XLNet: Generalized Autoregressive Pretraining for Language Understanding

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, Quoc V.Le

Carnegie Mellon University, Google Brain

Presented by
Junho Lee, AI LAB Kor
saitros2@gmail.com

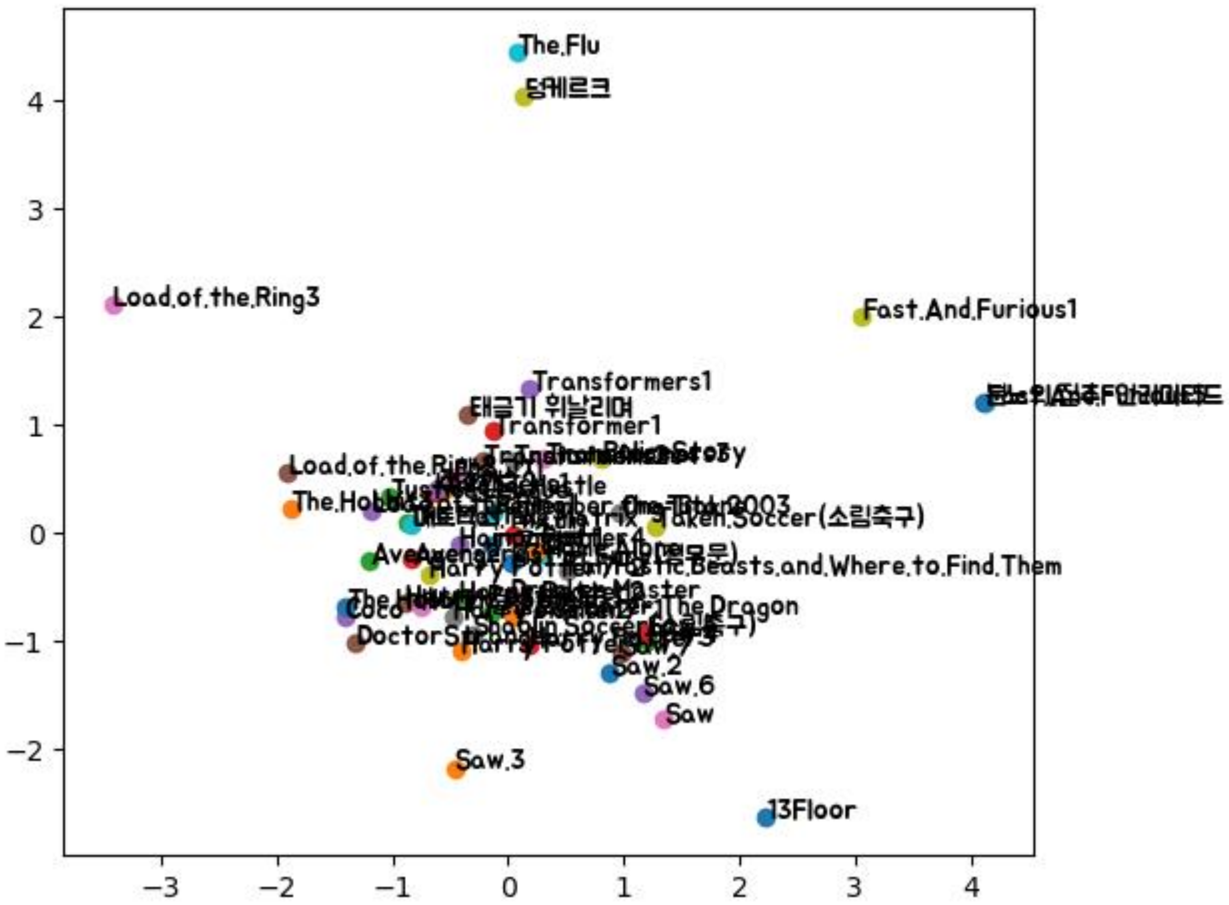
소개

이번 강의는 한국 인공지능 연구소 C’est La Vie 랩에서
최근 발표된 XLNet paper를 이해할 수 있도록 공부한
내용을 기반으로 합니다.

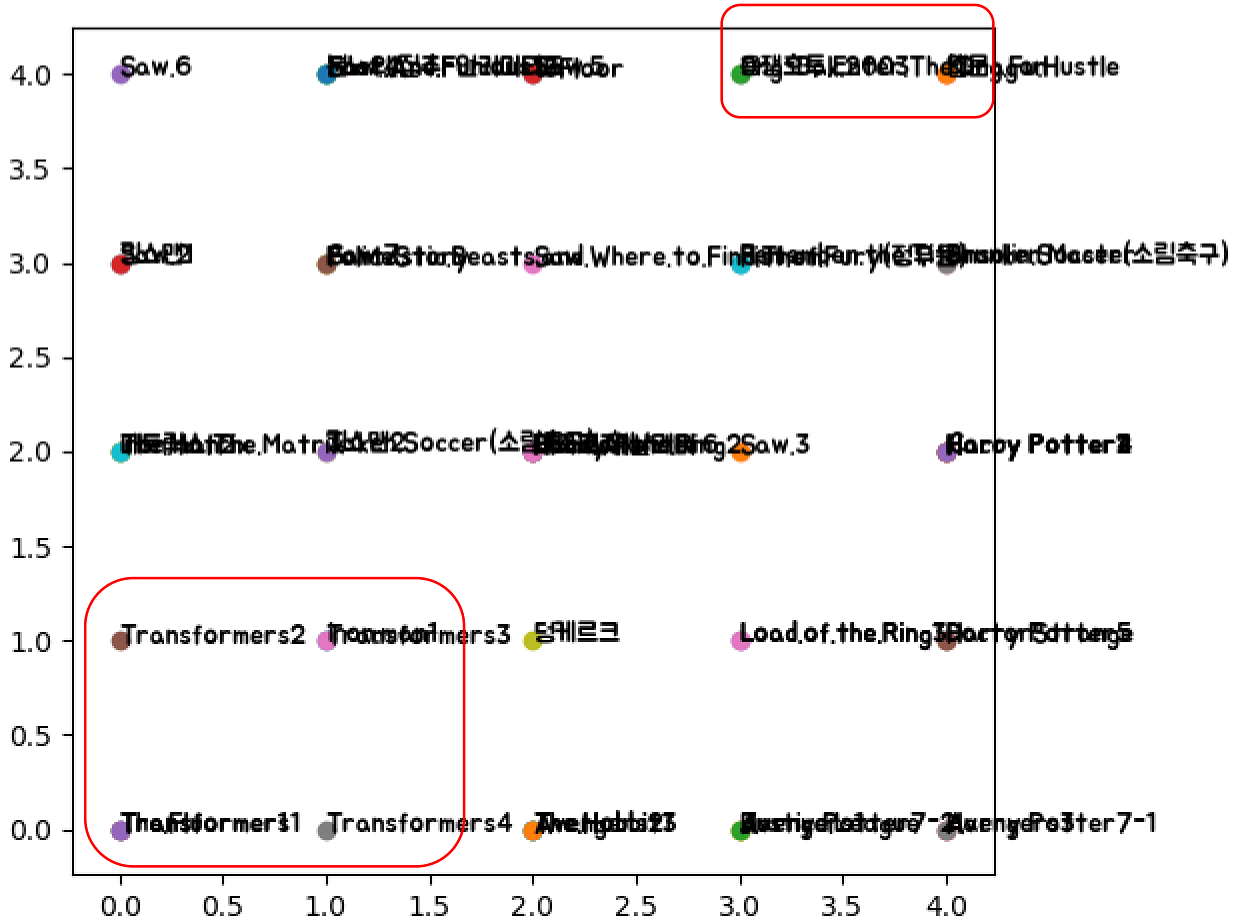
다루는 내용은 XLNet의 Core Idea 를 기준으로 합니다.

별다른 출처가 표시되어있지 않은 그림 및 자료는
원문 https://arxiv.org/pdf/1906.08237.pdf 을 참고하였습니다.

소개

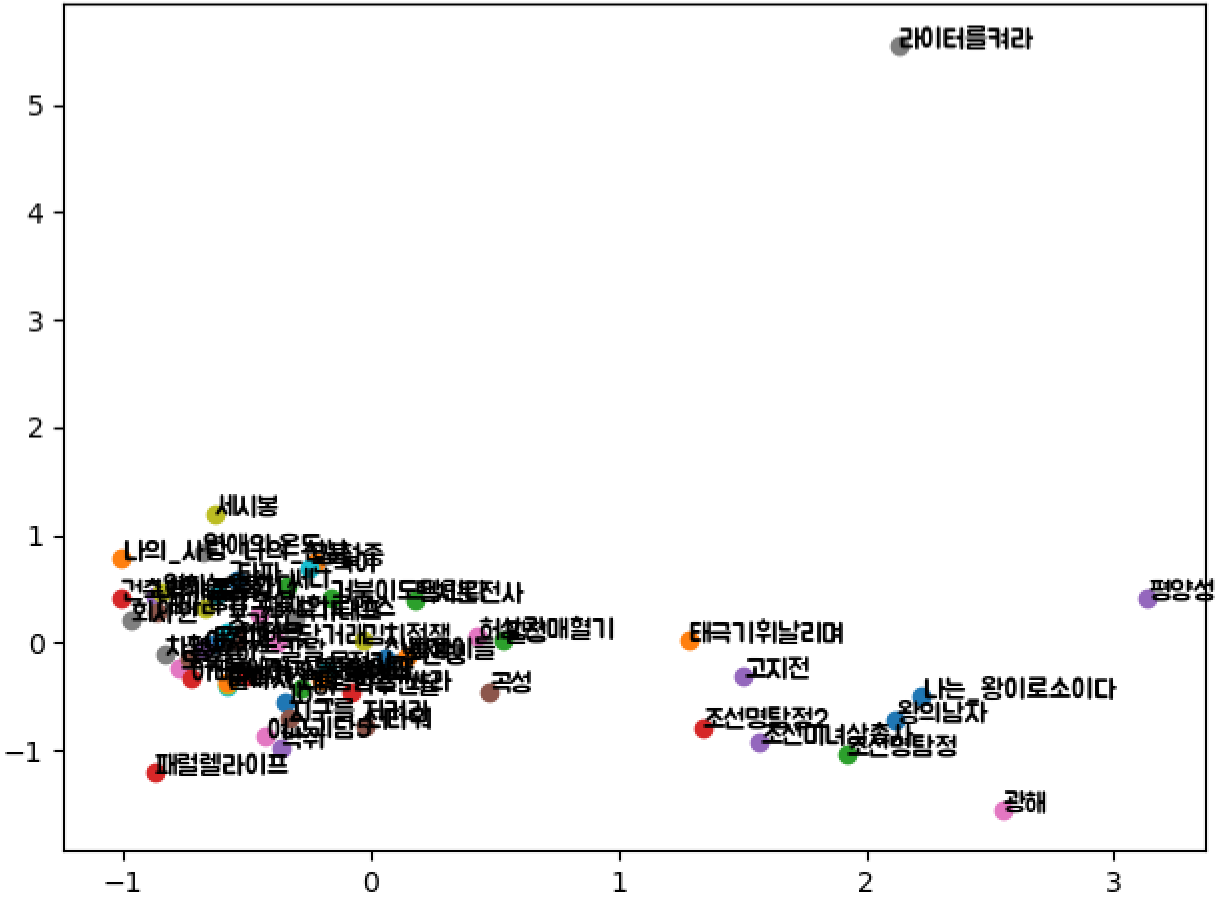


영화 자막 PCA(2D)

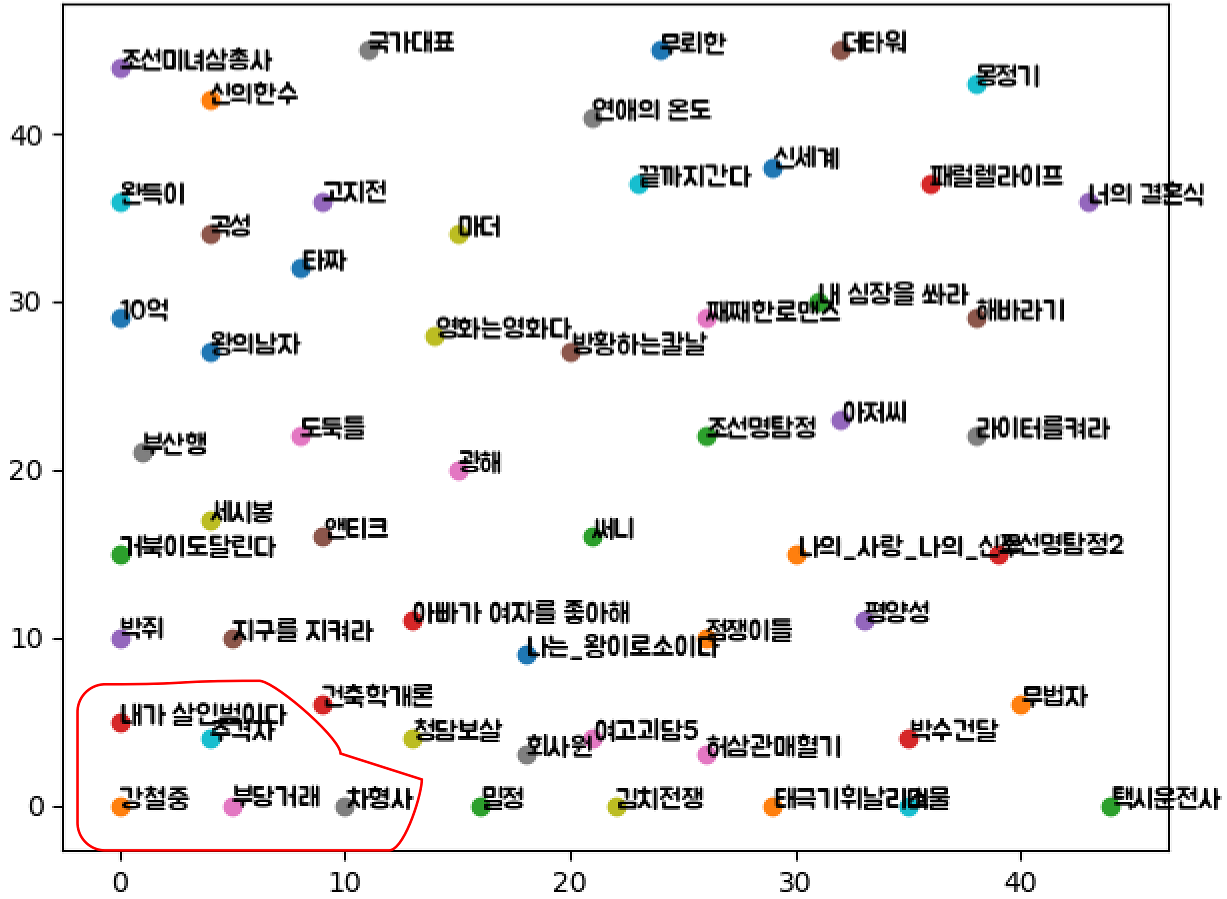


영화 자막 SOM (10x10)

소개



영화 시나리오 PCA(2D)



영화 시나리오 SOM(50x50)

Overview

- NLP Trends

- Autoregressive vs Autoencoding

- XLNet
  - Permutation Language model
  - Two-stream self-attention mechanism
  - Recurrence mechanism

Overview

- NLP Trends

- Autoregressive vs Autoencoding

- XLNet
    - Permutation Language model
    - Two-stream self-attention mechanism
    - Recurrence mechanism

2018 NLP

SESAME STREET

ELMo
2018년 3월

BERT
2018년 10월

BigBird
2018년 12월

2018 NLP

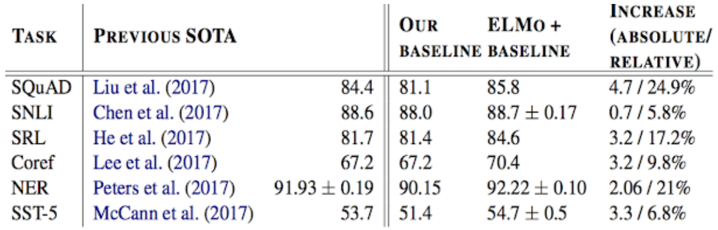
ELMo

- 풀 네임 : Embeddings from Language Models
- 등 장 : 2018년 3월

- 특 징 :
  1. Context 정보를 반영하는 방법 제시

  2. 같은 성과를 위한 필요 데이터를 감소

  3. 추가적인 성능 향상



TASK	PREVIOUS SOTA		OUR BASELINE	ELMO + BASELINE	INCREASE (ABSOLUTE/ RELATIVE)
SQuAD	Liu et al. (2017)	84.4	81.1	85.8	4.7 / 24.9%
SNLI	Chen et al. (2017)	88.6	88.0	88.7 ± 0.17	0.7 / 5.8%
SRL	He et al. (2017)	81.7	81.4	84.6	3.2 / 17.2%
Coref	Lee et al. (2017)	67.2	67.2	70.4	3.2 / 9.8%
NER	Peters et al. (2017)	91.93 ± 0.19	90.15	92.22 ± 0.10	2.06 / 21%
SST-5	McCann et al. (2017)	53.7	51.4	54.7 ± 0.5	3.3 / 6.8%

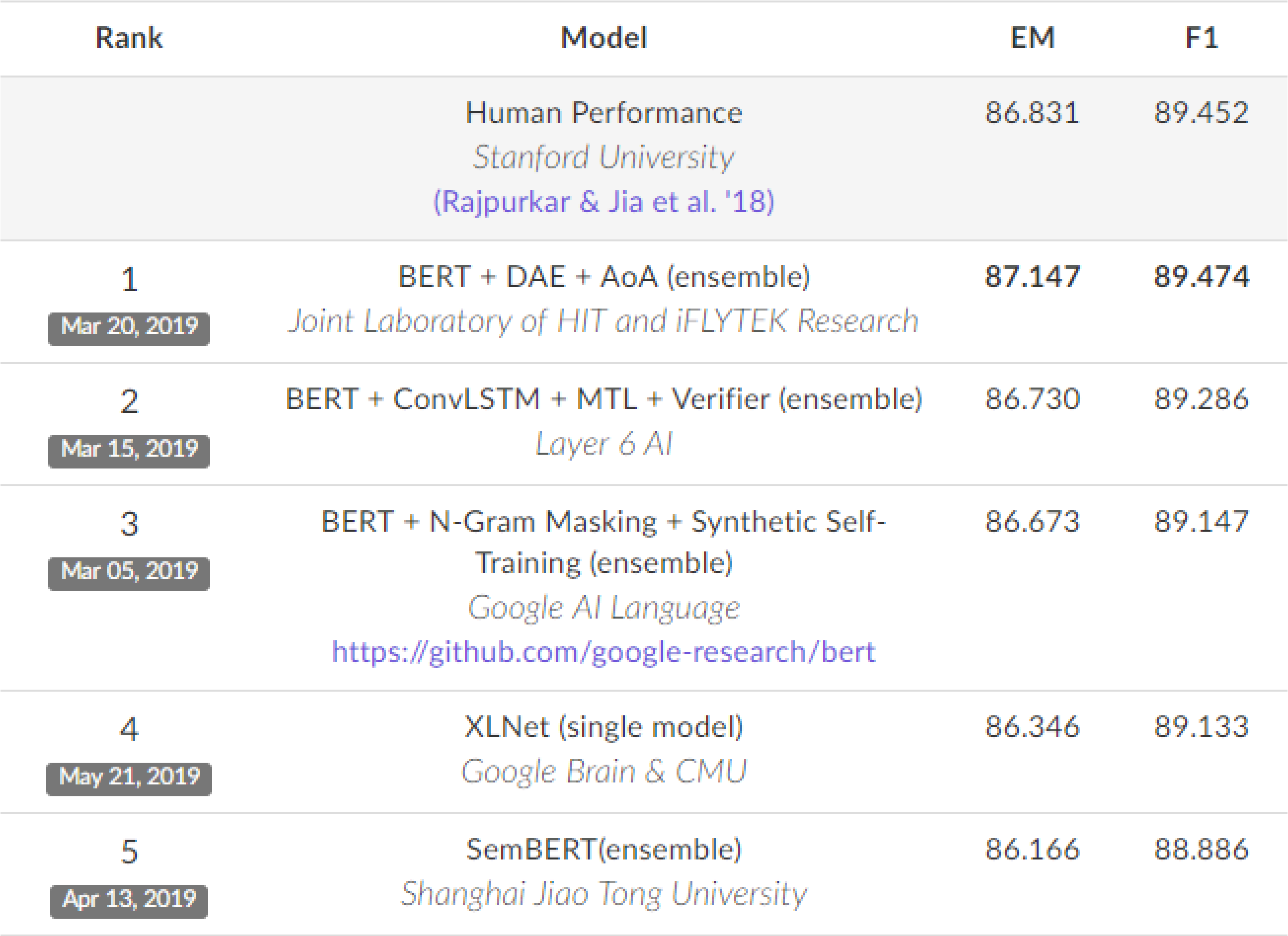
SQuAD : Q&A
SNLI(Textual entailment) : 두 문장 간의 상관 관계를 판별
Semantic Role Labeling : 행위주, 피동작주, 원인, 목적, 시제 등을 판별
Coreference resulotion : 같은 개체를 의미하는 것을 묶는 과제
Named Entity Recognition : 개체명 인식, 인물, 날짜, 동물, 의학 용어 등
SST-5 : 감정 분석

2018 NLP

BERT

- 풀 네임 : Bidirectional Encoder Representations from Transformers
- 등 장 : 2018년 10월
- 특 징 :
  1. 11월에 pretrain data를 오픈소스로 공개
  2. 가장 치열한 분야인 SQuAD에서 1위 등극
  3. 무려 11개 NLP Task에서 SOTA 달성
  4. BERT를 활용한 모델이 계속 등장
     (Bert for multi-label classification,
     Bert for History Answer Embedding,
     Bert for Re-Rank 등등…)

2018 NLP

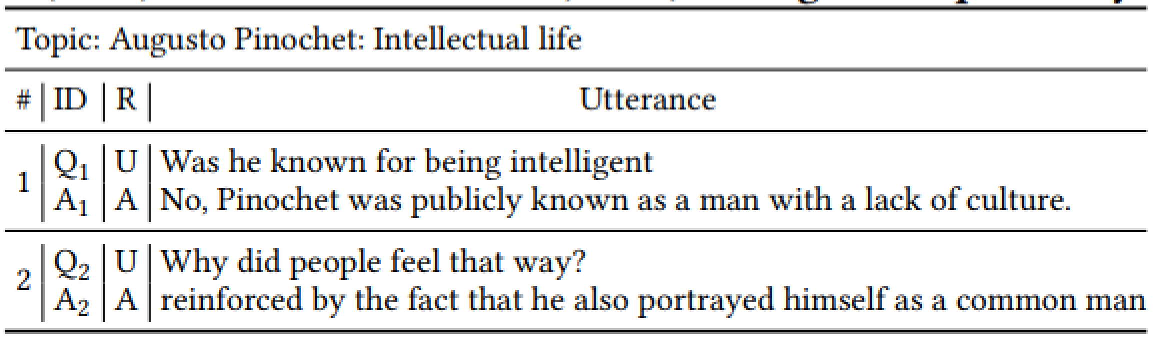


Rank	Model	EM	F1
	Human Performance *Stanford University* (Rajpurkar & Jia et al. '18)	86.831	89.452
1 Mar 20, 2019	BERT + DAE + AoA (ensemble) *Joint Laboratory of HIT and iFLYTEK Research*	**87.147**	**89.474**
2 Mar 15, 2019	BERT + ConvLSTM + MTL + Verifier (ensemble) *Layer 6 AI*	86.730	89.286
3 Mar 05, 2019	BERT + N-Gram Masking + Synthetic Self-Training (ensemble) *Google AI Language* https://github.com/google-research/bert	86.673	89.147
4 May 21, 2019	XLNet (single model) *Google Brain & CMU*	86.346	89.133
5 Apr 13, 2019	SemBERT(ensemble) *Shanghai Jiao Tong University*	86.166	88.886

2018 NLP

- 기존 BERT Embedding에 HAE 추가
- HAE 가 기존 BERT의 token 정보를 수정
  BERT가 대화의 history를 자연스럽게 수용
- Fine tuning을 통해서 specific 하게 학습을 진행
- 논문은 Bert-Base (Uncased) model이 사용
  max sequence length set 384
  batch size set 12
  learning rate set 3e-5
  stride set 128
  max question length set 64
  max answer length set 30

Table 1: A part of an information-seeking dialog from QuAC. "R", "U", and "A" denotes role, user, and agent respectively.



Topic: Augusto Pinochet: Intellectual life

#	ID	R	Utterance
1	Q_1	U	Was he known for being intelligent
	A_1	A	No, Pinochet was publicly known as a man with a lack of culture.
2	Q_2	U	Why did people feel that way?
	A_2	A	reinforced by the fact that he also portrayed himself as a common man

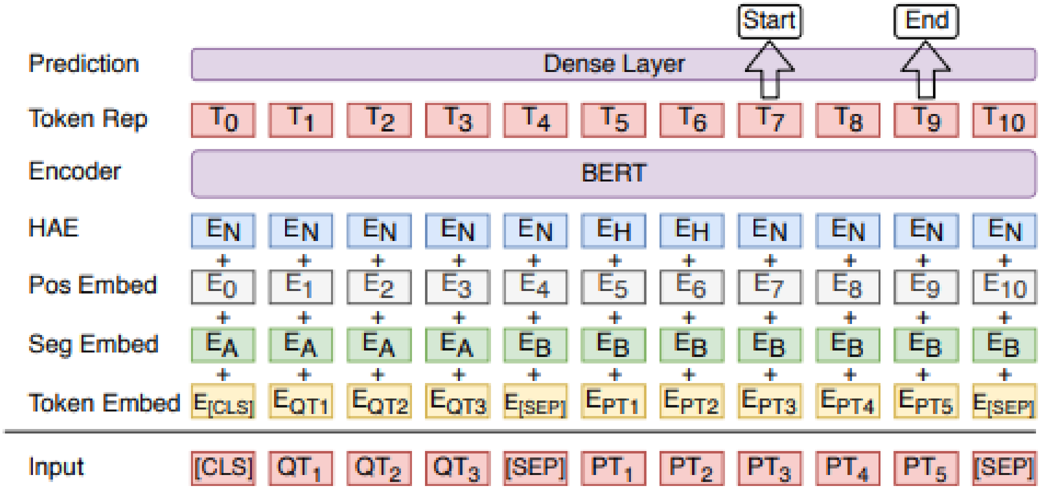

Start
End
Prediction
Dense Layer
Token Rep
T_0 T_1 T_2 T_3 T_4 T_5 T_6 T_7 T_8 T_9 T_{10}
Encoder
BERT
HAE
E_N E_N E_N E_N E_N E_H E_H E_N E_N E_N E_N
Pos Embed
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}
Seg Embed
E_A E_A E_A E_A E_B E_B E_B E_B E_B E_B E_B
Token Embed
$E_{[CLS]}$ E_{QT1} E_{QT2} E_{QT3} $E_{[SEP]}$ E_{PT1} E_{PT2} E_{PT3} E_{PT4} E_{PT5} $E_{[SEP]}$
Input
[CLS] QT_1 QT_2 QT_3 [SEP] PT_1 PT_2 PT_3 PT_4 PT_5 [SEP]

Figure 2: Architecture of the ConvQA model with HAE. E_H/E_N in HAE denote the token is in/not in history answers.

2018 NLP

- 기존 BERT Embedding에 HAE 추가
- HAE 가 기존 BERT의 token 정보를 수정
  BERT가 대화의 history를 자연스럽게 수용
- Fine tuning을 통해서 specific 하게 학습을 진행
- 논문은 Bert-Base (Uncased) model*을 사용
  max sequence length set 384
  batch size set 12
  learning rate set 3e-5
  stride set 128
  max question length set 64
  max answer length set 30

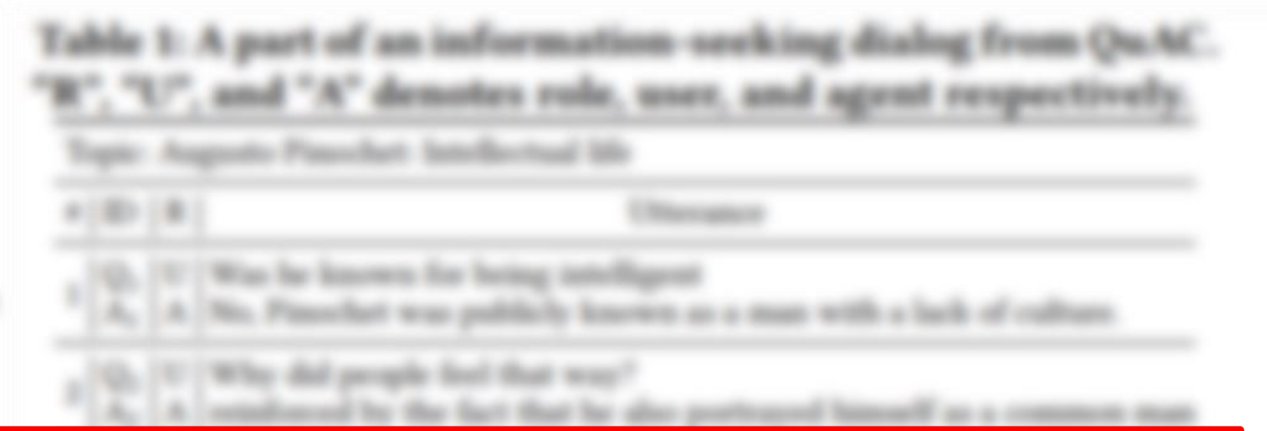


Table 1: A part of an information-seeking dialog from QuAC. "R", "U", and "A" denotes role, user, and agent respectively.

Topic: Augusto Pinochet: Intellectual life

#	ID	R	Utterance
1	Q_1	U	Was he known for being intelligent
	A_1	A	No, Pinochet was publicly known as a man with a lack of culture.
2	Q_2	U	Why did people feel that way?
	A_2	A	reinforced by the fact that he also portrayed himself as a common man

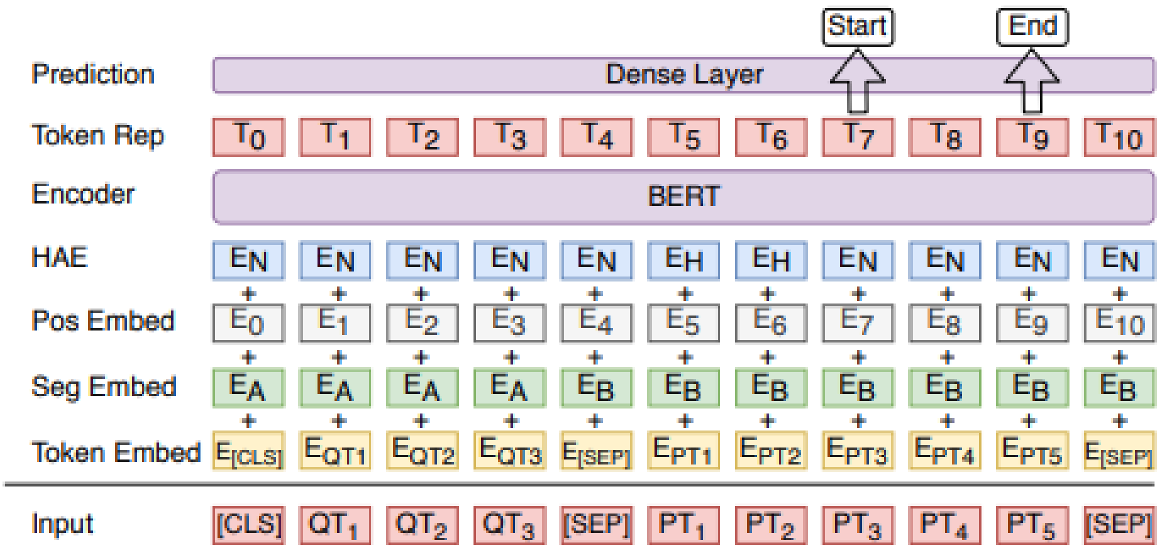
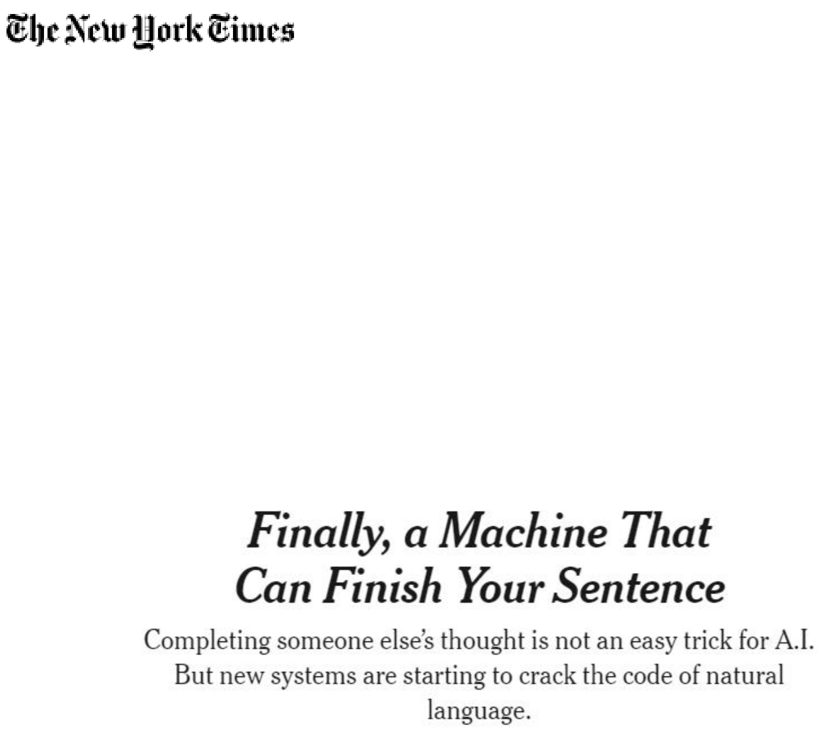

Start
End
Prediction
Dense Layer
Token Rep
T_0 T_1 T_2 T_3 T_4 T_5 T_6 T_7 T_8 T_9 T_{10}
Encoder
BERT
HAE
E_N E_N E_N E_N E_N E_H E_H E_N E_N E_N E_N
Pos Embed
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}
Seg Embed
E_A E_A E_A E_A E_B E_B E_B E_B E_B E_B E_B
Token Embed
$E_{[CLS]}$ E_{QT1} E_{QT2} E_{QT3} $E_{[SEP]}$ E_{PT1} E_{PT2} E_{PT3} E_{PT4} E_{PT5} $E_{[SEP]}$
Input
[CLS] QT_1 QT_2 QT_3 [SEP] PT_1 PT_2 PT_3 PT_4 PT_5 [SEP]

Figure 2: Architecture of the ConvQA model with HAE. E_H/E_N in HAE denote the token is in/not in history answers.

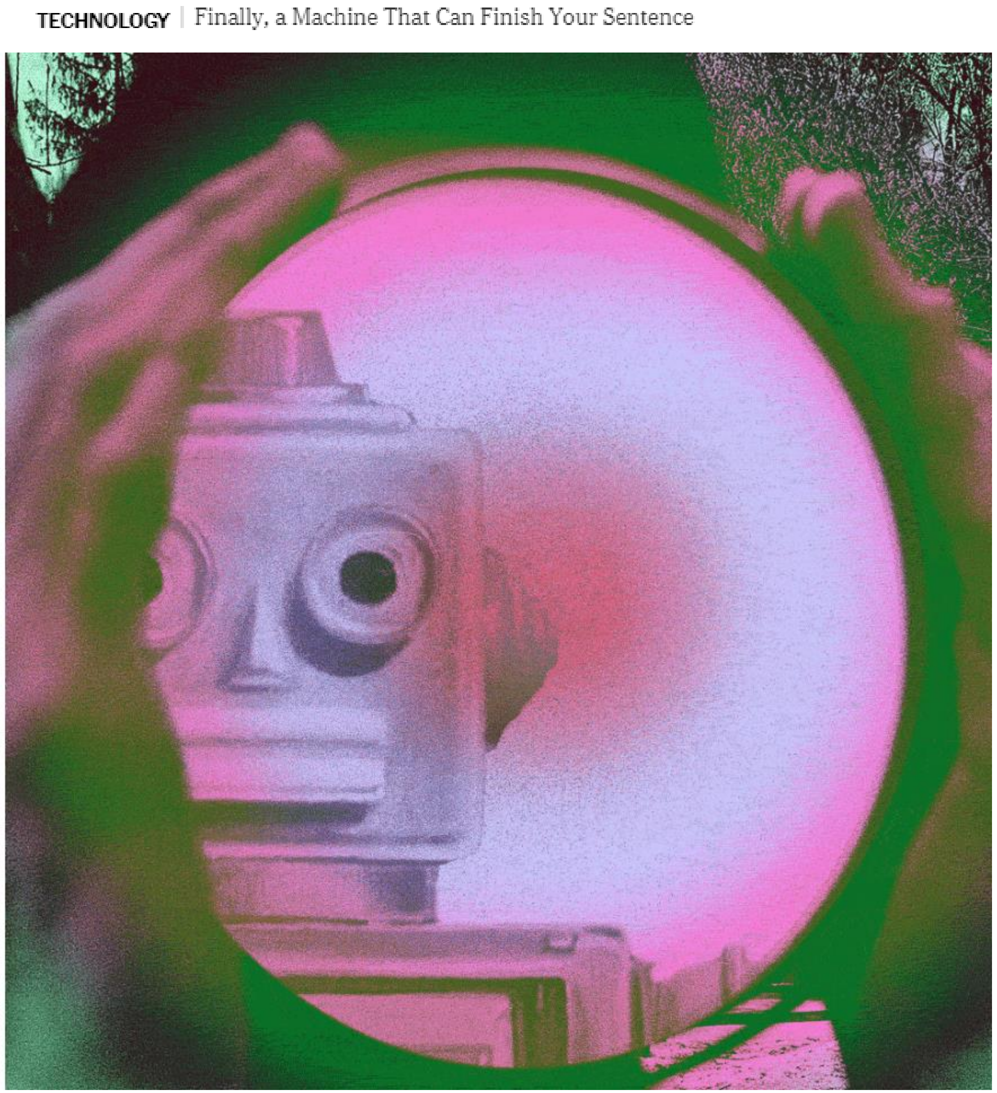
2018 NLP



The New York Times

Finally, a Machine That Can Finish Your Sentence

Completing someone else's thought is not an easy trick for A.I. But new systems are starting to crack the code of natural language.

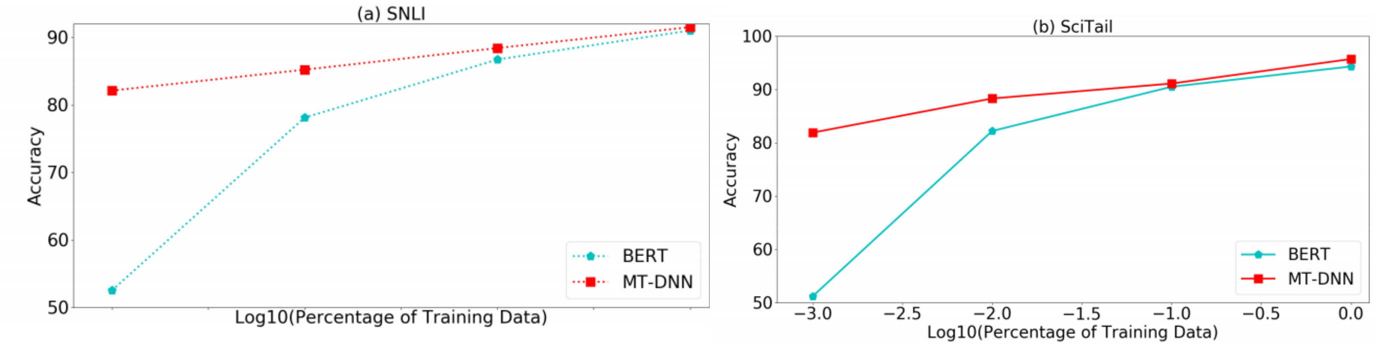


TECHNOLOGY | Finally, a Machine That Can Finish Your Sentence

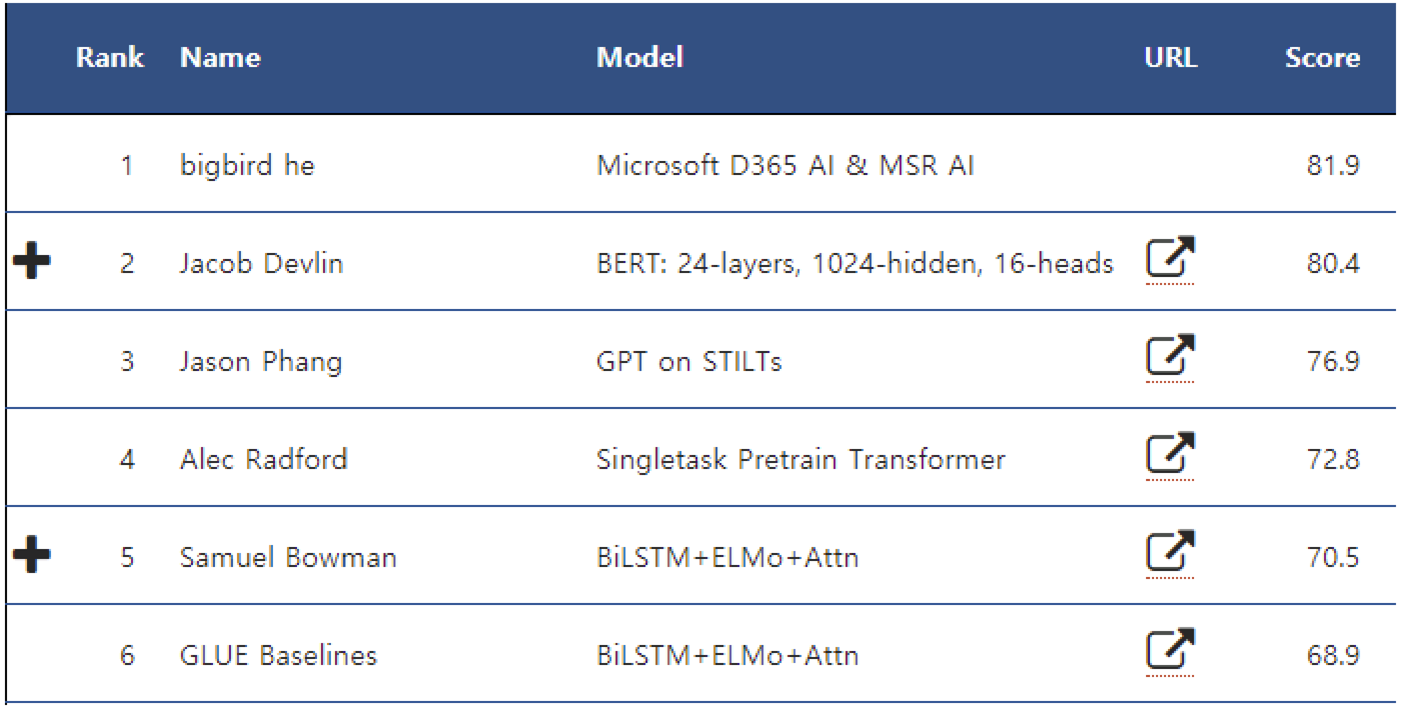
2018 NLP

Big Bird

- 실 명 : MT-DNN
- 등 장 : 2018년 12월
- 특 징 :
  1. 논문은 19년도 1월에 공개
  2. Elmo 와 BERT를 의식한 이름 선정
  3. 학습데이터가 적은 상황에서 좋은 성능

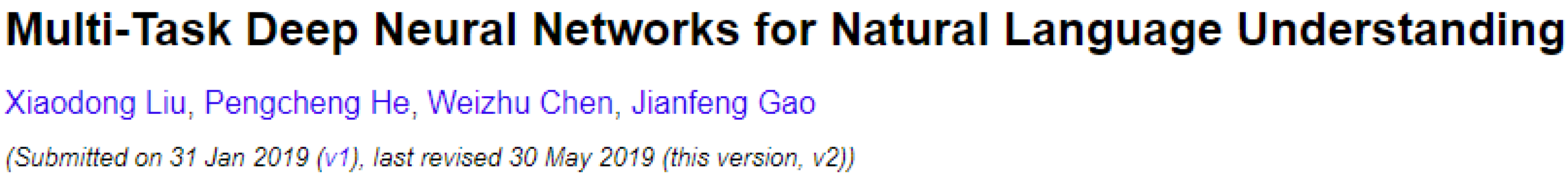


Bigbird



	Rank	Name	Model	URL	Score
	1	bigbird he	Microsoft D365 AI & MSR AI		81.9
+	2	Jacob Devlin	BERT: 24-layers, 1024-hidden, 16-heads	↗	80.4
	3	Jason Phang	GPT on STILTs	↗	76.9
	4	Alec Radford	Singletask Pretrain Transformer	↗	72.8
+	5	Samuel Bowman	BiLSTM+ELMo+Attn	↗	70.5
	6	GLUE Baselines	BiLSTM+ELMo+Attn	↗	68.9

GLUE Benchmark, 2018.12.27



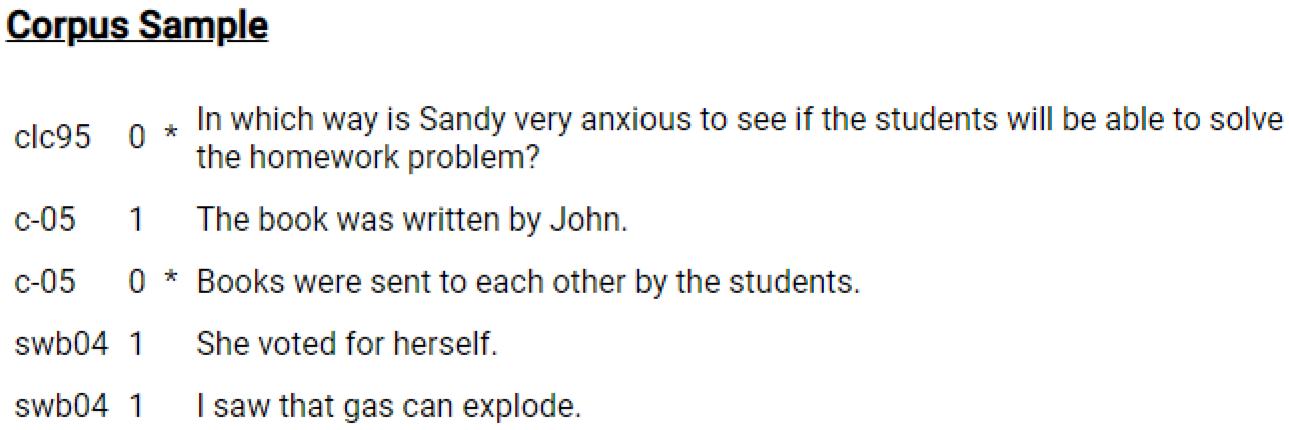
Multi-Task Deep Neural Networks for Natural Language Understanding

Xiaodong Liu, Pengcheng He, Weizhu Chen, Jianfeng Gao

(Submitted on 31 Jan 2019 (v1), last revised 30 May 2019 (this version, v2))

GLUE dataset

CoLA : The Corpus of Linguistic Acceptability (언어 적합성 판단 - 문법)



Corpus Sample

clc95	0	*	In which way is Sandy very anxious to see if the students will be able to solve the homework problem?
c-05	1		The book was written by John.
c-05	0	*	Books were sent to each other by the students.
swb04	1		She voted for herself.
swb04	1		I saw that gas can explode.

SST-2 : Stanford Sentiment Treebank (binary label classification problem)
-1 : (multi label classification problem)

MRPC : Microsoft Research Paraphrase Corpus (문장의 sentiment 가 같은지 판단)

STS-B : Semantic Textual Similarity Benchmark(문장의 유사도를 점수로 판단)

QQP : Quora Question Pair(두 질문이 같은 의미인지 파악)

SQuAD : Stanford Question Answering Database (질문을 보고 지문에서 정답 구간을 찾아낸다)

Unsupervised Learning

2018년 1월

1 GPU days

fast.ai

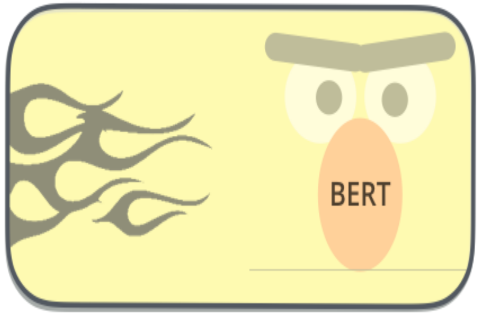
Making neural nets uncool again

GPT

2018년 6월

240 GPU days

OpenAI



2018년 10월

256 TPU days
320~560 GPU days

Google AI

GPT-2

2019년 2월

2048 TPU days

OpenAI

Large Batch

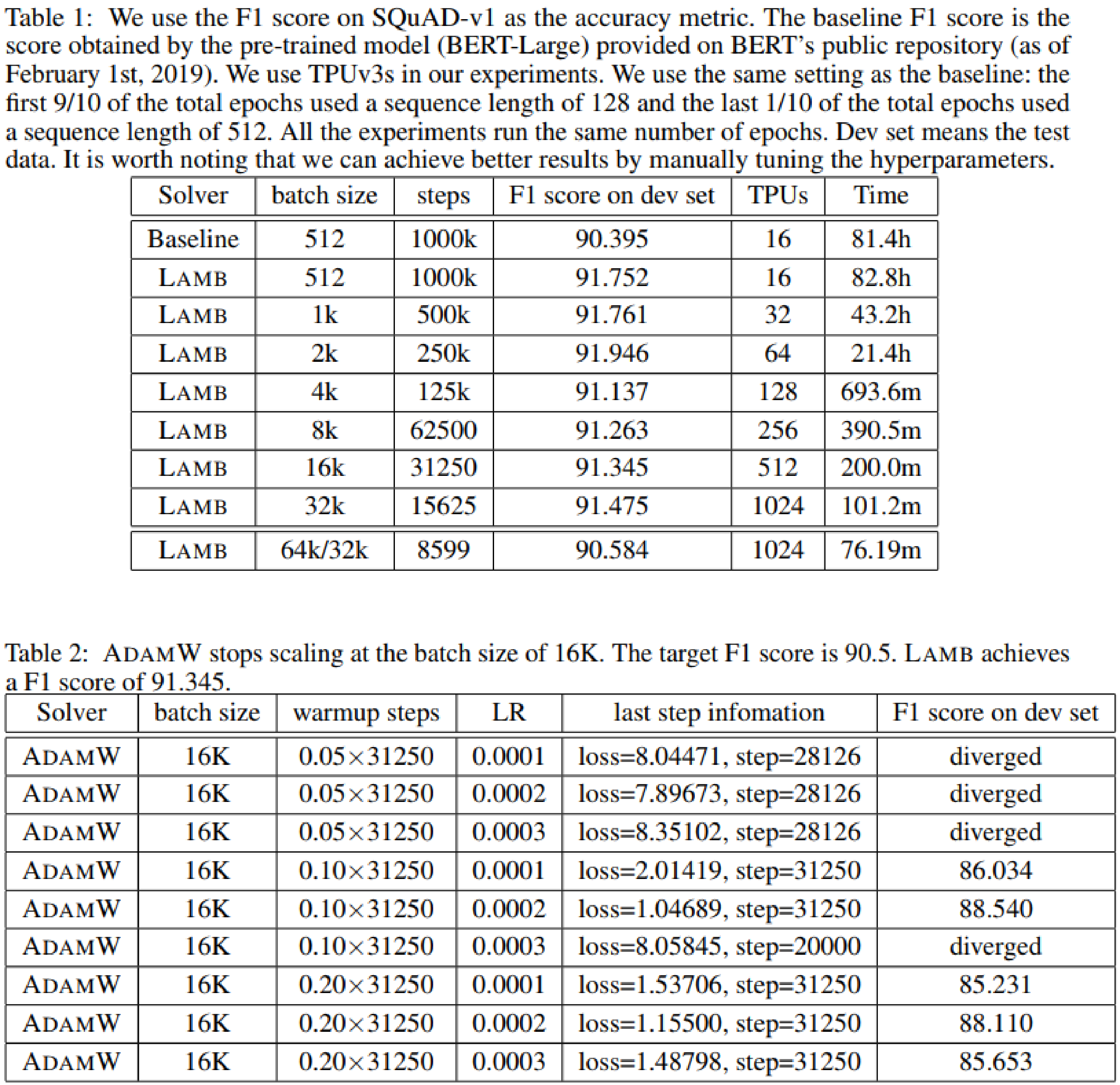


Table 1: We use the F1 score on SQuAD-v1 as the accuracy metric. The baseline F1 score is the score obtained by the pre-trained model (BERT-Large) provided on BERT's public repository (as of February 1st, 2019). We use TPUv3s in our experiments. We use the same setting as the baseline: the first 9/10 of the total epochs used a sequence length of 128 and the last 1/10 of the total epochs used a sequence length of 512. All the experiments run the same number of epochs. Dev set means the test data. It is worth noting that we can achieve better results by manually tuning the hyperparameters.

Solver	batch size	steps	F1 score on dev set	TPUs	Time
Baseline	512	1000k	90.395	16	81.4h
LAMB	512	1000k	91.752	16	82.8h
LAMB	1k	500k	91.761	32	43.2h
LAMB	2k	250k	91.946	64	21.4h
LAMB	4k	125k	91.137	128	693.6m
LAMB	8k	62500	91.263	256	390.5m
LAMB	16k	31250	91.345	512	200.0m
LAMB	32k	15625	91.475	1024	101.2m
LAMB	64k/32k	8599	90.584	1024	76.19m

Table 2: ADAMW stops scaling at the batch size of 16K. The target F1 score is 90.5. LAMB achieves a F1 score of 91.345.

Solver	batch size	warmup steps	LR	last step infomation	F1 score on dev set
ADAMW	16K	0.05×31250	0.0001	loss=8.04471, step=28126	diverged
ADAMW	16K	0.05×31250	0.0002	loss=7.89673, step=28126	diverged
ADAMW	16K	0.05×31250	0.0003	loss=8.35102, step=28126	diverged
ADAMW	16K	0.10×31250	0.0001	loss=2.01419, step=31250	86.034
ADAMW	16K	0.10×31250	0.0002	loss=1.04689, step=31250	88.540
ADAMW	16K	0.10×31250	0.0003	loss=8.05845, step=20000	diverged
ADAMW	16K	0.20×31250	0.0001	loss=1.53706, step=31250	85.231
ADAMW	16K	0.20×31250	0.0002	loss=1.15500, step=31250	88.110
ADAMW	16K	0.20×31250	0.0003	loss=1.48798, step=31250	85.653

Overview

- NLP Trends
- Autoregressive vs Autoencoding
- XLNet
  - Permutation Language model
  - Two-stream self-attention mechanism
  - Recurrence mechanism

Autoregressive language model (AR)

- 일반적인 Language Model의 학습방법
- 이전 token을 보고 다음 token을 예상 하는 방식
- 주어진 문장 $X = [x_1, x_2, \dots, x_T]$ 일때

정방향 AR 모델

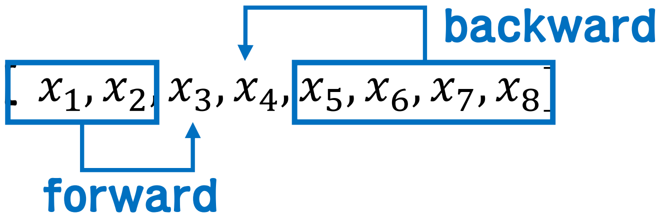
$$\max_{\theta} \log p_{\theta}(X) = \sum_{t=1}^{T} \log p_{\theta}(x_t \mid x_{<t})$$

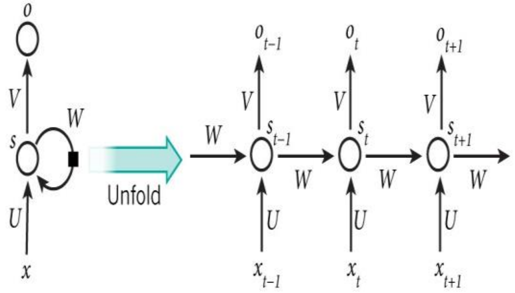
- AR 모델은 방향성(forward, backword)
- 한쪽 방향의 정보만 이용 가능
- 양방향성(bi-directional) 모델도 각 방향에 대해 독립적인 학습

Autoregressive Language Model

$[\ x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8]$

$[\ x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8]$





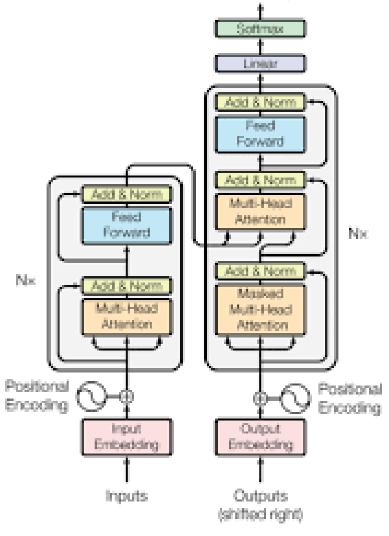


Figure 1: The Transformer - model architecture.

Autoencoding language model (AE)

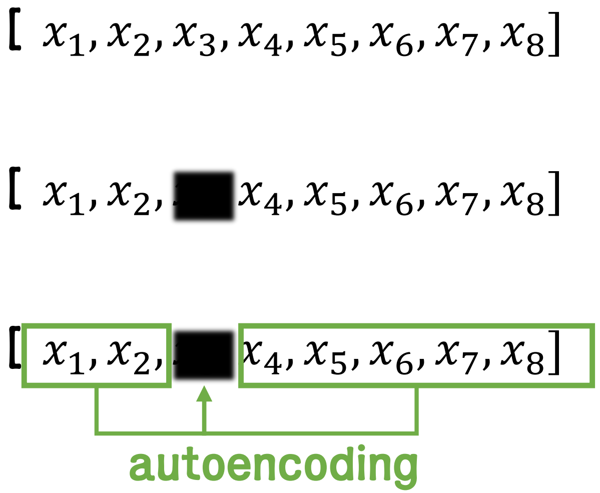
- 주어진 문장을 훼손시키고 복원시키는 방식
- [MASK] 토큰을 맞추기 위해 양방향 정보를 이용
- 오토인코더 모델은 [MASK] 토큰에 대해서만 예측을 진행

- 훼손시킨 문장$(\hat{X})$, 훼손된 token$(\bar{X})$ 이라 할 때 AE모델 수식은 다음과 같다.

$$\max_{\theta} \log p_{\theta}(\bar{X} \mid \hat{X}) \approx \sum_{t=1}^{T} m_t \log p_{\theta}(x_t \mid \hat{X})$$

- Token이 $[MASK]$ 일때 $m_t = 1$, 나머지 경우에는 $m_t = 0$
- [MASK] token 에 대해서 prediction을 진행

Autoencoding Language Model



$[\ x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8]$
$[\ x_1, x_2, ■ x_4, x_5, x_6, x_7, x_8]$
$[\ x_1, x_2, ■ x_4, x_5, x_6, x_7, x_8]$
autoencoding

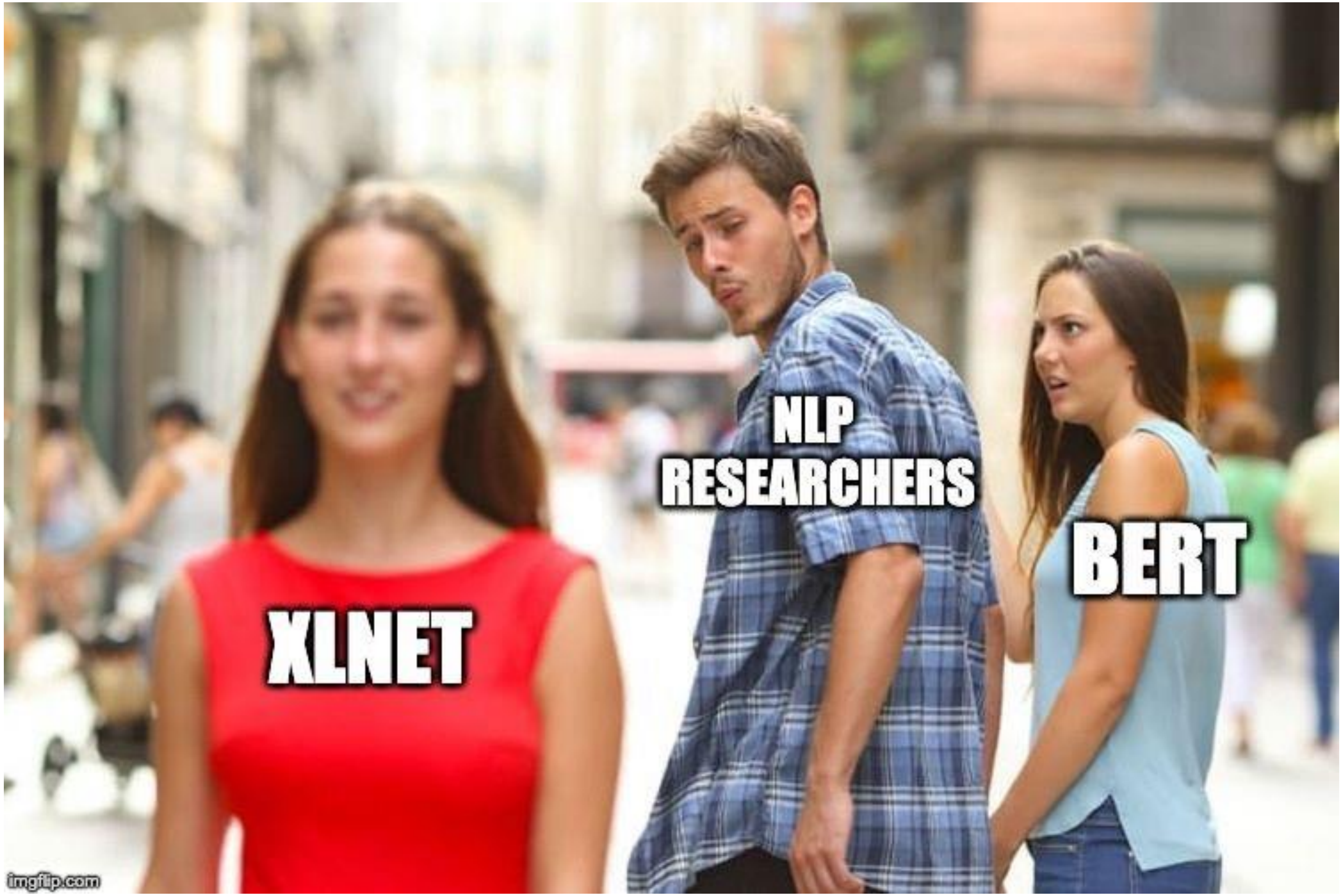
Overview

- NLP Trends

- Autoregressive vs Autoencoding

- XLNet
  - Permutation Language model
  - Two-stream self-attention mechanism
  - Recurrence mechanism

Hello, XLNet!



Experiments

- **Pretraining Datasets**
  - BERT : BookCorpus + English Wikipedia

    (총: 33억 token)

  - XLNet : BookCorpus + English Wikipedia + Giga5 + ClueWeb 2012-B + Common Crawl

    (2.78B) (1.09B) (4.75B) (4.30B) (19.97B) (총:328억 token)

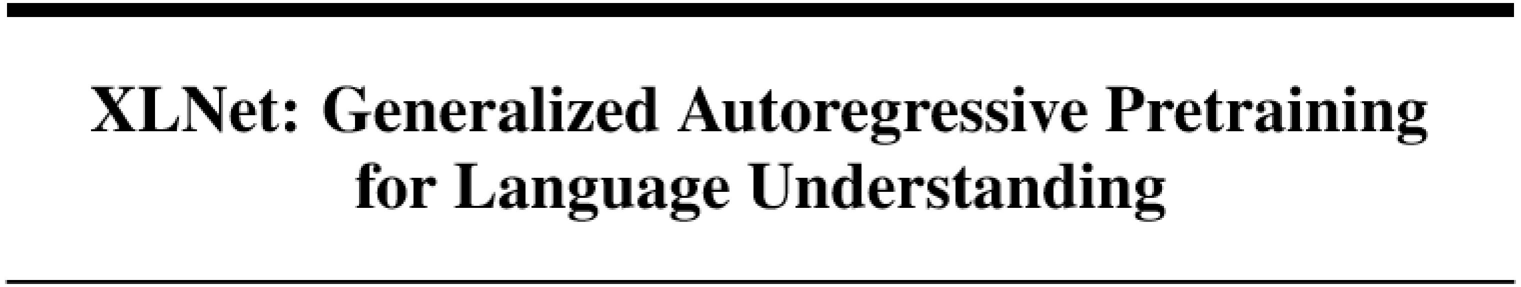
- **Model Size**
  - $XLNet - Large \approx BERT - Large$

- **Training Time**
  - 512TPU v3 chips for 500K steps with adam optimizer(batch size 2048) ➔ 2.5days

    ➔ 약 245,000$ (한화 2억 4500만원)

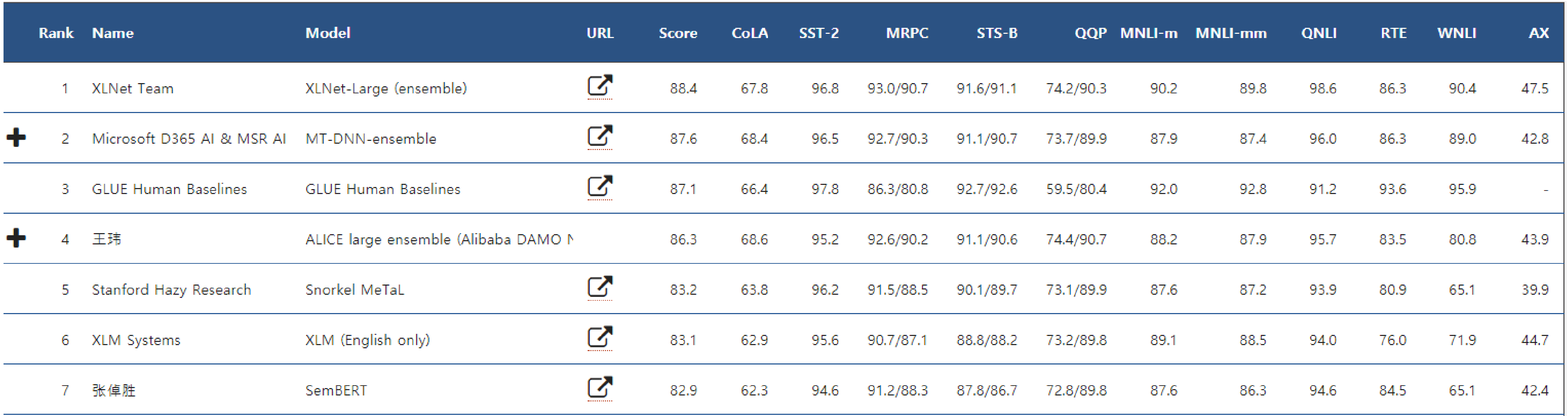
    ➔ Still underfits, 계속된 학습이 NLP task 에 대해서 도움이 되지 않는다고 판단

XLNet is good



XLNet: Generalized Autoregressive Pretraining for Language Understanding

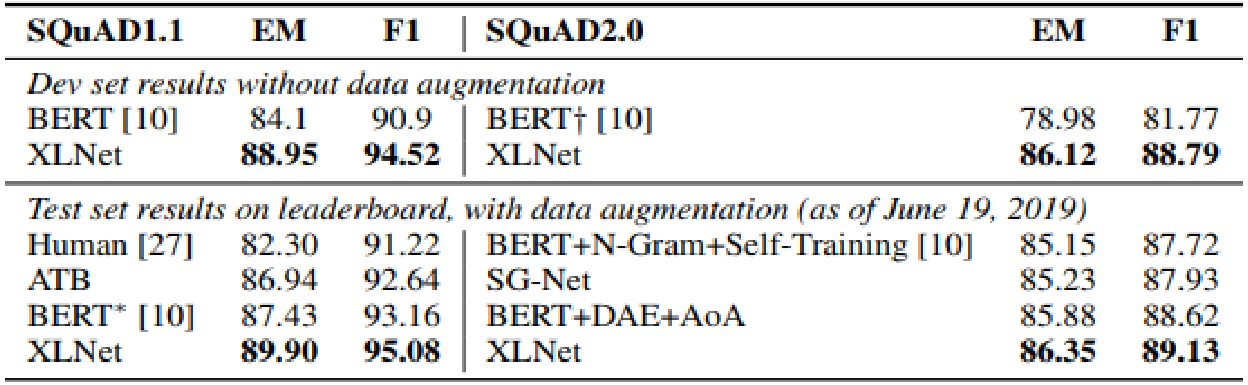
GLUE dataset



	Rank	Name	Model	URL	Score	CoLA	SST-2	MRPC	STS-B	QQP	MNLI-m	MNLI-mm	QNLI	RTE	WNLI	AX
	1	XLNet Team	XLNet-Large (ensemble)		88.4	67.8	96.8	93.0/90.7	91.6/91.1	74.2/90.3	90.2	89.8	98.6	86.3	90.4	47.5
+	2	Microsoft D365 AI & MSR AI	MT-DNN-ensemble		87.6	68.4	96.5	92.7/90.3	91.1/90.7	73.7/89.9	87.9	87.4	96.0	86.3	89.0	42.8
	3	GLUE Human Baselines	GLUE Human Baselines		87.1	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	-
+	4	王玮	ALICE large ensemble (Alibaba DAMO N		86.3	68.6	95.2	92.6/90.2	91.1/90.6	74.4/90.7	88.2	87.9	95.7	83.5	80.8	43.9
	5	Stanford Hazy Research	Snorkel MeTaL		83.2	63.8	96.2	91.5/88.5	90.1/89.7	73.1/89.9	87.6	87.2	93.9	80.9	65.1	39.9
	6	XLM Systems	XLM (English only)		83.1	62.9	95.6	90.7/87.1	88.8/88.2	73.2/89.8	89.1	88.5	94.0	76.0	71.9	44.7
	7	张倬胜	SemBERT		82.9	62.3	94.6	91.2/88.3	87.8/86.7	72.8/89.8	87.6	86.3	94.6	84.5	65.1	42.4

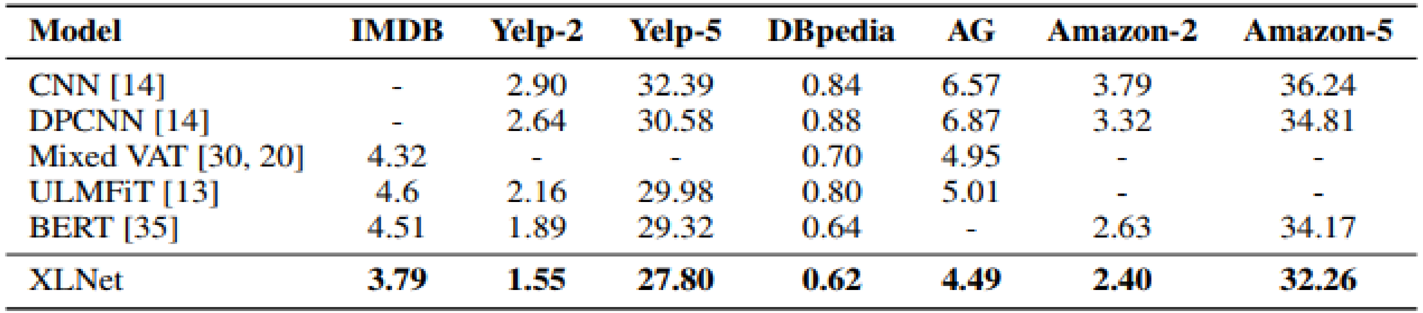
2019. 07. 05 기준

XLNet is good good



SQuAD1.1	EM	F1	SQuAD2.0	EM	F1
Dev set results without data augmentation					
BERT [10]	84.1	90.9	BERT† [10]	78.98	81.77
XLNet	**88.95**	**94.52**	XLNet	**86.12**	**88.79**
Test set results on leaderboard, with data augmentation (as of June 19, 2019)					
Human [27]	82.30	91.22	BERT+N-Gram+Self-Training [10]	85.15	87.72
ATB	86.94	92.64	SG-Net	85.23	87.93
BERT* [10]	87.43	93.16	BERT+DAE+AoA	85.88	88.62
XLNet	**89.90**	**95.08**	XLNet	**86.35**	**89.13**

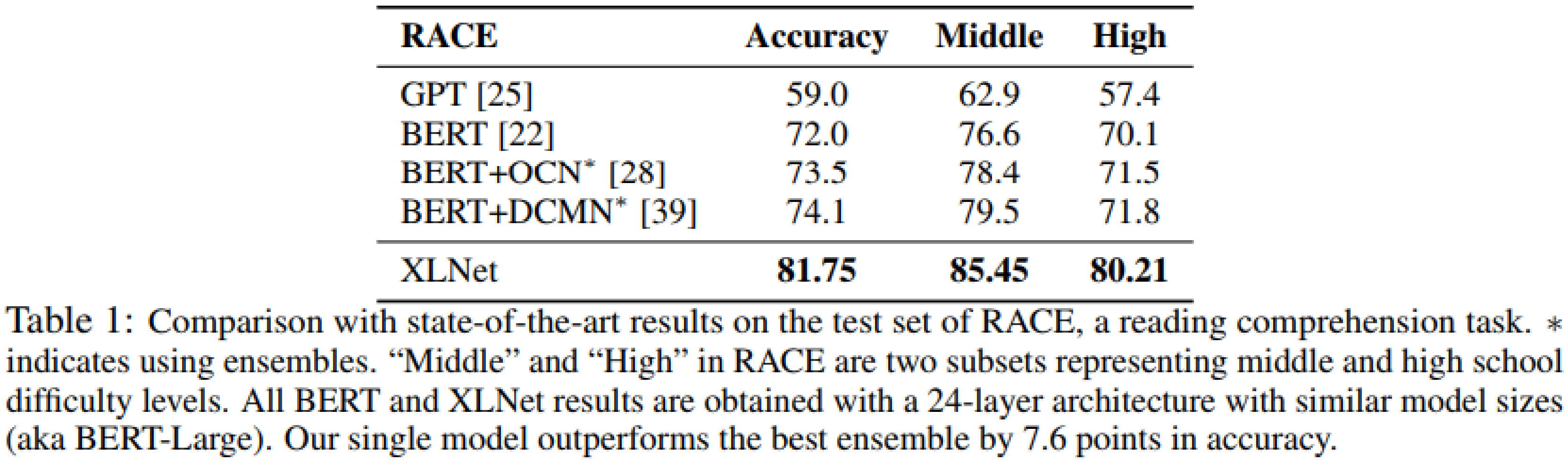
Table 2: A single model XLNet outperforms human and the best ensemble by 7.6 EM and 2.5 EM on SQuAD1.1. ∗ means ensembles, † marks our runs with the official code.



Model	IMDB	Yelp-2	Yelp-5	DBpedia	AG	Amazon-2	Amazon-5
CNN [14]	-	2.90	32.39	0.84	6.57	3.79	36.24
DPCNN [14]	-	2.64	30.58	0.88	6.87	3.32	34.81
Mixed VAT [30, 20]	4.32	-	-	0.70	4.95	-	-
ULMFiT [13]	4.6	2.16	29.98	0.80	5.01	-	-
BERT [35]	4.51	1.89	29.32	0.64	-	2.63	34.17
XLNet	**3.79**	**1.55**	**27.80**	**0.62**	**4.49**	**2.40**	**32.26**

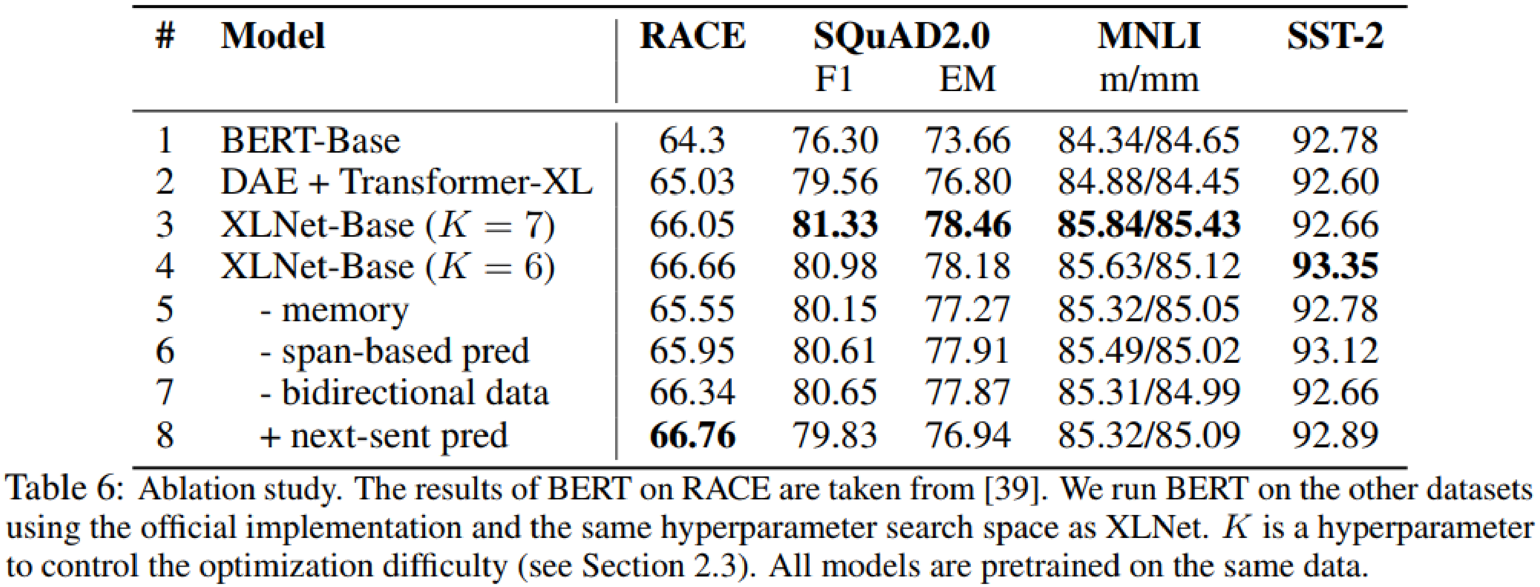
Table 3: Comparison with state-of-the-art error rates on the test sets of several text classification datasets. All BERT and XLNet results are obtained with a 24-layer architecture with similar model sizes (aka BERT-Large).

XLNet is good good



RACE	Accuracy	Middle	High
GPT [25]	59.0	62.9	57.4
BERT [22]	72.0	76.6	70.1
BERT+OCN* [28]	73.5	78.4	71.5
BERT+DCMN* [39]	74.1	79.5	71.8
XLNet	**81.75**	**85.45**	**80.21**

Table 1: Comparison with state-of-the-art results on the test set of RACE, a reading comprehension task. $*$ indicates using ensembles. "Middle" and "High" in RACE are two subsets representing middle and high school difficulty levels. All BERT and XLNet results are obtained with a 24-layer architecture with similar model sizes (aka BERT-Large). Our single model outperforms the best ensemble by 7.6 points in accuracy.



#	Model	RACE	SQuAD2.0		MNLI	SST-2
			F1	EM	m/mm	
1	BERT-Base	64.3	76.30	73.66	84.34/84.65	92.78
2	DAE + Transformer-XL	65.03	79.56	76.80	84.88/84.45	92.60
3	XLNet-Base ($K = 7$)	66.05	**81.33**	**78.46**	**85.84/85.43**	92.66
4	XLNet-Base ($K = 6$)	66.66	80.98	78.18	85.63/85.12	**93.35**
5	- memory	65.55	80.15	77.27	85.32/85.05	92.78
6	- span-based pred	65.95	80.61	77.91	85.49/85.02	93.12
7	- bidirectional data	66.34	80.65	77.87	85.31/84.99	92.66
8	+ next-sent pred	**66.76**	79.83	76.94	85.32/85.09	92.89

Table 6: Ablation study. The results of BERT on RACE are taken from [39]. We run BERT on the other datasets using the official implementation and the same hyperparameter search space as XLNet. K is a hyperparameter to control the optimization difficulty (see Section 2.3). All models are pretrained on the same data.

XLNet Summary

Generalized Autoregressive Pretraining for Language Understanding

- AR 모델과 AE 모델의 장점을 합친 새로운 Language model

  ➔ Permutation Language Model (순열 언어 모델)

- Two-stream attention mechanism을 제안

- Recurrence Mechanism

다양한 NLP task에서 뛰어난 성능 향상을 보이며 State-of-the-art 달성

Autoencoding language model (AE) Problem

1. Independence Assumption

무작위로 [MASK]를 손상시키고 복원하는 방식을 위해서

Independence Assumption 을 가정함 ➔ [MASK] token 의 확률을 계산하기 위해서 독립시행으로 구분

각 token을 독립적으로 예측을 하면서 단어 사이의 dependency가 학습되지 않는다

Train sentence(X) : 나는 남산에 가서 남산 케이블카를 봤다

Corrupted version$(\hat{X})$: 나는 [MASK] 에 가서 [MASK] [MASK] 를 봤다

Reconstruct : 나는 남산에 가서 남산 케이블카 를 봤다 (O)

나는 음식점에 가서 모집 공고 를 봤다 (O)

나는 남산에 가서 모집 공고 를 봤다 (X)

나는 음식점에 가서 모집 케이블카 를 봤다 (X)

Autoencoding language model (AE) Problem

2. Input noise

AE모델에서는 정상적인 문장을 학습을 위해서 [MASK] token으로 훼손

이후 복원하는 작업을 통해서 context에 대한 정보를 학습함 (Pre-training)

하지만 downstream task를 위해서 fine-tuning시 [MASK] token은 사용되지 않음

-> Pre-training 과 fine-tuning 시 학습방법에 대한 불일치

Overview

- NLP Trends

- Autoregressive vs Autoencoding

- XLNet
  - **Permutation Language model**
  - Two-stream self-attention mechanism
  - Recurrence mechanism

Permutation Language Modeling

Input sequence 의 모든 permutation을 고려

$$input\ sequence = [x_1, x_2, x_3, x_4]$$

Permutation 집합은 총 $4! = 24$개 가 존재

x_3 를 예측하기 위해서 오른쪽 그림과 같은 계산

모든 permutation 에 대해서 이 과정을 반복

→ 양방향에서 정보를 수집

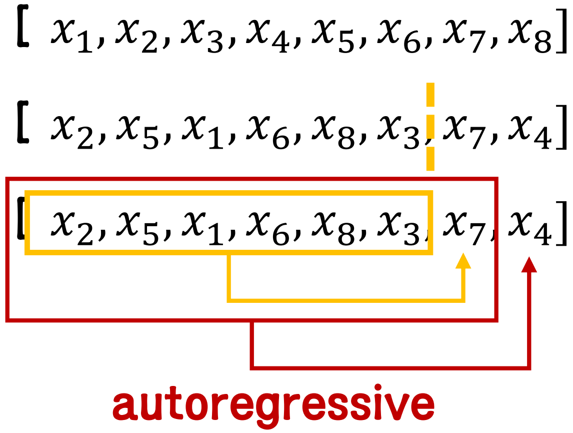
→ bi-directional 한 AR 모델

→ Independent Assumption 필요 X

→ Pre-training 과 fine-tuning 사이의 불일치 X

→ 기존 모델의 단점을 극복

Permutation Language Model

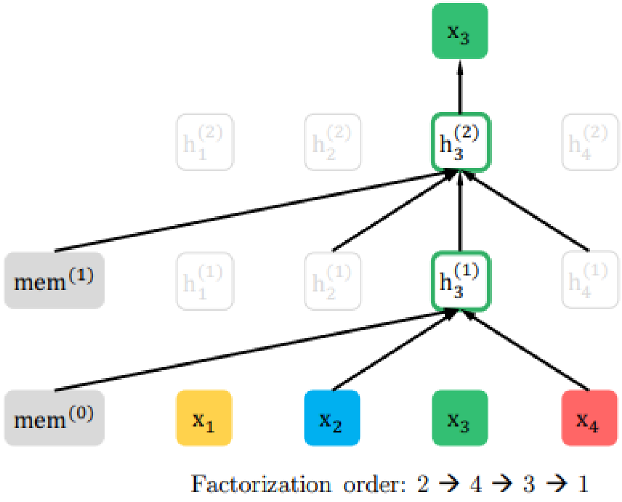


$[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8]$

$[x_2, x_5, x_1, x_6, x_8, x_3, x_7, x_4]$

$[x_2, x_5, x_1, x_6, x_8, x_3, x_7, x_4]$

autoregressive



$$p(x) = p(x_2|\,mem)p(x_4|\,mem, x_2)p(x_3|mem, x_2, x_4)$$

Permutation Language Modeling

Input sequence 의 모든 permutation을 고려

$$input\ sequence = [x_1, x_2, x_3, x_4]$$

Permutation 집합은 총 $4! = 24$개 가 존재

x_3 를 예측하기 위해서 오른쪽 그림과 같은 계산

모든 permutation 에 대해서 이 과정을 반복

→ 양방향에서 정보를 수집

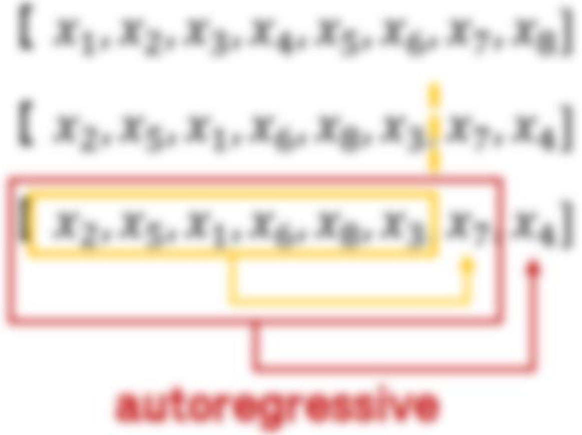
→ bi-directional 한 AR 모델

→ Independent Assumption 필요 X

→ Pre-training 과 fine-tuning 사이의 불일치 X

→ 기존 모델의 단점을 극복

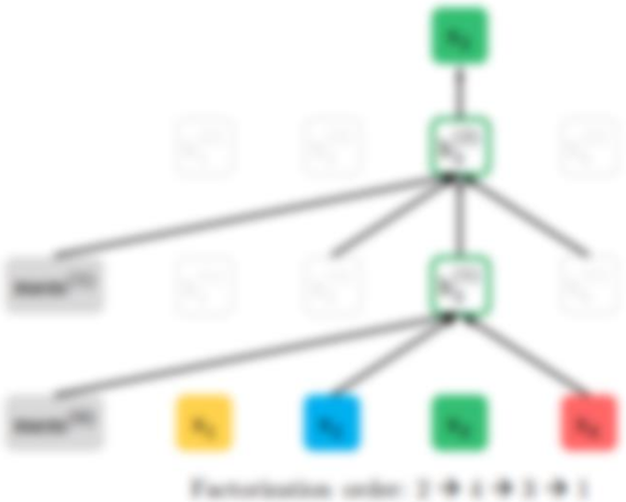
Permutation Language Model



$[\,x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8]$

$[\,x_2, x_5, x_1, x_6, x_8, x_3, x_7, x_4]$

$[\,x_2, x_5, x_1, x_6, x_8, x_3, x_7, x_4]$

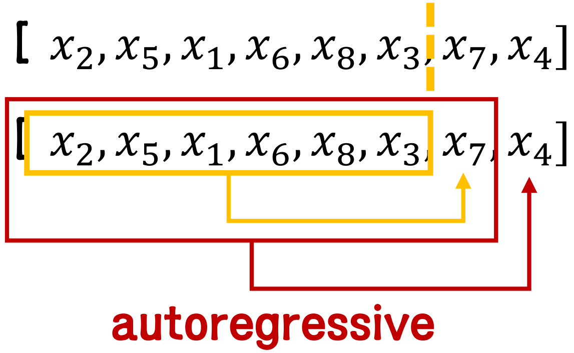


$$p(x) = p(x_2|\,mem)p(x_4|\,mem, x_2)p(x_3|mem, x_2, x_4)$$

Permutation Language Modeling

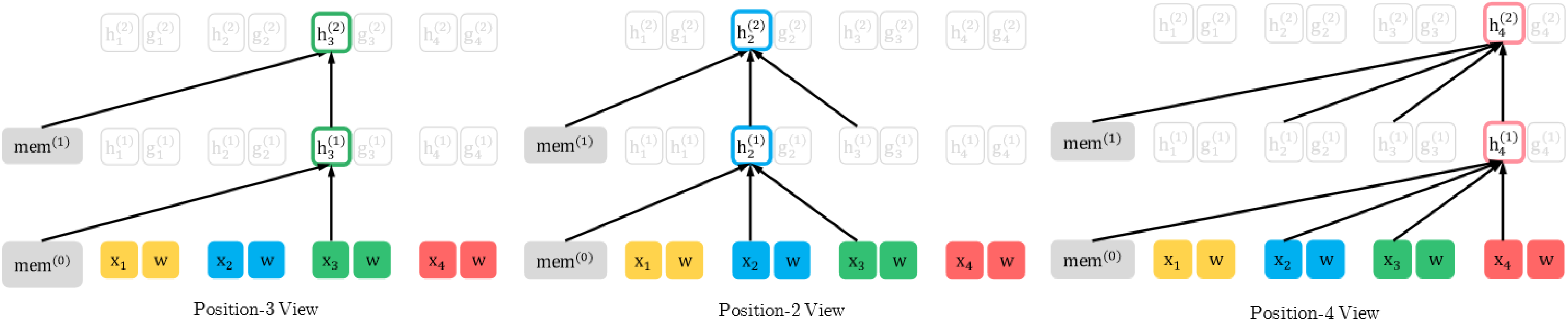
Permutation Language Model

$$[\ x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8]$$



$$[\ x_2, x_5, x_1, x_6, x_8, x_3, x_7, x_4]$$

$$[\ x_2, x_5, x_1, x_6, x_8, x_3, x_7, x_4]$$



$h_1^{(2)}$ $g_1^{(2)}$ $h_2^{(2)}$ $g_2^{(2)}$ $h_3^{(2)}$ $g_3^{(2)}$ $h_4^{(2)}$ $g_4^{(2)}$
mem$^{(1)}$
$h_1^{(1)}$ $g_1^{(1)}$ $h_2^{(1)}$ $g_2^{(1)}$ $h_3^{(1)}$ $g_3^{(1)}$ $h_4^{(1)}$ $g_4^{(1)}$
mem$^{(0)}$
x_1 w x_2 w x_3 w x_4 w

Position-3 View

Position-2 View

Position-4 View

Permutation Language Modeling

Permutation language model의 수식 표현

$$\mathcal{Z}_t\ is\ all\ possible\ permutations\ of\ length - T\ index\ sequence\ ([1,2,3,\ldots,T])$$

$$\max_{\theta} \mathbb{E}_{z\sim\mathcal{Z}_T}\left[\sum_{t=1}^{T} \log p_{\theta}\left(x_{z_t} \mid x_{z_{<t}}\right)\right]$$

[$x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8$] **8개의 토큰 → 8!(40,320) 개의 permutations order 생성 → 연산량** ⬆

부분 예측(Partial Prediction)

- **Hyper parameter K - 토큰의 1/K 만 예측**

K = 8

$$[\ x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8] \rightarrow \left.\begin{array}{l} [\ x_2, x_5, x_1, x_6, x_8, x_3, x_7, \mid x_4] \\ [\ x_3, x_7, x_6, x_1, x_8, x_2, x_4, \mid x_5] \\ \qquad\qquad\vdots \end{array}\right\}$$ **40,320개**

BERT와 XLNet 모델 비교

Sequence : New York is a city

$$\mathcal{J}_{BERT} = \log p(New|\ is\ a\ city\ ) + \log p\ (\ York|\ is\ a\ city),$$

$$\mathcal{J}_{XLNet} = \log p(New|\ is\ a\ city\ ) + \log p(\ York\ |\ {\color{red}New}\ is\ a\ city\ )$$

Permutation Language Modeling Problem

Input sequence 의 token의 order를 permutation 하기 때문에 기존 AR 모델에서 관측되지 않는

Position 구분에 문제가 발생

기존의 AR 모델은 방향성이 존재 (forward, backward) 하기 때문에 position 구분할 필요가 없었음

Example)

Input sequence : [New, York, is, a, city]

Factorization order : 1 → 2 → 5 → ⋯ $p(New \mid mem)p(York \mid mem, New)p(city|mem, New, York)$

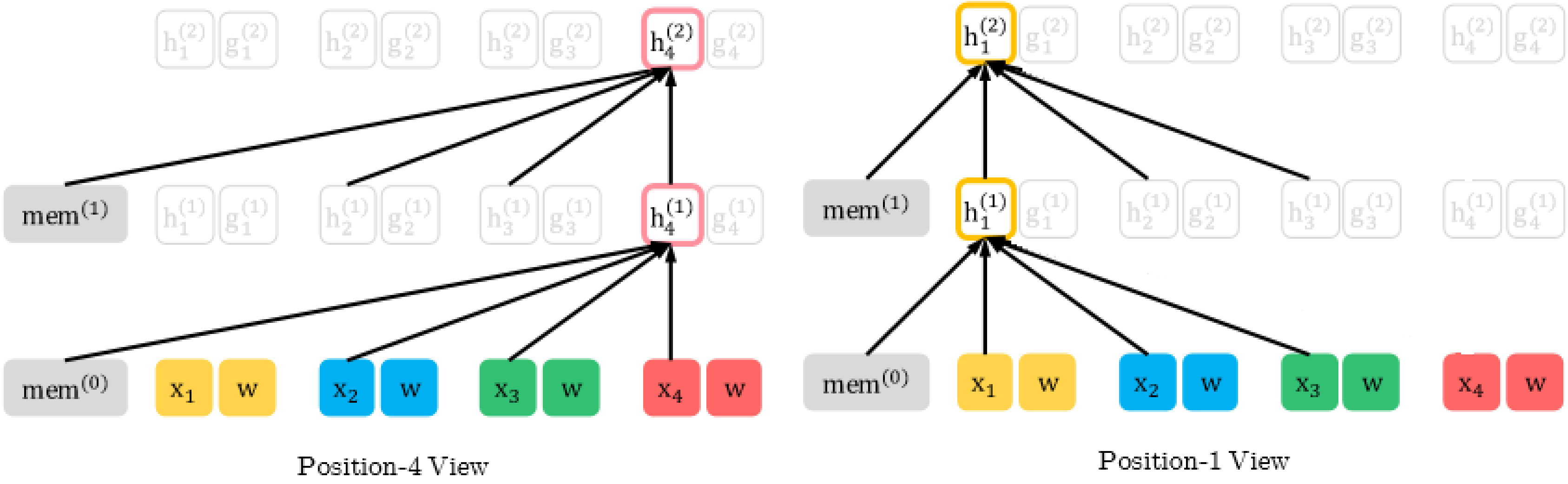
1 → 2 → 3 → ⋯ $p(New \mid mem)p(York \mid mem, New)p(\boldsymbol{is}|mem, New, York)$

같은 representation 을 이용해서 다른 token을 예측해야 하는 문제가 발생

Overview

- NLP Trends
- Autoregressive vs Autoencoding
- XLNet
  - Permutation Language model
  - Two-stream self-attention mechanism
  - Recurrence mechanism

Two-Stream Self-Attention


$h_1^{(2)}$ $g_1^{(2)}$ $h_2^{(2)}$ $g_2^{(2)}$ $h_3^{(2)}$ $g_3^{(2)}$ $h_4^{(2)}$ $g_4^{(2)}$
mem$^{(1)}$
$h_1^{(1)}$ $g_1^{(1)}$ $h_2^{(1)}$ $g_2^{(1)}$ $h_3^{(1)}$ $g_3^{(1)}$ $h_4^{(1)}$ $g_4^{(1)}$
mem$^{(0)}$
x_1 w x_2 w x_3 w x_4 w
Position-4 View
Position-1 View

Two-Stream Self-Attention

문제 해결을 위해서 새로운 next-token distribution 을 제안

$h_\theta(x_{z<t}) \rightarrow g_\theta(x_{z<t}, z_t) \quad context\ information\ (x_{z<t}),\ target\ position\ z_t$

기존에 context information 만 사용하던 $h_\theta(x_{z<t})$ **을 개선하여**
Target position 을 반영한 $g_\theta(x_{z<t}, z_t)$ **를 사용**

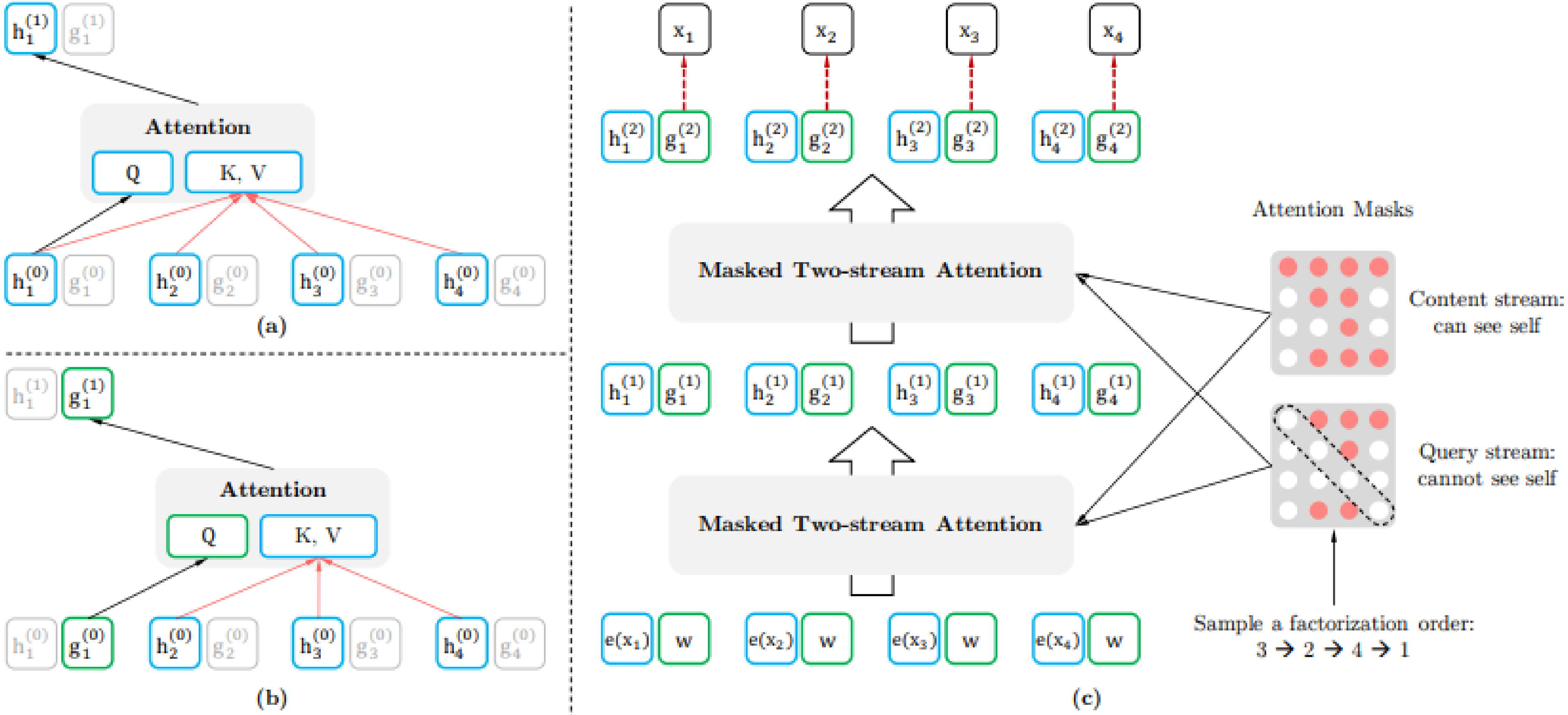
$g_\theta(x_{z<t}, z_t)$**를 표현하기 위해 Two-Stream Self-Attention을 제안**

Two-Stream

- $\boldsymbol{query\ stream}$: $g_{zt}^{(m)}$**으로 표시되는** $Attention\ stream$ **(Permutation 계산시 본인 제외)**
  무작위로 생성된 가중치 값, 위치 정보

- $\boldsymbol{content\ stream}$: $h_{zt}^{(m)}$ **으로 표시되는** $Attention\ stream$ **(Permutation 계산시 본인 포함)**
  워드 임베딩 값

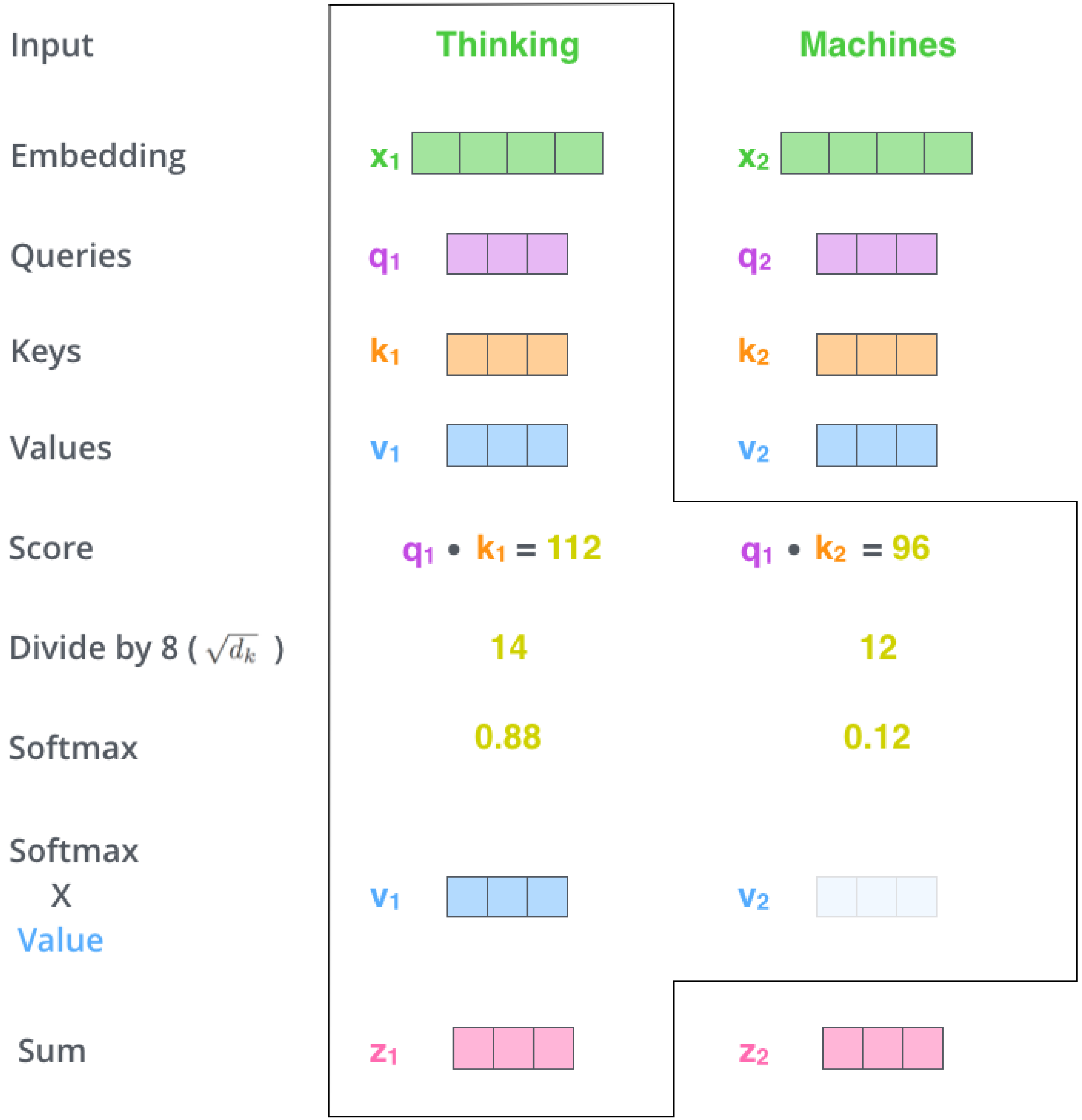
Two-Stream Self-Attention



$h_1^{(1)}$ $g_1^{(1)}$
Attention
Q
K, V
$h_1^{(0)}$ $g_1^{(0)}$ $h_2^{(0)}$ $g_2^{(0)}$ $h_3^{(0)}$ $g_3^{(0)}$ $h_4^{(0)}$ $g_4^{(0)}$
(a)
$h_1^{(1)}$ $g_1^{(1)}$
Attention
Q
K, V
$h_1^{(0)}$ $g_1^{(0)}$ $h_2^{(0)}$ $g_2^{(0)}$ $h_3^{(0)}$ $g_3^{(0)}$ $h_4^{(0)}$ $g_4^{(0)}$
(b)
x_1 x_2 x_3 x_4
$h_1^{(2)}$ $g_1^{(2)}$ $h_2^{(2)}$ $g_2^{(2)}$ $h_3^{(2)}$ $g_3^{(2)}$ $h_4^{(2)}$ $g_4^{(2)}$
Masked Two-stream Attention
$h_1^{(1)}$ $g_1^{(1)}$ $h_2^{(1)}$ $g_2^{(1)}$ $h_3^{(1)}$ $g_3^{(1)}$ $h_4^{(1)}$ $g_4^{(1)}$
Masked Two-stream Attention
$e(x_1)$ w $e(x_2)$ w $e(x_3)$ w $e(x_4)$ w
Attention Masks
Content stream: can see self
Query stream: cannot see self
Sample a factorization order: 3 → 2 → 4 → 1
(c)

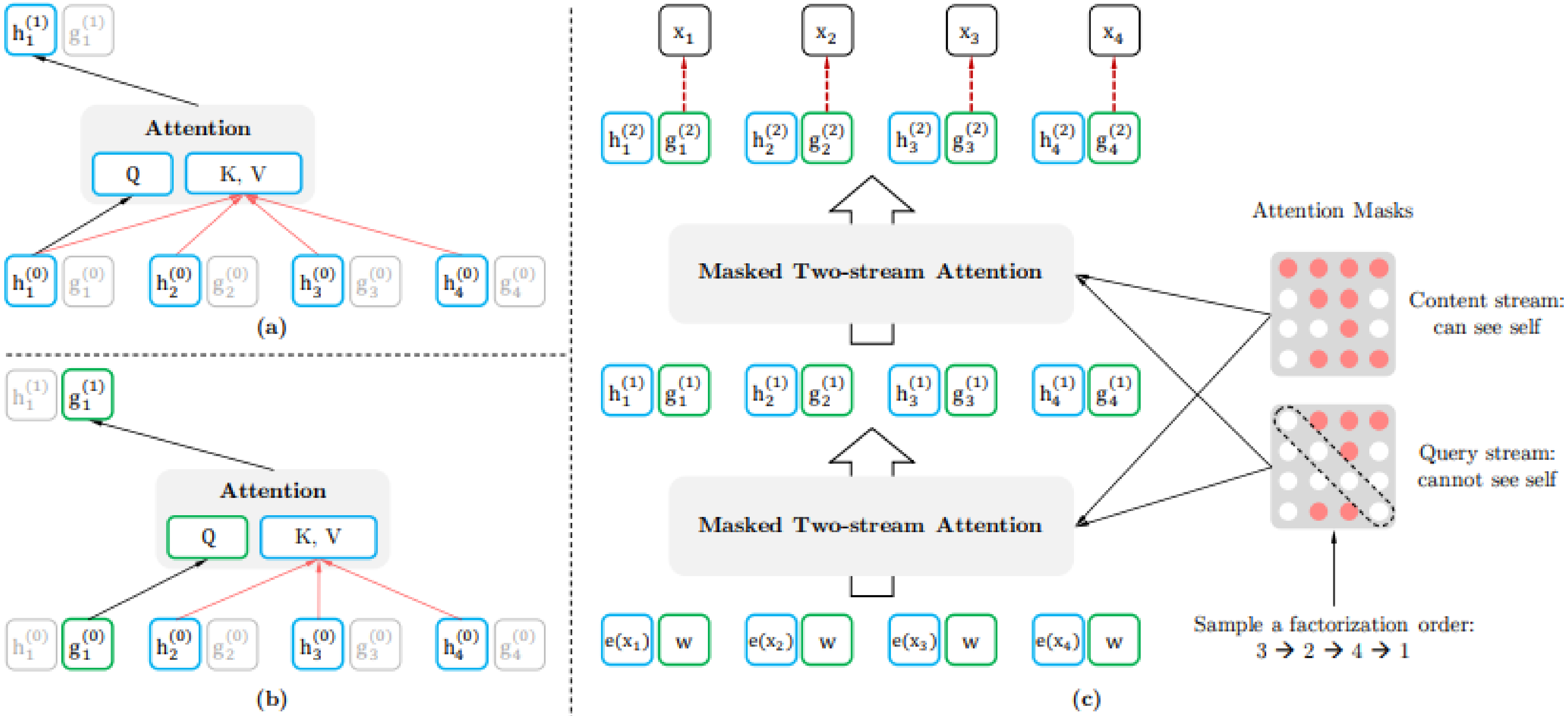
Figure 2: (a): Content stream attention, which is the same as the standard self-attention. (b): Query stream attention, which does not have access information about the content x_{z_t}. (c): Overview of the permutation language modeling training with two-stream attention.

Attention score



Input	Thinking	Machines
Embedding	x_1	x_2
Queries	q_1	q_2
Keys	k_1	k_2
Values	v_1	v_2
Score	$q_1 \bullet k_1 = 112$	$q_1 \bullet k_2 = 96$
Divide by 8 ($\sqrt{d_k}$)	14	12
Softmax	0.88	0.12
Softmax X Value	v_1	v_2
Sum	z_1	z_2

Two-Stream Self-Attention



$h_1^{(1)}$ $g_1^{(1)}$
Attention
Q
K, V
$h_1^{(0)}$ $g_1^{(0)}$ $h_2^{(0)}$ $g_2^{(0)}$ $h_3^{(0)}$ $g_3^{(0)}$ $h_4^{(0)}$ $g_4^{(0)}$
(a)
$h_1^{(1)}$ $g_1^{(1)}$
Attention
Q
K, V
$h_1^{(0)}$ $g_1^{(0)}$ $h_2^{(0)}$ $g_2^{(0)}$ $h_3^{(0)}$ $g_3^{(0)}$ $h_4^{(0)}$ $g_4^{(0)}$
(b)
x_1 x_2 x_3 x_4
$h_1^{(2)}$ $g_1^{(2)}$ $h_2^{(2)}$ $g_2^{(2)}$ $h_3^{(2)}$ $g_3^{(2)}$ $h_4^{(2)}$ $g_4^{(2)}$
Masked Two-stream Attention
$h_1^{(1)}$ $g_1^{(1)}$ $h_2^{(1)}$ $g_2^{(1)}$ $h_3^{(1)}$ $g_3^{(1)}$ $h_4^{(1)}$ $g_4^{(1)}$
Masked Two-stream Attention
$e(x_1)$ w $e(x_2)$ w $e(x_3)$ w $e(x_4)$ w
Attention Masks
Content stream: can see self
Query stream: cannot see self
Sample a factorization order: 3 → 2 → 4 → 1
(c)

Figure 2: (a): Content stream attention, which is the same as the standard self-attention. (b): Query stream attention, which does not have access information about the content x_{z_t}. (c): Overview of the permutation language modeling training with two-stream attention.

Two-Stream Self-Attention

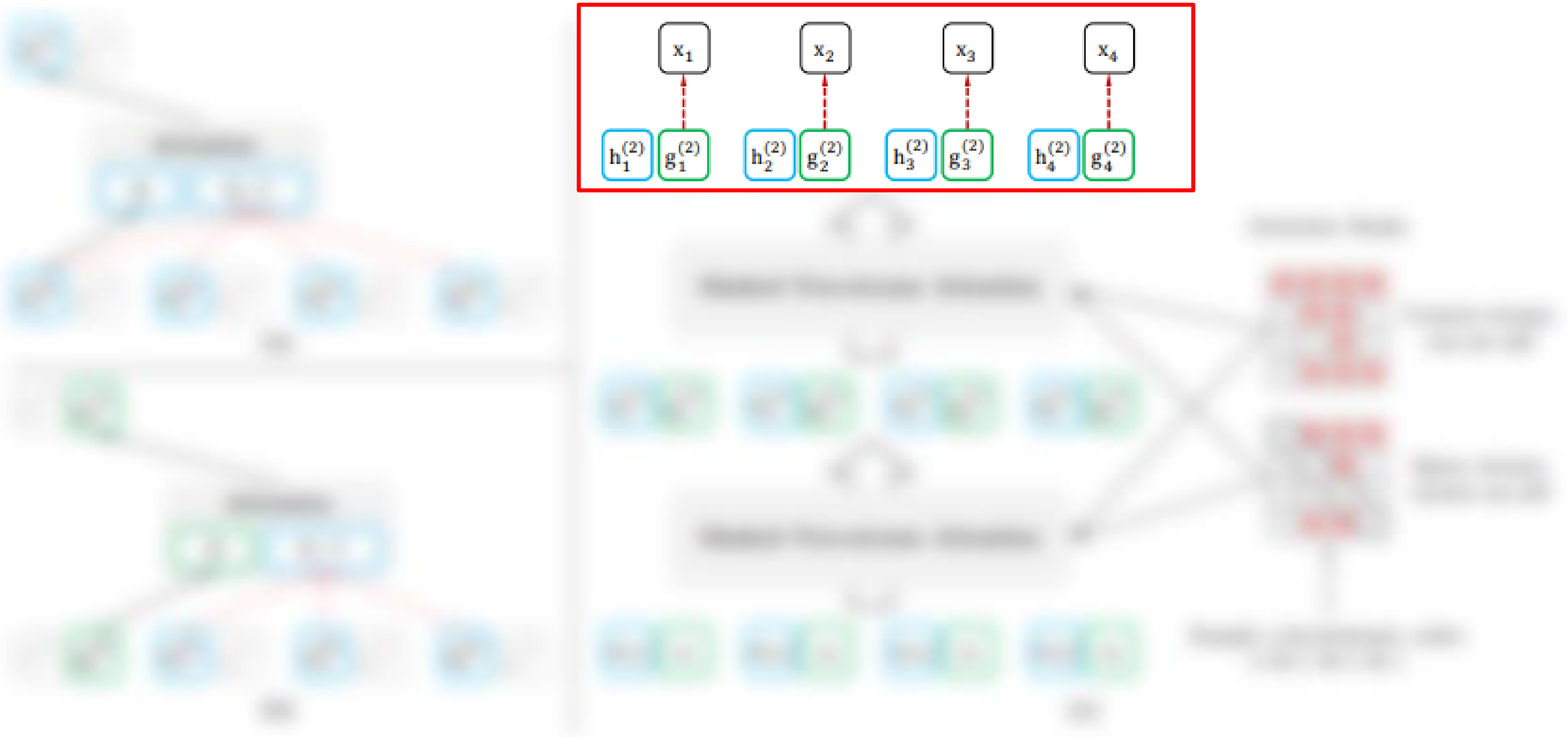
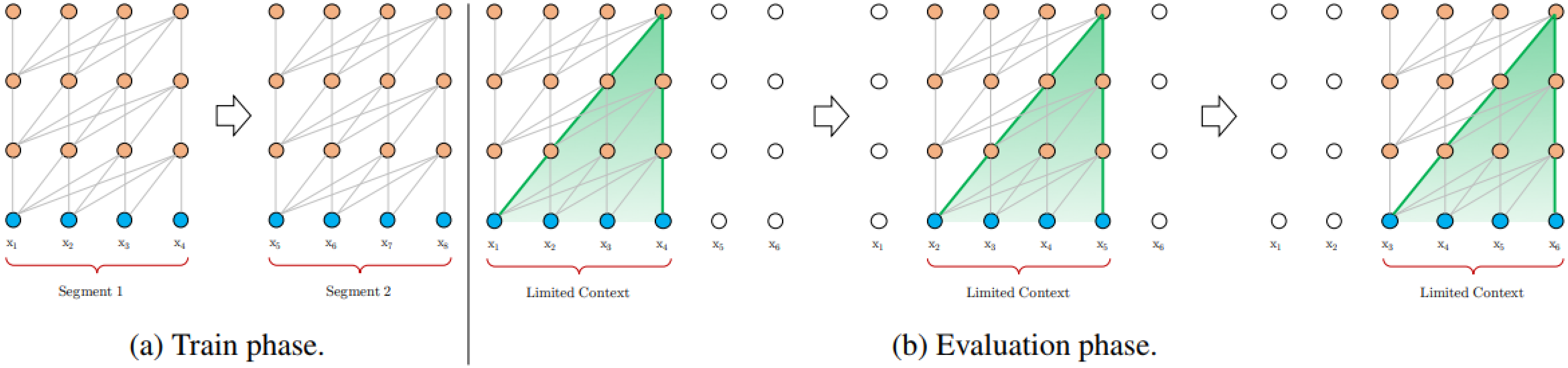


Figure 1: (a): Content stream attention, which is the same as the standard self-attention. (b): Query stream attention, which does not have access information about the content x_{z_t}. (c): Overview of the permutation language modeling training with two-stream attention.

Overview

- NLP Trends
- Autoregressive vs Autoencoding
- XLNet
    - Permutation Language model
    - Two-stream self-attention mechanism
    - Recurrence mechanism

Vanilla Transformer

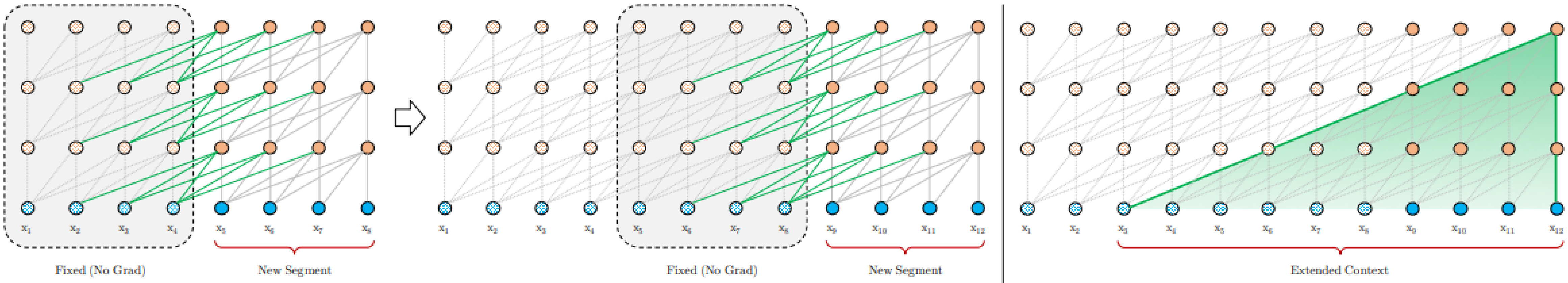


(a) Train phase.

(b) Evaluation phase.

Figure 1: Illustration of the vanilla model with a segment length 4.

Transformer-XL



(a) Training phase.

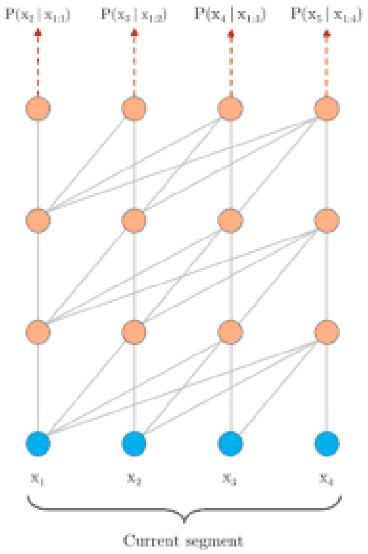
(b) Evaluation phase.

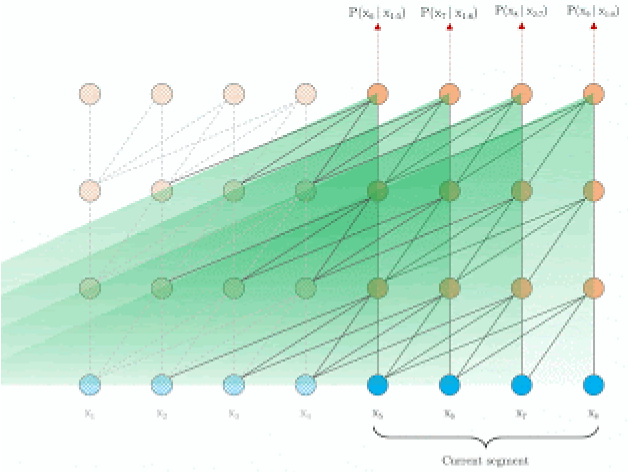
Figure 2: Illustration of the Transformer-XL model with a segment length 4.

Segment Recurrence

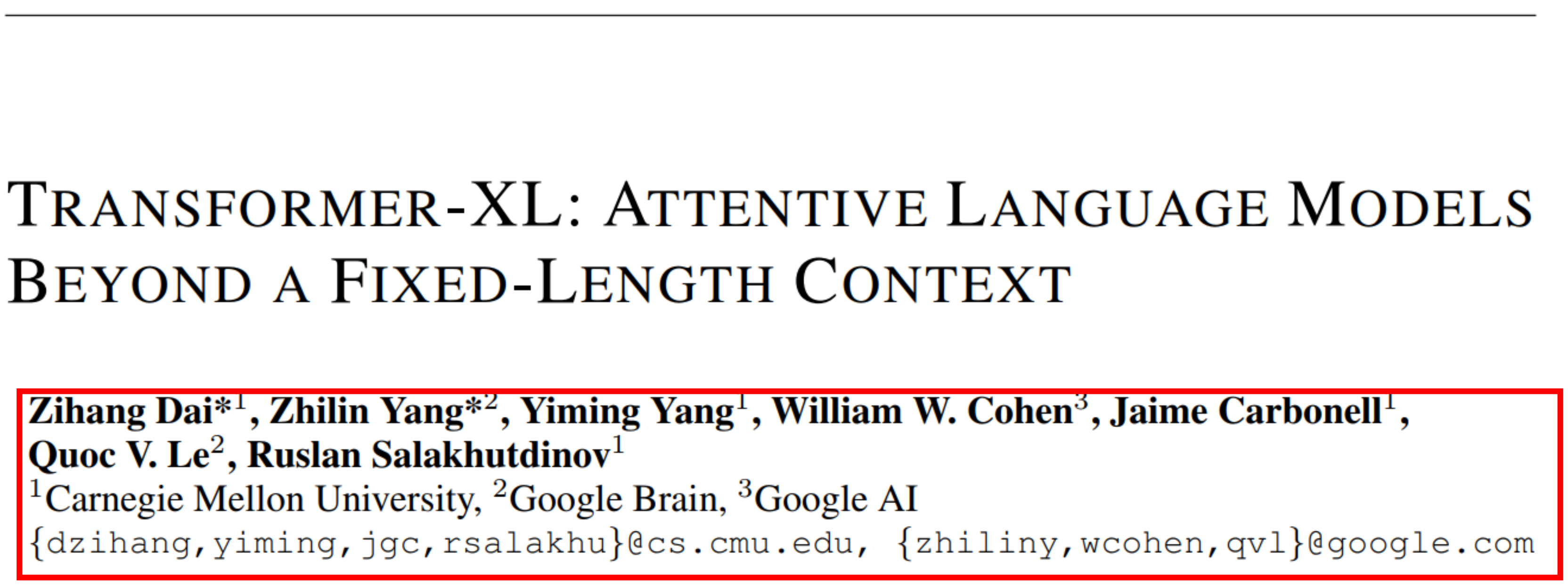
- RNNs 의 Hidden 처럼 이전 segment 의 결과를 다음 segment 계산에 반영

Transformer-XL



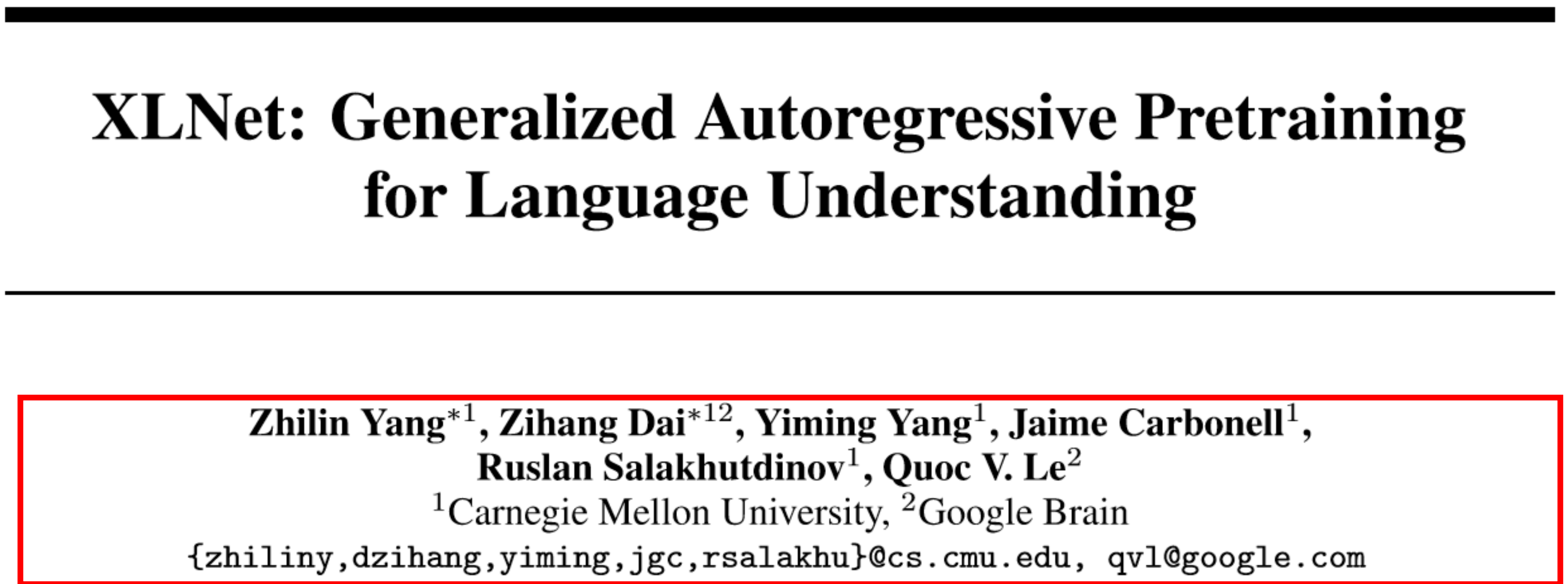


Transformer-XL & XLNet



TRANSFORMER-XL: ATTENTIVE LANGUAGE MODELS BEYOND A FIXED-LENGTH CONTEXT

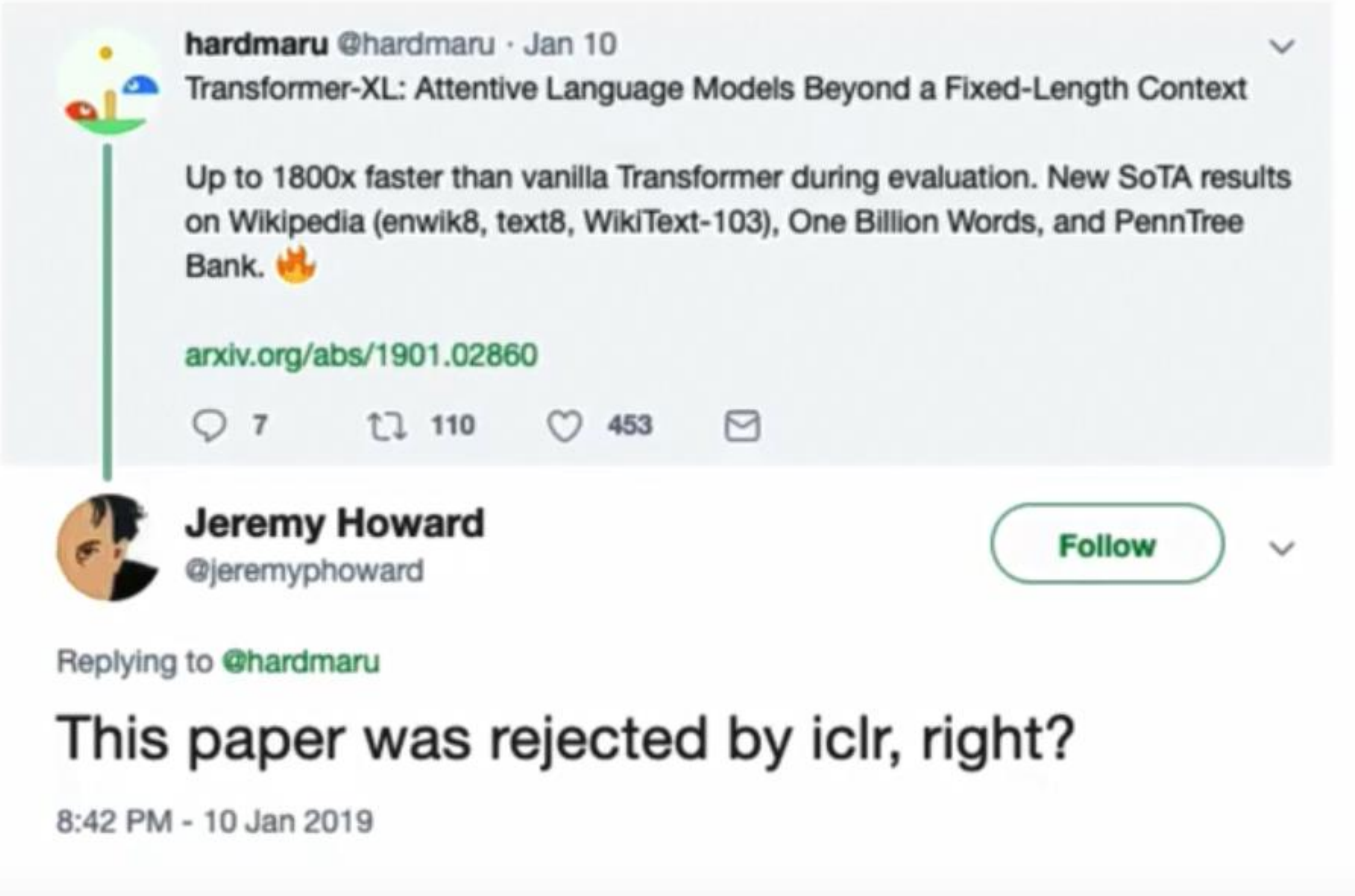
Zihang Dai*[1], Zhilin Yang*[2], Yiming Yang[1], William W. Cohen[3], Jaime Carbonell[1], Quoc V. Le[2], Ruslan Salakhutdinov[1]
[1]Carnegie Mellon University, [2]Google Brain, [3]Google AI
{dzihang,yiming,jgc,rsalakhu}@cs.cmu.edu, {zhiliny,wcohen,qvl}@google.com



XLNet: Generalized Autoregressive Pretraining for Language Understanding

Zhilin Yang*[1], Zihang Dai*[12], Yiming Yang[1], Jaime Carbonell[1], Ruslan Salakhutdinov[1], Quoc V. Le[2]
[1]Carnegie Mellon University, [2]Google Brain
{zhiliny,dzihang,yiming,jgc,rsalakhu}@cs.cmu.edu, qvl@google.com

Transformer-XL



hardmaru @hardmaru · Jan 10

Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context

Up to 1800x faster than vanilla Transformer during evaluation. New SoTA results on Wikipedia (enwik8, text8, WikiText-103), One Billion Words, and PennTree Bank. 🔥

arxiv.org/abs/1901.02860

Jeremy Howard
@jeremyphoward

Replying to @hardmaru

This paper was rejected by iclr, right?

8:42 PM - 10 Jan 2019

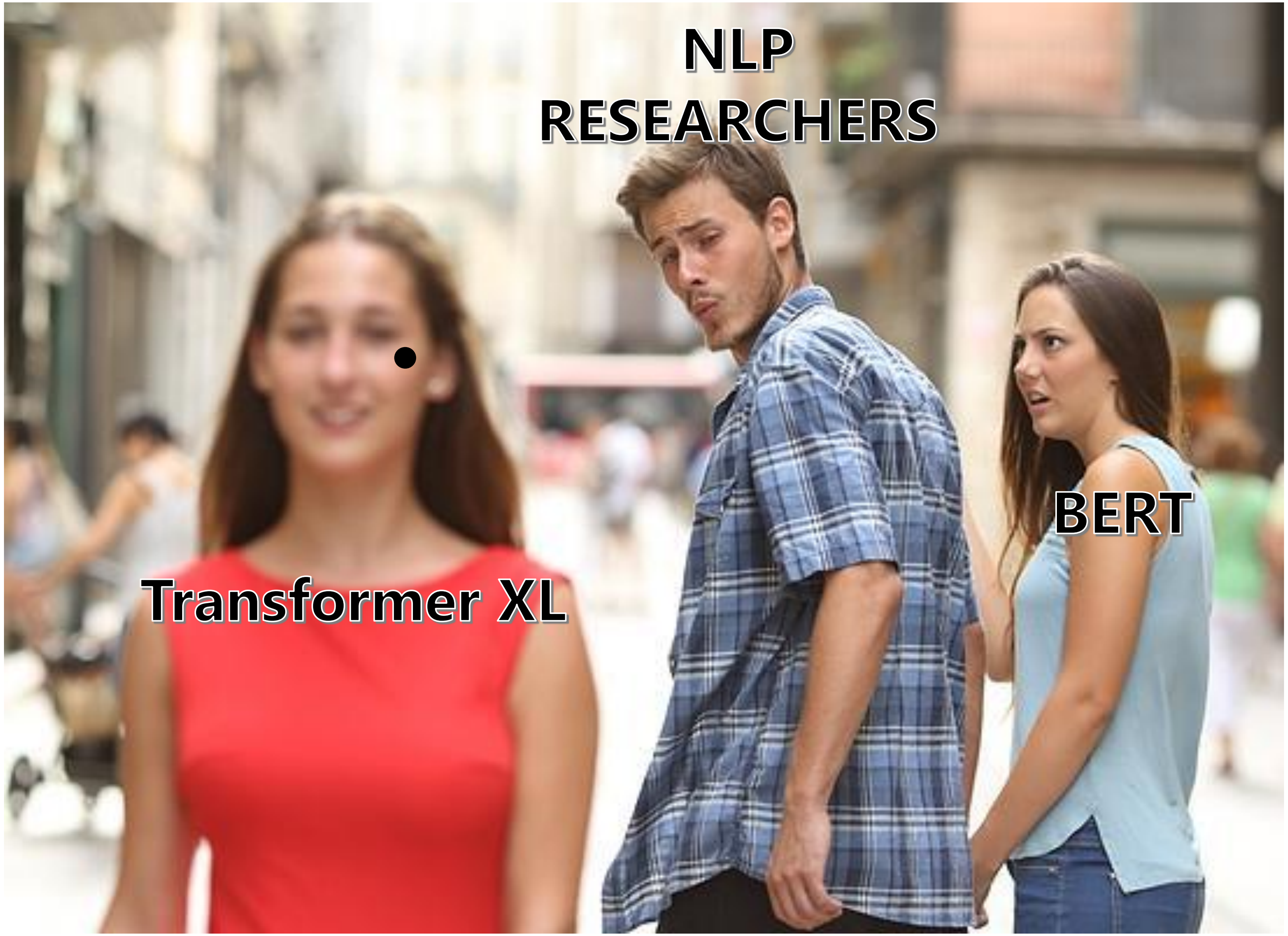
Why Transformer XL is rejected?

- 좋은 모델인건 알겠지만, downstream task에 대해서 좋은 결과를 보여줄지 미지수

- Document generation 에 관련한 결과가 없다

- 비슷한 시기에 OpenAI GPT-2 가 뛰어난 성능 입증 잘 함

Transformer XL 의 유혹



참고문헌

원문 : https://arxiv.org/pdf/1906.08237.pdf

7 : https://www.nytimes.com/2018/11/18/technology/artificial-intelligence-language.html

11 : http://jalammar.github.io/illustrated-bert/

18 : https://mlexplained.com/2019/06/30/paper-dissected-xlnet-generalized-autoregressive-pretraining-for-language-understanding-explained/

33 : https://www.youtube.com/watch?v=naOuE9gLbZo

Transformer-XL paper : https://arxiv.org/pdf/1901.02860.pdf

LAMB optimizer paper : https://arxiv.org/pdf/1904.00962.pdf