## Architecture



BERT와 동일 한 구조

# MT-DNN

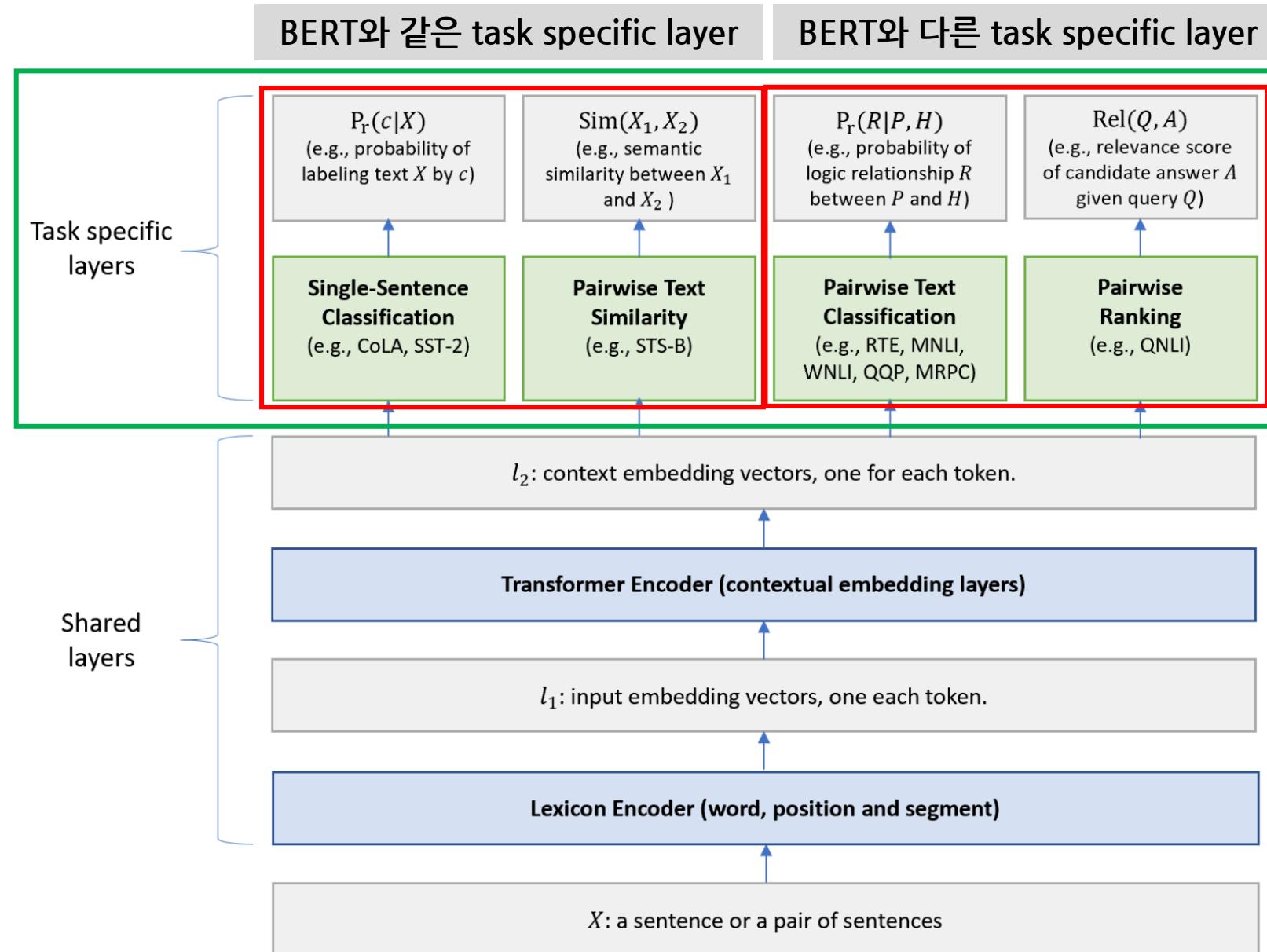
## Architecture



BERT와 동일한  
task specific layer  
사용할 때,  
아닐 때 모두 존재

# MT-DNN

## Architecture



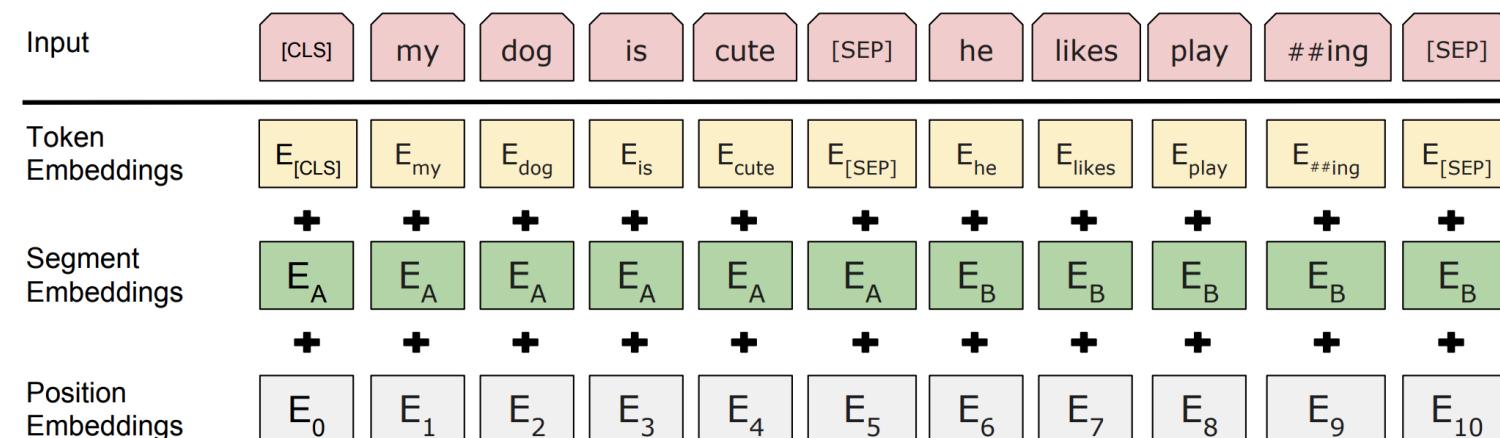
**BERT와 동일한  
task specific layer  
사용할 때,  
아닐 때 모두 존재**

# MT-DNN Input

## Lexicon Encoder

### Lexicon Encoder

- Input 생성 Layer (BERT와 동일)
- 3개의 Embedding으로 구성
  - Token Embeddings** : Wordpiece Embedding Vector (각 토큰에 대한 임베딩 vector)
    - 첫번째 토큰은 [CLS] Token으로 추후 Output에서 Classification 등을 위해 사용됨
    - 각 문장은 wordpiece로 Tokenization된 vector로 구성
    - [SEP] Token이 두 문장 사이의 구분자로 사용됨
  - Segment Embeddings** : 여러 문장아 Input으로 들어갈 때 몇 번째 문장인지 알려주는 임베딩 vector
  - Position Embeddings** : 각 Token의 position 정보를 표현하는 embedding vector

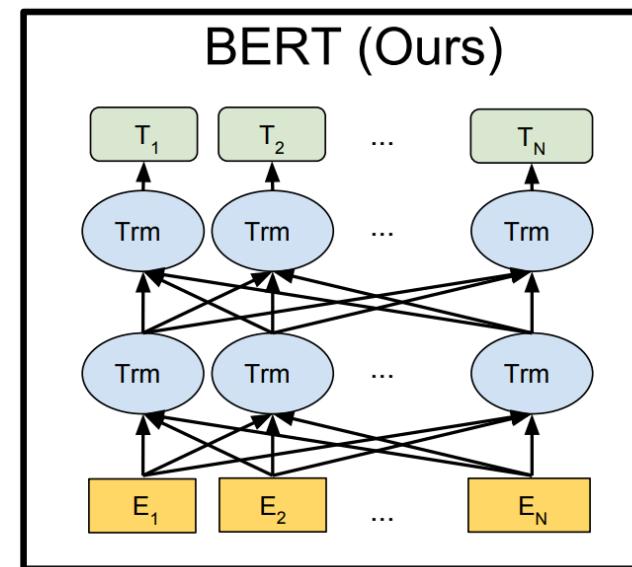


## MT-DNN Input

### Transformer Encoder

#### Transformer Encoder

- Lexicon Encoder로 부터 각 Token의 Input Vector를 입력으로 받아 Transformer를 통해 각 Token의 Output Vector 추출
- Transformer기 때문에 생성된 Vector는 Self Attention을 통해 주변 Token 정보를 반영한 Contextual Embedding Vector



## MT-DNN Output

## Single Sentence Classification

## Single Sentence Classification (BERT와 같은 구조)

- CoLA : 문장이 문법적으로 맞는지 Classification
  - SST-2 : Movie Review 문장의 Sentiment Classification (Positive/Negative)

$$P_r(c|X) = \text{softmax}(\mathbf{W}_{SST}^\top \cdot \mathbf{x}), \quad (1)$$

class   Input Sentence      Task specific parameter      class Token

## MT-DNN Output

### Text Similarity

#### Text Similarity (BERT와 같은 구조)

- STS-B : 문장 쌍의 Similarity를 점수로 Regression
- a real value of the range  $(-\infty, \infty)$

$$\text{Sim}(\underline{X_1}, \underline{X_2}) = \underline{\mathbf{w}_{STS}^\top} \cdot \underline{\mathbf{x}}, \quad (2)$$

Input Sentence pair      Task specific class Token parameter

## MT-DNN Output

### Pairwise Text Classification

#### Pairwise Text Classification (BERT에서 구조 바뀐 부분)

- **NMLI** : 문장 쌍의 entailment, contradiction, neutral을 classification
- **RTE** : 문장 쌍의 Entailment 여부를 Binary Classification
- **QQP, MRPC** : 문장 쌍의 의미가 같은지 여부를 Classification

#### SAN(Stochastic Answer Network)를 활용한 Classification

- **Multi-step Reasoning** : RNN을 활용해 2번 이상의 예측을 통해 문장 간 관계 분류
- Ex. 아래 두개의 문장이 있을 때 한번에 문장 간 관계를 분류하기는 어려움
  - If you need this book, it is probably too late unless you are about to take an SAT or GRE
  - It's never too late, unless you're about to take a test
  - → 두 문장이 Contraction이라고 분류되기 위해서는 SAT, GRE가 시험이라는 것을 예측하는 단계가 선행되어야 함

# MT-DNN Output

## Pairwise Text Classification

RNN을 통해 Hypothesis 문장과 Premise 문장 간 관계를 예측

- **Hidden State** : Hypothesis Token Vector들의 Weighted Sum
- **Input** : Premise Token Vector들의 Weighted Sum (이전 hidden state를 활용해 weight 계산)

ning, the initial state  $s^0$  is the summary of  $\mathbf{M}^h$ :

$$\mathbf{s}^0 = \sum_j \alpha_j \mathbf{M}_j^h, \text{ where } \alpha_j = \frac{\exp(\mathbf{w}_1^\top \cdot \mathbf{M}_j^h)}{\sum_i \exp(\mathbf{w}_1^\top \cdot \mathbf{M}_i^h)}.$$

At time step  $k$  in the range of  $\{1, 2, \dots, K - 1\}$ ,

Parameter 줘서  
Attention 같은  
느낌으로 계산

the state is defined by  $\mathbf{s}^k = \text{GRU}(\mathbf{s}^{k-1}, \mathbf{x}^k)$ .

GRU cell이용  
문장 임베딩  
값 계산

Here,  $\mathbf{x}^k$  is computed from the previous state  $\mathbf{s}^{k-1}$

and memory  $\mathbf{M}^p$ :  $\mathbf{x}^k = \sum_j \beta_j \mathbf{M}_j^p$  and  $\beta_j =$

Premise  
Sentence  
Token의 vector

$\text{softmax}(\mathbf{s}^{k-1} \mathbf{W}_2^\top \mathbf{M}^p)$ . A one-layer classifier is

학습 웨이트 벡터(W),  
이전 히든 스테이트(s) 참고

# MT-DNN Output

## Pairwise Text Classification

Time Step마다 hidden state와 input을 각 문장의 vector로 활용하여 문장 간 관계 분류

- 문장 간 관계를 예측하기 위해 다음 수식처럼 Heuristic한 vector를 구성해서 예측
- 문장 간 거리, Similarity(dot-product)를 문장 간 관계를 나타내는 값으로 활용 (추가 feature)

$$P_r^k = \text{softmax}(\mathbf{W}_3^\top [\underbrace{\mathbf{s}^k; \mathbf{x}^k}_{\text{각 문장의 vector (concat)}}; \underbrace{|\mathbf{s}^k - \mathbf{x}^k|}_{\text{문장 간 거리}}; \underbrace{\mathbf{s}^k \cdot \mathbf{x}^k}_{\text{문장 간 Similarity (dot-product)}}]).$$

K Step 예측 후 평균 값을 최종 예측 값으로 사용

$$P_r = \text{avg}([P_r^0, P_r^1, \dots, P_r^{K-1}]). \quad (4)$$

## MT-DNN Output

### Relevance Ranking

#### Relevance Ranking (BERT 구조에서 바뀐 부분)

- QNLI : (질문, 지문)쌍에서 지문에 질문에 대한 답이 있는지 여부를 Binary Classification
- Sigmoid function을 통해 모든 answer candidate 문장을 scoring, Ranking을 통해 1개 문장만 정답으로 분류

$$\underline{\text{Rel}}(\underline{Q}, \underline{A}) = \underline{g}(\underline{\mathbf{w}}_{QNLI}^\top \cdot \underline{\mathbf{x}}), \quad (5)$$

Input Sentence pair      sigmoid      Task specific class Token parameter

# Experiments

# Experiments

## Training Procedure

- 9개의 GLUE Task Dataset을 모아서 Training dataset 구성
- 같은 Task로 이루어지도록 batch를 뽑아서
- Batch 단위로 학습 마다 Shared Layer와 해당 Task specific Layer 학습

$$-\sum_c \mathbb{1}(X, c) \log(P_r(c|X)), \quad (6)$$

$$(y - \text{Sim}(X_1, X_2))^2, \quad (7)$$

$$-\sum_{(Q, A^+)} P_r(A^+|Q), \quad (8)$$

$$P_r(\underline{A^+}|Q) = \frac{\exp(\gamma \text{Rel}(Q, A^+))}{\sum_{A' \in \mathcal{A}} \exp(\gamma \text{Rel}(Q, A'))}, \quad (9)$$

Answer가 포함된 문장

---

**Algorithm 1:** Training a MT-DNN model.

---

```

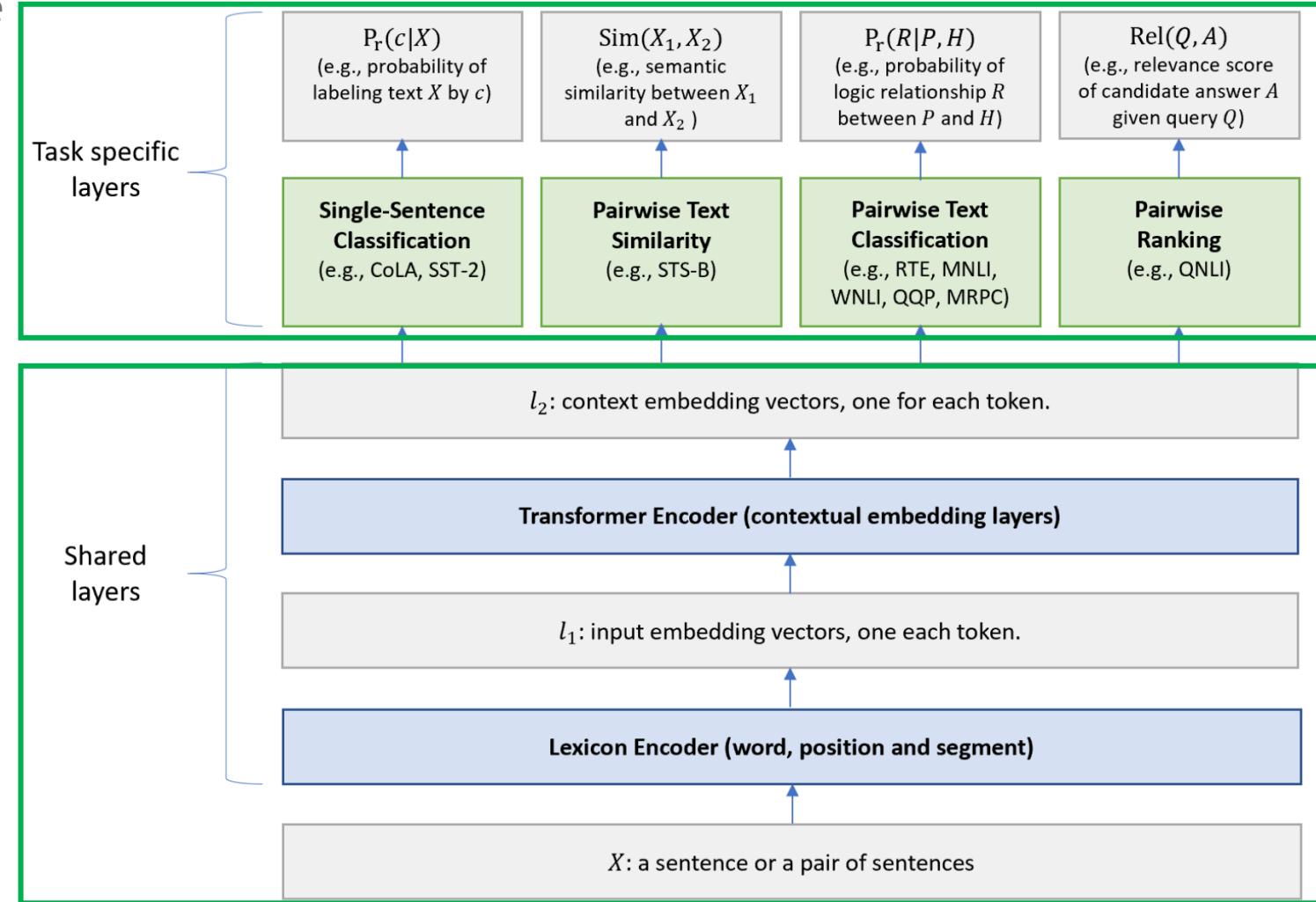
Initialize model parameters  $\Theta$  randomly.
Pre-train the shared layers (i.e., the lexicon
encoder and the transformer encoder).
Set the max number of epoch:  $epoch_{max}$ .
//Prepare the data for  $T$  tasks.
for  $t$  in  $1, 2, \dots, T$  do
| Pack the dataset  $t$  into mini-batch:  $D_t$ .
end
for epoch in  $1, 2, \dots, epoch_{max}$  do
  1. Merge all the datasets:
      $D = D_1 \cup D_2 \dots \cup D_T$ 
  2. Shuffle  $D$ 
  for  $b_t$  in  $D$  do
    // $b_t$  is a mini-batch of task  $t$ .
    3. Compute loss :  $L(\Theta)$ 
        $L(\Theta) = \text{Eq. 6}$  for classification
        $L(\Theta) = \text{Eq. 7}$  for regression
        $L(\Theta) = \text{Eq. 8}$  for ranking
    4. Compute gradient:  $\nabla(\Theta)$ 
    5. Update model:  $\Theta = \Theta - \epsilon \nabla(\Theta)$ 
  end
end

```

---

# Experiments

## Training Procedure



여기는 task specific  
하게 학습되고

여기는 모든 task에  
계속 해서 학습되고

# Experiments

## Testing set 평가 결과

- BERT Large로 학습 후 GLUE Task 9개로 fine tuning
- BERT (80.5) → MT-DNN (82.7) : 2.2% 향상

<b>Model</b>	Single Sentence Classification	Pairwise Text Classification	Text Similarity Regression	Pairwise Text Classification		Relevance Ranking	Pairwise Text Classification		AX	<b>Score</b>
	CoLA 8.5k	SST-2 67k	MRPC 3.7k	STS-B 7k	QQP 364k	MNLI-m/mm 393k	QNLI 108k	RTE 2.5k		
BiLSTM+ELMo+Attn <sup>1</sup>	36.0	90.4	84.9/77.9	75.1/73.3	64.8/84.7	76.4/76.1	-	56.8	65.1	26.5
Singletask Pretrain Transformer <sup>2</sup>	45.4	91.3	82.3/75.7	82.0/80.0	70.3/88.5	82.1/81.4	-	56.0	53.4	29.8
GPT on STILTs <sup>3</sup>	47.2	93.1	87.7/83.7	85.3/84.8	70.1/88.1	80.8/80.6	-	69.1	65.1	29.4
BERT <sup>4</sup> LARGE	60.5	94.9	89.3/85.4	87.6/86.5	72.1/89.3	86.7/85.9	92.7	70.1	65.1	39.6
MT-DNN <sub>no-fine-tune</sub>	58.9	94.6	90.1/86.4	89.5/88.8	72.7/89.6	86.5/85.8	93.1	79.1	65.1	39.4
MT-DNN	<b>62.5</b>	<b>95.6</b>	<b>91.1/88.2</b>	<b>89.5/88.8</b>	<b>72.7/89.6</b>	<b>86.7/86.0</b>	<b>93.1</b>	<b>81.4</b>	65.1	<b>40.3</b>
Human Performance	66.4	97.8	86.3/80.8	92.7/92.6	59.5/80.4	92.0/92.8	91.2	93.6	95.9	-

Semantically Equivalent Problem      NLI Problem

### Multi-Task Learning의 기본 Idea!

학습한 내용이 Task간에 서로 영향을 주게 되니까(가정)

여러 Task를 한꺼번에 학습하면 LM을 더 잘 학습 시킬 수 있지 않을까?

## Experiments

### Testing set 평가 결과

- ST-DNN : Multi Task Learning을 하지 않고 Task Specific Layer만 바꿔서 (SAN, Pairwise Ranking loss) 학습한 모델
  - SAN을 사용하여 얻은 성능 향상 효과가 크진 않음
  - Class Token을 Token Embedding을 썼다는 것에 의미가 있음
  - QNLI는 Ranking Approach로 큰 성능 향상

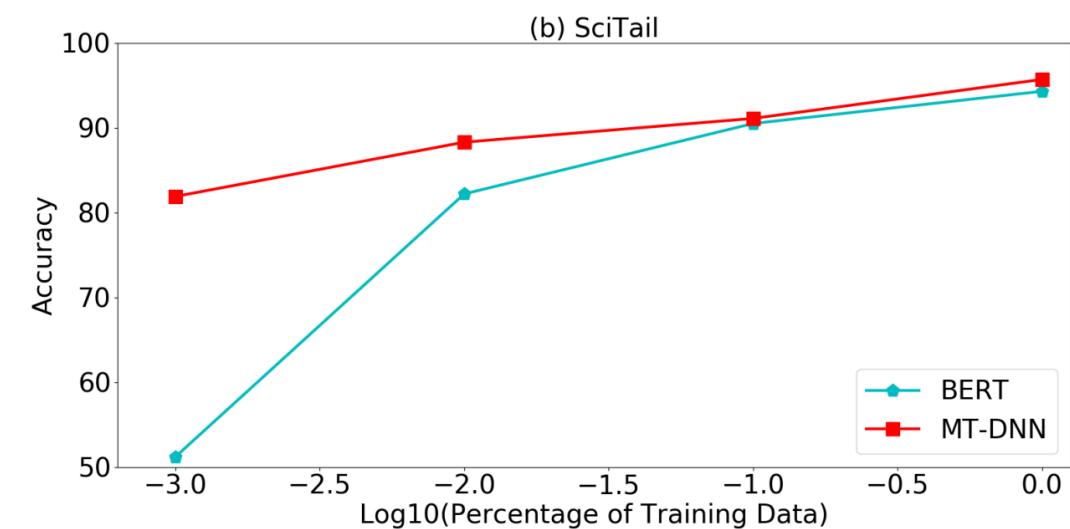
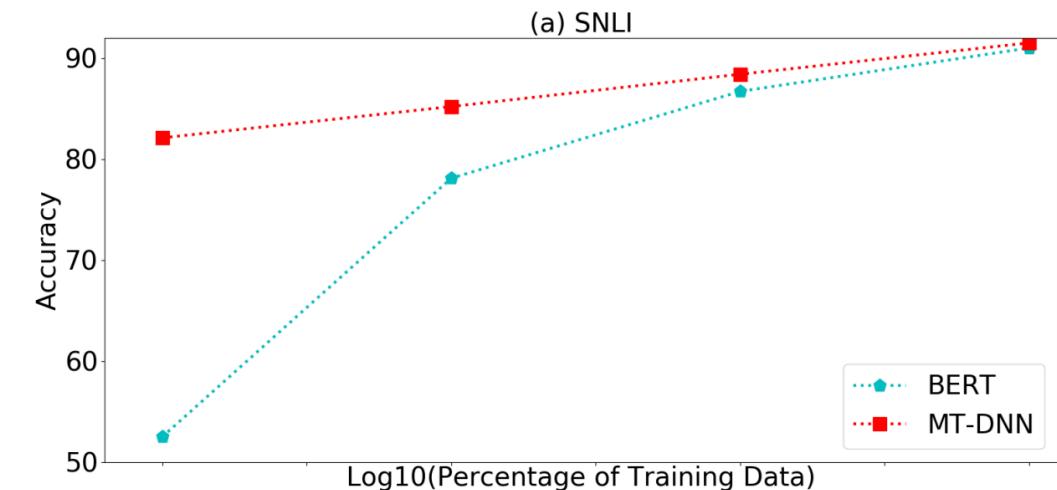
Model	MNLI-m/mm	QQP	RTE	QNLI (v1/v2)	MRPC	CoLa	SST-2	STS-B
BERT <sub>LARGE</sub>	86.3/86.2	91.1/88.0	71.1	90.5/92.4	89.5/85.8	61.8	93.5	89.6/89.3
ST-DNN	86.6/86.3	91.3/88.4	72.0	96.1/-	89.7/86.4	-	-	-
MT-DNN	<b>87.1/86.7</b>	<b>91.9/89.2</b>	<b>83.4</b>	<b>97.4/92.9</b>	<b>91.0/87.5</b>	<b>63.5</b>	<b>94.3</b>	<b>90.7/90.6</b>

## Experiments

### Domain Adaptation

- BERT, MT-DNN에 각각 새로운 Task Specific Layer 붙여서 학습
  - NLI Dataset(SNLI, SciTail) 활용해 Domain adaptation 평가
- 데이터가 적을 때 MT-DNN의 성능이 BERT보다 훨씬 높음!

## 4.4 Domain Adaptation Results on SNLI and SciTail



Thank you.

# Appendix

## Appendix

Corpus	Task	#Train	#Dev	#Test	#Label	Metrics
<b>Single-Sentence Classification (GLUE)</b>						
CoLA	Acceptability	8.5k	1k	1k	2	Matthews corr
SST-2	Sentiment	67k	872	1.8k	2	Accuracy
<b>Pairwise Text Classification (GLUE)</b>						
MNLI	NLI	393k	20k	20k	3	Accuracy
RTE	NLI	2.5k	276	3k	2	Accuracy
WNLI	NLI	634	71	146	2	Accuracy
QQP	Paraphrase	364k	40k	391k	2	Accuracy/F1
MRPC	Paraphrase	3.7k	408	1.7k	2	Accuracy/F1
<b>Text Similarity (GLUE)</b>						
STS-B	Similarity	7k	1.5k	1.4k	1	Pearson/Spearman corr
<b>Relevance Ranking (GLUE)</b>						
QNLI	QA/NLI	108k	5.7k	5.7k	2	Accuracy
<b>Pairwise Text Classification</b>						
SNLI	NLI	549k	9.8k	9.8k	3	Accuracy
SciTail	NLI	23.5k	1.3k	2.1k	2	Accuracy

Table 1: Summary of the three benchmarks: GLUE, SNLI and SciTail.

## Appendix

### GLUE MNLI Example

### MNLI : Multi-Genre NLI Corpus

## Examples

Premise	Label	Hypothesis
<b>Fiction</b>  The Old One always comforted Ca'daan, except today.	neutral	Ca'daan knew the Old One very well.
<b>Letters</b>  Your gift is appreciated by each and every student who will benefit from your generosity.	neutral	Hundreds of students will benefit from your generosity.
<b>Telephone Speech</b>  yes now you know if if everybody like in August when everybody's on vacation or something we can dress a little more casual or	contradiction	August is a black out month for vacations in the company.
<b>9/11 Report</b>  At the other end of Pennsylvania Avenue, people began to line up for a White House tour.	entailment	People formed a line at the end of Pennsylvania Avenue.

# Appendix

## Transformer Architecture

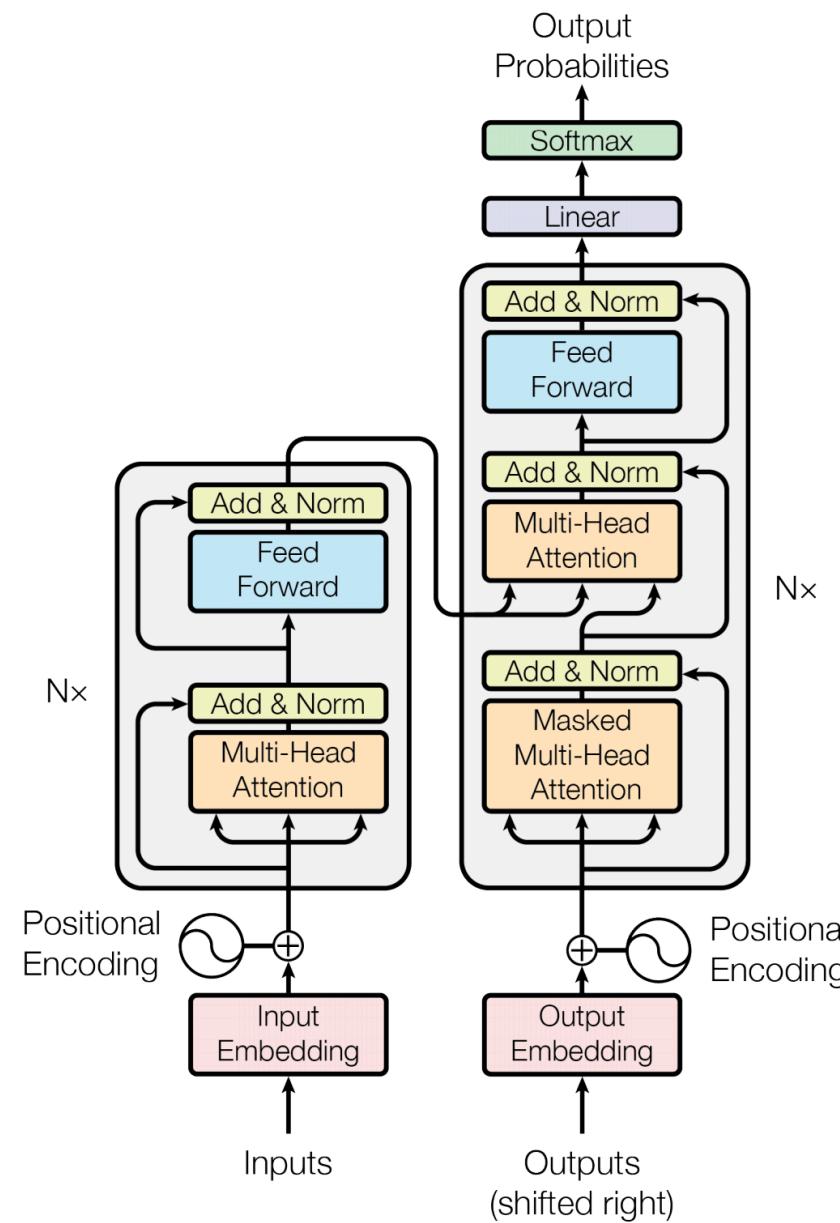


Figure 1: The Transformer - model architecture.

## Appendix references

### Paper

MT-DNN : <https://arxiv.org/pdf/1901.11504.pdf>

BERT : <https://arxiv.org/pdf/1810.04805.pdf>

Attention is All You Need : <https://arxiv.org/pdf/1706.03762.pdf>

### Other

티맥스 데이터 AI 논문 리뷰 자료