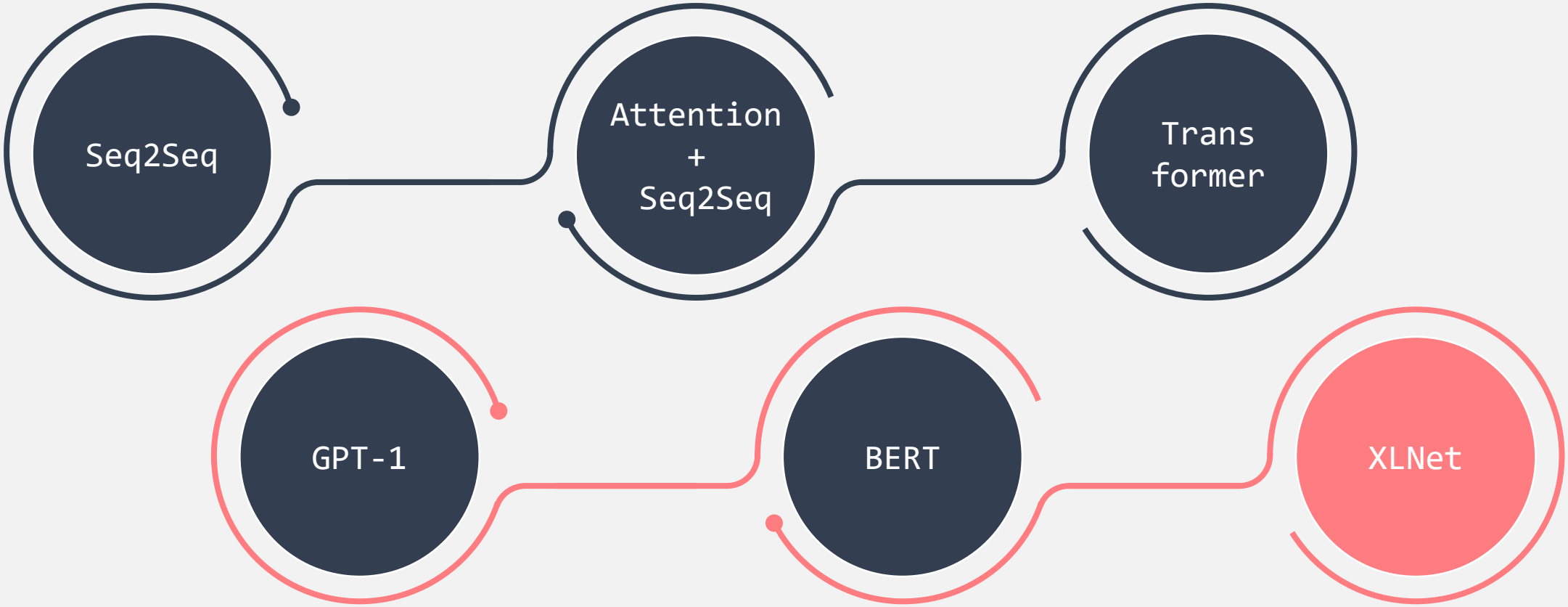
XLNet

Permutation-based AR Language Modeling

김유진, 문예진, 송경민, 이상민, 한유경

NLP 2조

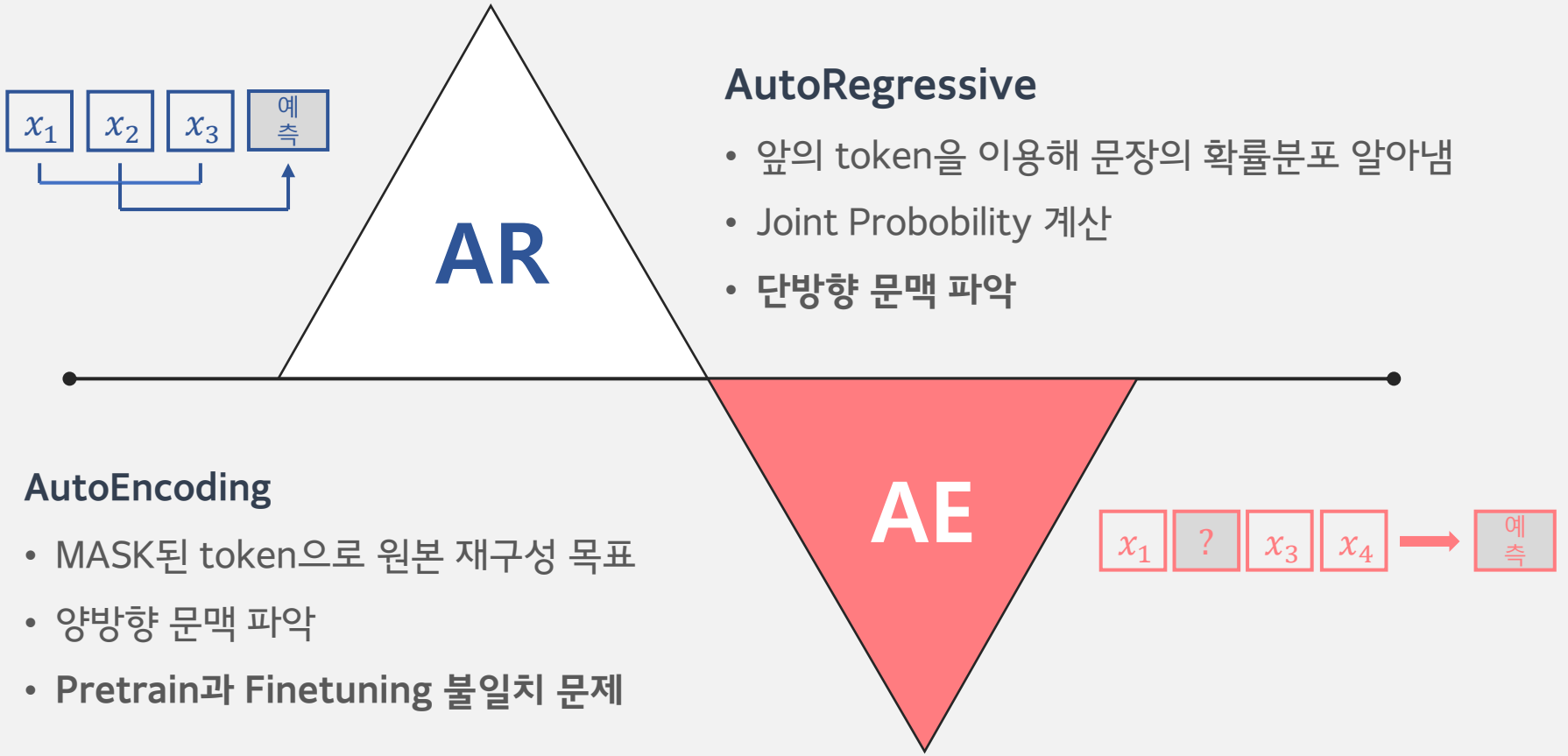
Models Review



XLNet

Permutation-based AR Language Modeling

AR과 AE를 최대한 활용할 수 있는 generalized AR method



AutoRegressive

- 앞의 token을 이용해 문장의 확률분포 알아냄
- Joint Probobility 계산
- **단방향 문맥 파악**

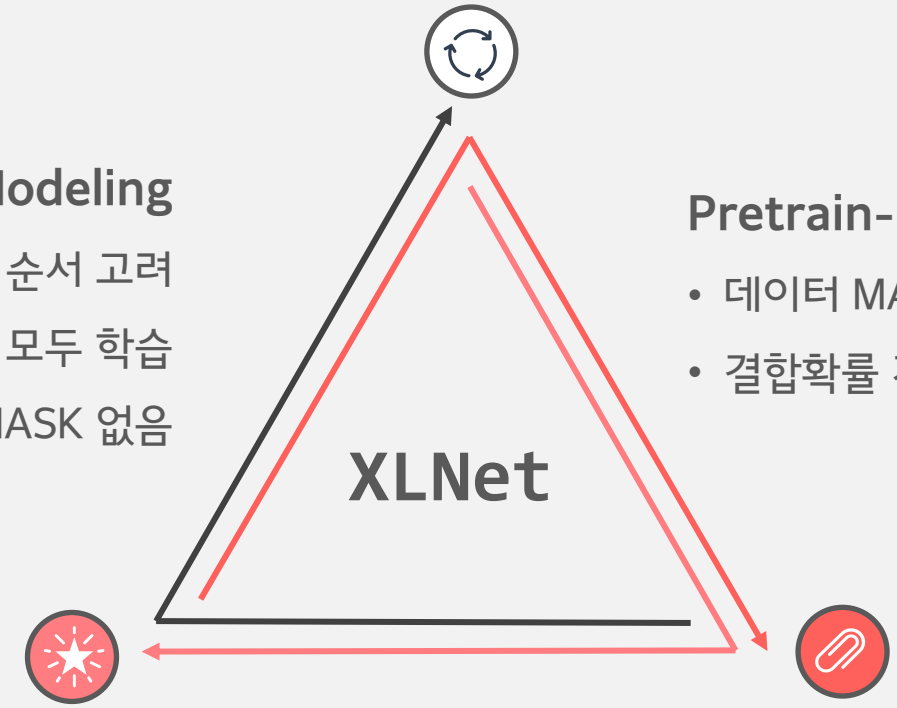
AutoEncoding

- MASK된 token으로 원본 재구성 목표
- 양방향 문맥 파악
- **Pretrain과 Finetuning 불일치 문제**

XLNet

Permutation-based AR Language Modeling

AR과 AE를 최대한 활용할 수 있는 generalized AR method



Permutation Language Modeling

- 가능한 모든 token 순서 고려
- 양방향 문맥 모두 학습
- MASK 없음

Pretrain-Finetune 일치

- 데이터 MASK에 의존하지 않음
- 결합확률 계산 가능 = token의 의존성 확인 가능

Transformer-XL 활용

- segment recurrence mechanism & relative encoding scheme 적용
- Target-Aware Representations

XLNet

Permutation-based AR Language Modeling

Objective: Permutation Language Modeling

Input Sequence (T)

x1 x2 x3 x4

모든 permutation 집합
4! = 24

모든 permutation 집합 (Zt)

[[3,2,4,1],[2,4,3,1],[1,4,2,3]
,…,[4,3,1,2]]

다양한 sequence 고려

likelihood

$$\mathbb{E}_{z}^{Z_T}\left[\prod_{t=1}^{T} p(x_t|\mathbf{x}_{<t})\right]$$

AR Objective function에 대입

Objective function

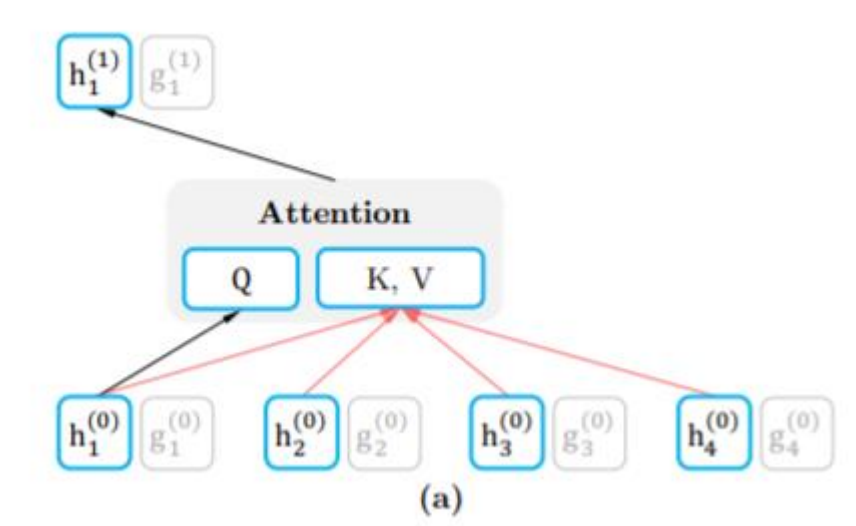
$$Max_{\theta}\ \mathbb{E}_{z}^{Z_T}\left[\sum_{t=1}^{T} logp_{\theta}(x_t|\mathbf{x}_{<t})\right]$$

특정 토큰에 양방향 context 고려 가능

Architecture : Two-Stream Self-Attention for Target-Aware Representations

Two-Stream Self Attention = 1 Content Stream + 2 Query Stream

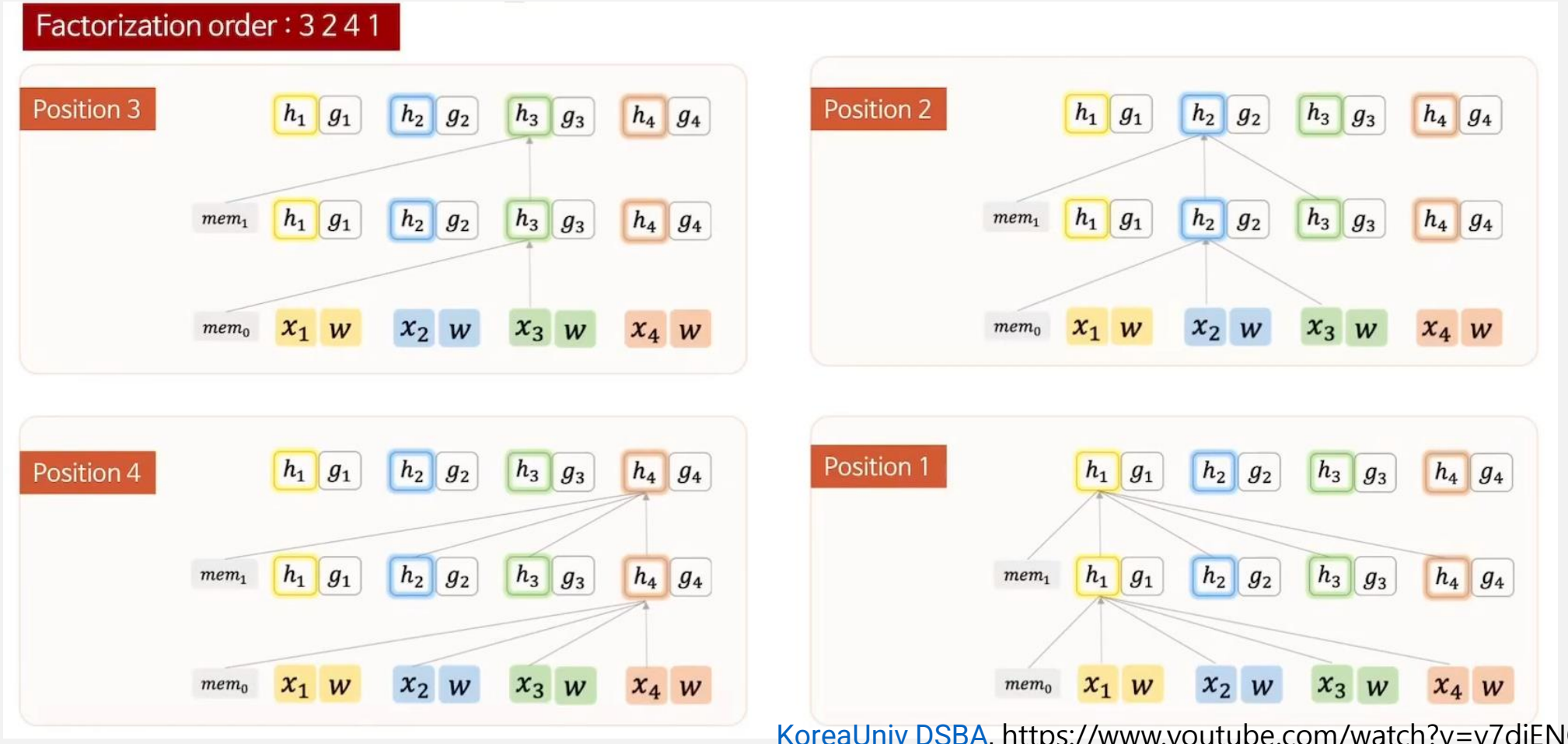
1 Content Stream : 현재 시점과 현재 시점 이전의 토큰 정보 활용



$$h_{z_t}^{(m)} \leftarrow \text{Attention}(\mathbf{Q} = h_{z_t}^{(m-1)}, \mathbf{KV} = \mathbf{h}_{\mathbf{z}_{\leq t}}^{(m-1)}; \theta), \quad \text{(content stream: use both } z_t \text{ and } x_{z_t}\text{)}.$$

Architecture : Two-Stream Self-Attention for Target-Aware Representations

1 Content Stream : 현재 시점과 현재 시점 이전의 토큰 정보 활용



KoreaUniv DSBA, https://www.youtube.com/watch?v=v7diENO2mEA

Architecture : Two-Stream Self-Attention for Target-Aware Representations

Input sequence [x1, x2, x3, x4]

Index의 permutation
Z = [[1,2,3,4], [1,3,2,4], … [4,3,2,1]] 학습

[2, 3, 1, 4]의 경우 p(x1|x2, x3) -> hθ(x2, x3)

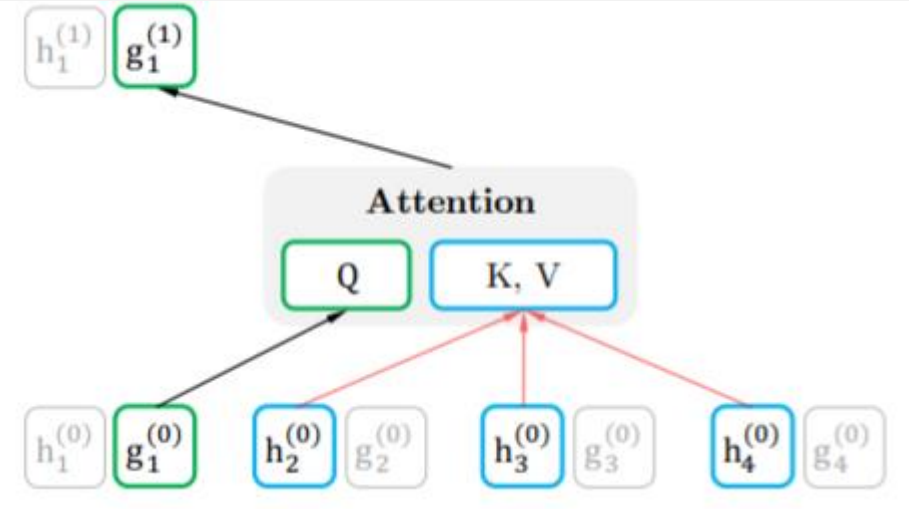
[2, 3, 4, 1]의 경우 p(x4|x2, x3) -> hθ(x2, x3)

위치가 다른데 같은 representation

Position 정보가 필요

Architecture : Two-Stream Self-Attention for Target-Aware Representations

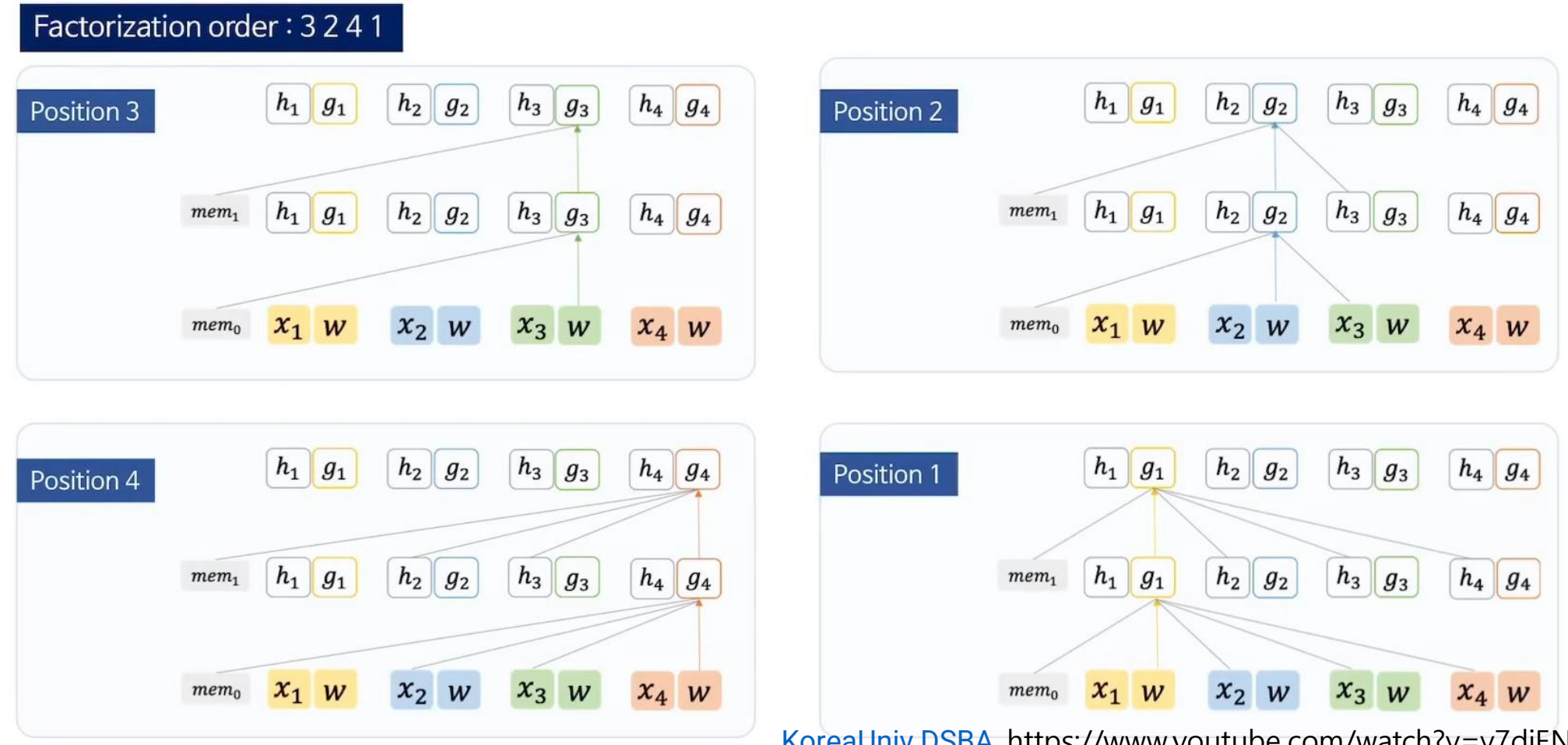
2 Query Stream : 이전의 토큰정보 + 현재의 위치 정보



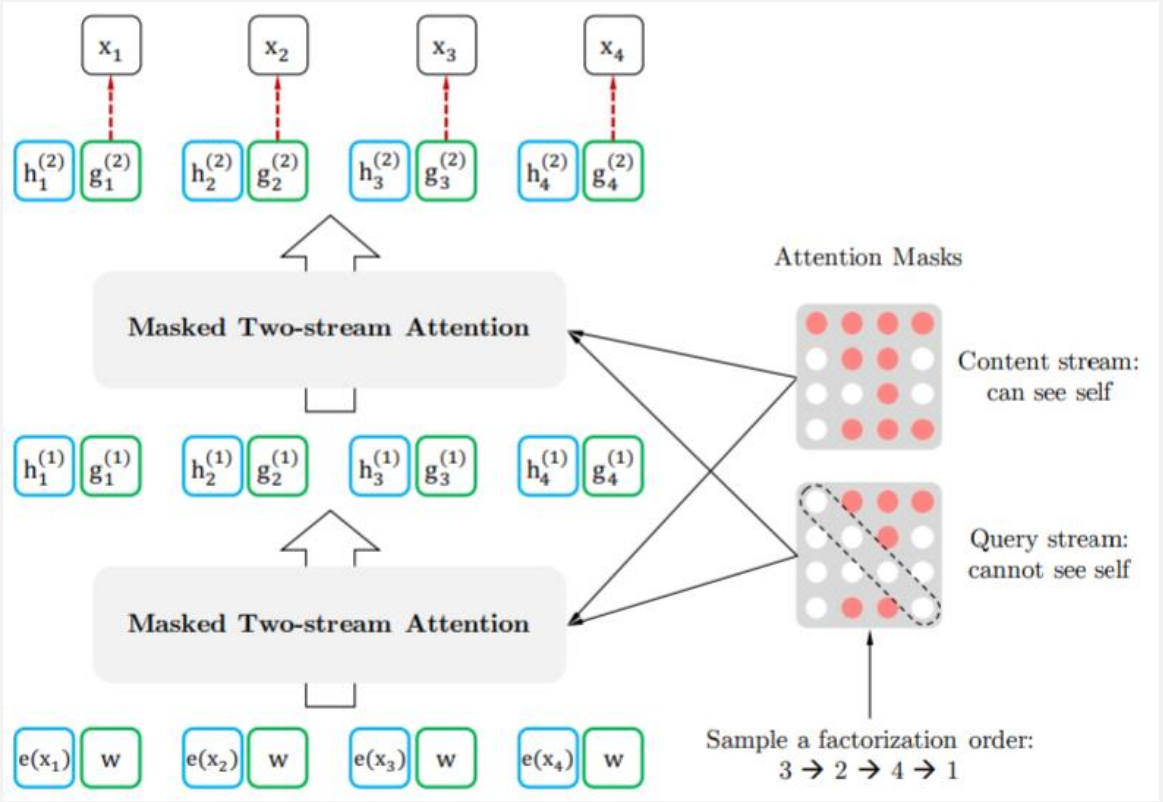
$$g_{z_t}^{(m)} \leftarrow \text{Attention}(\mathrm{Q} = g_{z_t}^{(m-1)}, \mathrm{KV} = \mathbf{h}_{\mathbf{z}<t}^{(m-1)}; \theta), \quad \text{(query stream: use } z_t \text{ but cannot see } x_{z_t})$$

Architecture : Two-Stream Self-Attention for Target-Aware Representations

2 Query Stream : 이전의 토큰정보 + 현재의 위치 정보



Architecture : Two-Stream Self-Attention for Target-Aware Representations



- Content Stream으로부터 hidden representation
  - Query Stream으로부터 토큰의 위치정보

Architecture : Two-Stream Self-Attention for Target-Aware Representations

Partial Prediction : Pretrain의 계산량을 줄이기 위한 방법
입력값의 부분 집합만을 학습

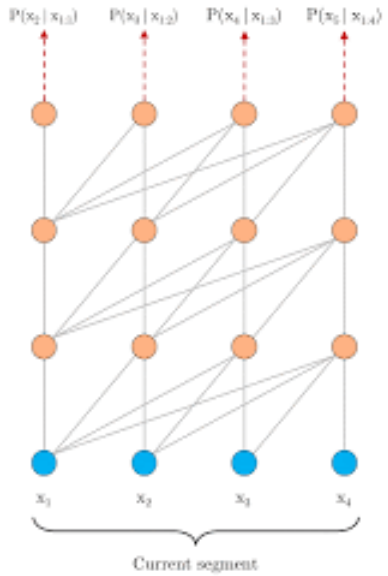
$$\max_{\theta} \quad \mathbb{E}_{\mathbf{z}\sim\mathcal{Z}_T}\left[\log p_\theta(\mathbf{x}_{\mathbf{z}_{>c}} \mid \mathbf{x}_{\mathbf{z}_{\le c}})\right] = \mathbb{E}_{\mathbf{z}\sim\mathcal{Z}_T}\left[\sum_{t=c+1}^{|\mathbf{z}|} \log p_\theta(x_{z_t} \mid \mathbf{x}_{\mathbf{z}_{<t}})\right]$$

[3,2,4,1] 에서 K = 2일 경우

$$p(x_3)p(x_2 \mid x_3)p(x_4 \mid x_2, x_3)p(x_1 \mid x_3, x_2, x_4) \rightarrow p(x_4 \mid x_2, x_3)p(x_1 \mid x_3, x_2, x_4)$$

Incorporating Ideas from Transformer-XL

1 segment recurrence mechanism + 2 relative positional encoding scheme



Incorporating Ideas from Transformer-XL

1 segment recurrence mechanism + 2 relative positional encoding scheme

- 이전 segment를 재사용할 때, positional encoding을 정의하는 방법

[1,2,3,4] segment1 [1,2,3,4] segment2 [1,2,3,4] segment3

- Standard Transformer

$$\mathbf{A}^{\text{abs}}_{i,j} = q_i^\top k_j = \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{U}_j}_{(b)} + \underbrace{\mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j}}_{(c)} + \underbrace{\mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{U}_j}_{(d)}.$$

- Transformer-XL

$$\mathbf{A}^{\text{rel}}_{i,j} = \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_{k,E} \mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_{k,R} \mathbf{R}_{i-j}}_{(b)} + \underbrace{u^\top \mathbf{W}_{k,E} \mathbf{E}_{x_j}}_{(c)} + \underbrace{v^\top \mathbf{W}_{k,R} \mathbf{R}_{i-j}}_{(d)}.$$

Experiments : Pretraining and Implementation

- Pretraining Dataset
  - Bert : BookCorpus + English Wikipedia
  - XLNet : BookCorpus + English Wikipedia + Giga5 + ClubWeb + Common Crawl

- Bert보다 10배 많은 데이터 사용

- Model size
  - XLNet-Large는 BERT-Large와 같은 구조의 하이퍼파라미터 매개 변수를 가짐
    → 비슷한 모델 사이즈

Experiments : Fair Comparison with BERT

Model	SQuAD1.1	SQuAD2.0	RACE	MNLI	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B
BERT-Large (Best of 3)	86.7/92.8	82.8/85.5	75.1	87.3	93.0	91.4	74.0	94.0	88.7	63.7	90.2
XLNet-Large-wikibooks	88.2/94.0	85.1/87.8	77.4	88.4	93.9	91.8	81.2	94.4	90.0	65.2	91.1

BERT와 XLNet을 공정한 환경에서 비교
→ 모든 데이터셋에서 BERT를 능가함!

Experiments : Comparison with RoBERTa: Scaling Up

RACE	Accuracy	Middle	High	Model	NDCG@20	ERR@20
GPT [28]	59.0	62.9	57.4	DRMM [13]	24.3	13.8
BERT [25]	72.0	76.6	70.1	KNRM [8]	26.9	14.9
BERT+DCMN* [38]	74.1	79.5	71.8	Conv [8]	28.7	18.1
RoBERTa [21]	83.2	86.5	81.8	BERT†	30.53	18.67
XLNet	**85.4**	**88.6**	**84.0**	XLNet	**31.10**	**20.28**

RACE Dataset

Model	IMDB	Yelp-2	Yelp-5	DBpedia	AG	Amazon-2	Amazon-5
CNN [15]	-	2.90	32.39	0.84	6.57	3.79	36.24
DPCNN [15]	-	2.64	30.58	0.88	6.87	3.32	34.81
Mixed VAT [31, 23]	4.32	-	-	0.70	4.95	-	-
ULMFiT [14]	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	**3.20**	**1.37**	**27.05**	**0.60**	**4.45**	**2.11**	**31.67**

Text Classification

SQuAD2.0	EM	F1	SQuAD1.1	EM	F1
Dev set results (single model)					
BERT [10]	78.98	81.77	BERT† [10]	84.1	90.9
RoBERTa [21]	86.5	89.4	RoBERTa [21]	88.9	94.6
XLNet	**87.9**	**90.6**	XLNet	**89.7**	**95.1**
Test set results on leaderboard (single model, as of Dec 14, 2019)					
BERT [10]	80.005	83.061	BERT [10]	85.083	91.835
RoBERTa [21]	86.820	89.795	BERT* [10]	87.433	93.294
XLNet	**87.926**	**90.689**	XLNet	**89.898‡**	**95.080‡**

SQuAD Dataset

Model	MNLI	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B	WNLI
Single-task single models on dev									
BERT [2]	86.6/-	92.3	91.3	70.4	93.2	88.0	60.6	90.0	-
RoBERTa [21]	90.2/90.2	94.7	92.2	**86.6**	96.4	**90.9**	68.0	92.4	-
XLNet	**90.8/90.8**	**94.9**	**92.3**	85.9	**97.0**	90.8	**69.0**	**92.5**	-
Multi-task ensembles on test (from leaderboard as of Oct 28, 2019)									
MT-DNN* [20]	87.9/87.4	96.0	89.9	86.3	96.5	92.7	68.4	91.1	89.0
RoBERTa* [21]	90.8/90.2	98.9	90.2	88.2	96.7	92.3	67.8	92.2	89.0
XLNet*	**90.9/90.9†**	**99.0†**	**90.4†**	**88.5**	**97.1†**	**92.9**	**70.2**	**93.0**	**92.5**

GLUE Dataset

Experiments : Ablation Study

#	Model	RACE	SQuAD2.0 F1	SQuAD2.0 EM	MNLI m/mm	SST-2
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

Conclusions

- XLNet은 AR pretrainig 방법을 일반화한 것
  & Permutation language modeling objective를 사용해 AR과 AE의 장점을 결합한 것
- XLNet의 구조는 AR objective을 작업하는데 적용되고
  Transformer-XL와 two-stream attention mechanism을 결합하여 설계됨
- 다양한 작업에서 이전 pretraining 목표들보다 상당한 개선을 달성