XLNet: Generalized Autoregressive Pretraining for Language Understanding

모두의 연구소 풀잎스쿨

NLP bootcamp 5th 박성찬

2019. 11. 30(sat)

I. Introduction: XLNet

- A SOTA pretrained model for language understanding
- Generalized AR pretraining model
  - Permutation language modeling objectives
  - two-stream attention mechanism
- 여러 NLP task에서 original BERT 대비 우수한 성능
- 논문 이외의 여러 블로그 참고
  - https://novdov.github.io/machnielearning/nlp/2019/07/13/XLNet-%EB%A6%AC%EB%B7%B0/
  - https://ratsgo.github.io/natural%20language%20processing/2019/09/11/xlnet/
  - https://blog.pingpong.us/xlnet-review/
  - https://ai-information.blogspot.com/2019/07/nl-041-xlnet-generalized-autoregressive.html

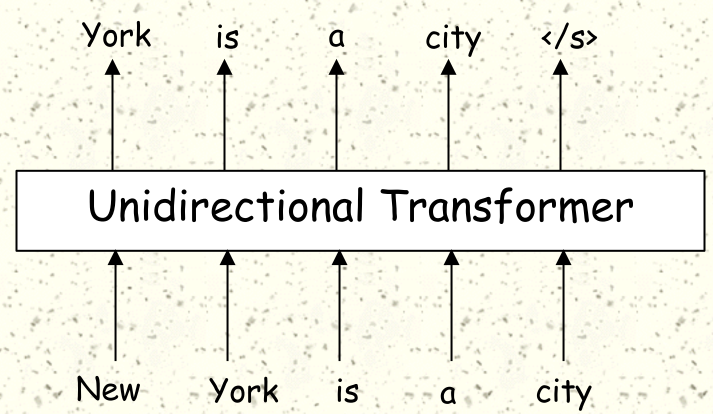
I. Introduction: Pretraining

- How it works
  - Pretrain a model on unlabeled data based on language modeling
  - Finetune the model or use the model for feature extraction on downstream tasks

Word2vec(Mikolov et al.), GloVe(Pennington et), Semi-supervised sequence learning(Dai & Le), ELMo(Peters, et al.), CoVe(McCann et al.), GPT(Radford et al.), BERT(Devlin et al.)

2.1 Background: Two Objectives for Pretraining

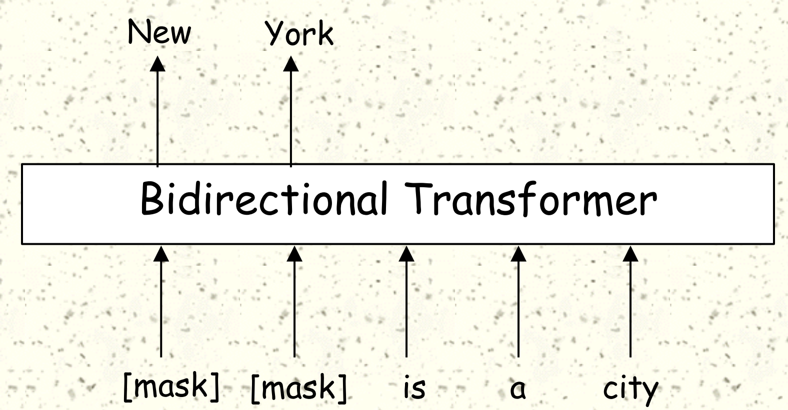
Auto-regressive(AR) language modeling



$$\log p(\mathrm{x}) = \sum_{t=1}^{T} \log p(x_t | \mathrm{x}_{<t})$$

- 양방향 context 학습 불가

(Denoising) Auto-encoding(AE)



$$\log p(\mathrm{x}^{-} | \mathrm{x}^{\wedge}) = \sum_{t=1}^{T} mask_t \log p(x_t | \mathrm{x}^{\wedge})$$

- Masked token들 사이는 독립이라는 가정
- Finetuning과정에는 mask token 없음

2.2 Objective: Permutation Language Modeling(PLM)

- Sample a factorization order z
- Determine the attention masks based on the order
- Optimize a standard language modeling objective

$$\mathbb{E}_{z \sim \mathcal{Z}_T}\left[\sum_{t=1}^{T} \log p(x_{z_t} | \mathbf{x}_{z<t})\right]$$

Ex.) *p(a, b)=p(a)p(b/a)*

=p(b)p(a/b)

2.2 PLM example

- Sentence: New York is a city
- Factorization order: New York is a city

p(New York is a city) =p(New)*p(York|New)*p(is|New York)*p(a|New York is)*p(city|New York is a)

- Factorization order: city a is New York

p(New York is a city) =p(city)*p(a|city)*p(is|city, a)*p(New|city a is)*p(York|city a is New)

- Remarks on Permutation
  - Sequence order is not shuffled: 기존 sequence order는 유지하고 positional encoding만 바꾸어 transformer의 attention mask에 적용
  - Attention masks are changed to reflect factorization order

2.2 PLM illustration

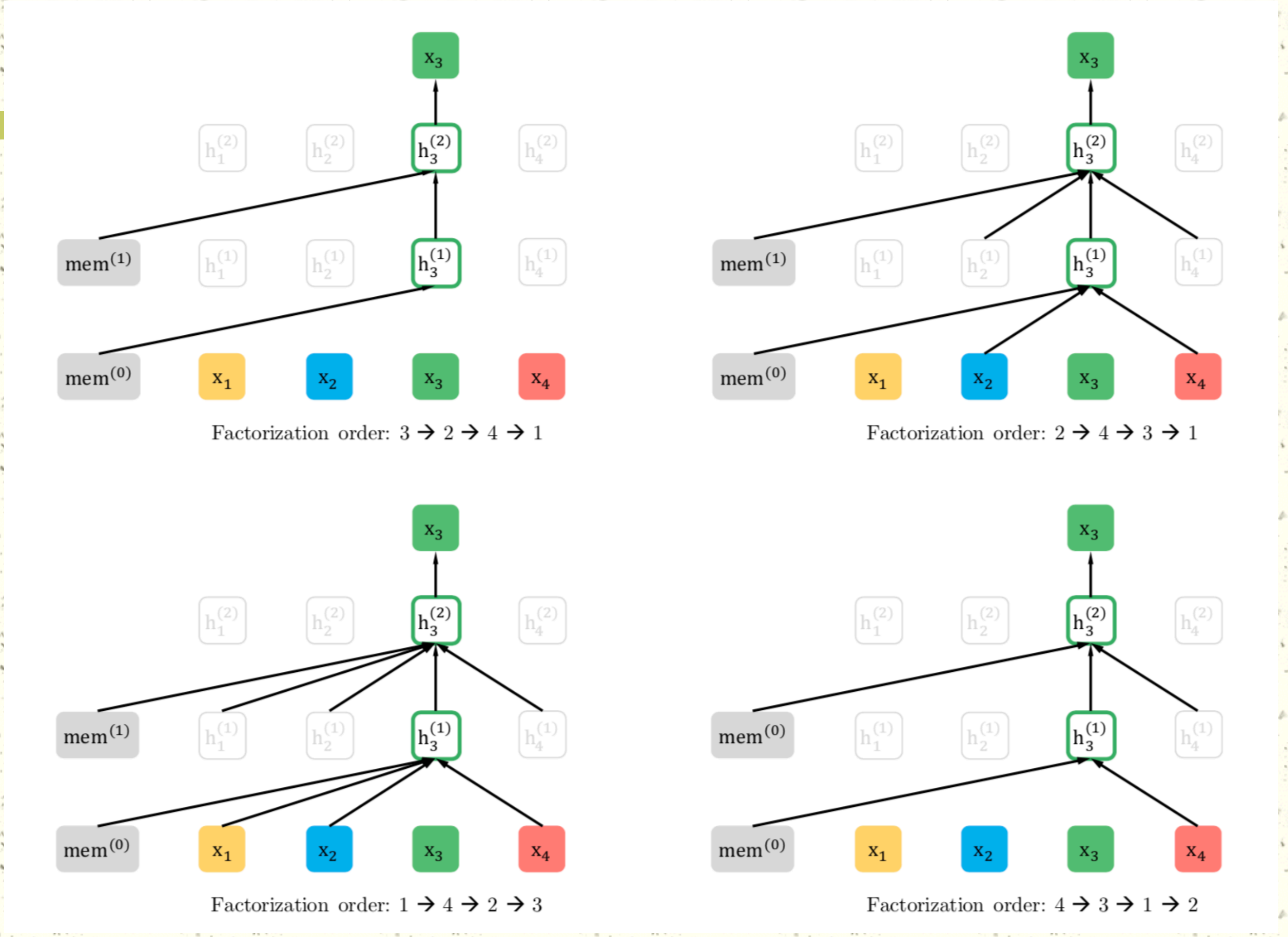


Fig1: Illustration of the permutation language modeling(PLM) objective for predicting x_3, given the same input sequence x but with different factorization orders

2.2 PLM objectives

- Not only the benefits of AR models, but capture bidirectional contexts

$\mathbb{Z}_t$ is set of all possible permutations of length-t index seq. [1,2,…,T]

z_t is t-th element and $\mathbf{z} < t$ is the first t-1 elements of a permutation $\mathbf{z} \in \mathbb{Z}_t$

$$\max_{\theta} \mathbb{E}_{\mathbf{z}\sim \mathcal{Z}_{\mathrm{T}}} \left[\sum_{t=1}^{T} log p_{\theta}\left(x_{z_t} \middle| \mathrm{x}_{\mathbf{z}<t}\right) \right] (3)$$

For text sequence x, sample a factorization order z and decompose $p_{\theta}(x)$

Same model parameter θ is shared across all factorization orders during training, it could possibly capture the bidirectional context

2.3 Two-Stream Self-Attention for Target-Aware Representation: Reparameterization

- Standard Parameterization

$$p_\theta(X_{z_t} = x | \mathbf{x}_{\mathbf{z}<t}) = \frac{e(x)^T h_\theta(x_{\mathbf{z}<t})}{\sum_{x'} e(x')^T h_\theta(x_{\mathbf{z}<t})}$$

h does not contain the position of the target

For input sequence [x_1, x_2, x_3, x_4], its permutation $Z_T = [[1,2,3,4],[1,3,2,4], \ldots,, [4,3,2,1]]$

[2,3,1,4]: $p(x_1 | \, x_2, x_3)$

[2,3,4,1]: $p(x_4 | \, x_2, x_3)$

-> Ambiguity!

- Solutions: condition the distribution on the position

$$p_\theta(X_{z_t} = x | \mathbf{x}_{\mathbf{z}<t}) = \frac{e(x)^T g_\theta(x_{\mathbf{z}<t}, z_t)}{\sum_{x'} e(x')^T g_\theta(x_{\mathbf{z}<t}, z_t)}$$

-> But how to compute $g_\theta(x_{\mathbf{z}<t}, z_t)$ effectively?

2.3 Two-Stream Self-Attention for Target-Aware Representation: Two-Stream Self-Attention

- If predict token X_{z_t}, $g_\theta(x_{\mathbf{z}<t}, z_t)$,
  - it should use the position z_t and not the context X_{z_t} => $g_\theta(x_{\mathbf{z}<t}, z_t)$, abbreviated g_{z_t}
  - $g_{z_t}{}^{(m)} \leftarrow Attention(Q = g_{z_t}{}^{(m-1)}, \mathrm{KV} = h_{z_{<t}}{}^{(m-1)}; \theta)$
  - Query representation: access the contextual information $x_{\mathbf{z}<t}$ and position z_t but cannot see X_{z_t}
- Otherwise,
  - it should encode content X_{z_t} => $h_\theta(x_{\mathbf{z}<t})$, abbreviated h_{z_t}
  - $h_{z_t}{}^{(m)} \leftarrow Attention(Q = h_{z_t}{}^{(m-1)}, \mathrm{KV} = h_{z \le t}{}^{(m-1)}; \theta)$
  - Content representation: encodes both the context and X_{z_t}itself

2.3 Two-Stream Self-Attention for Target-Aware Representation: Illustration1

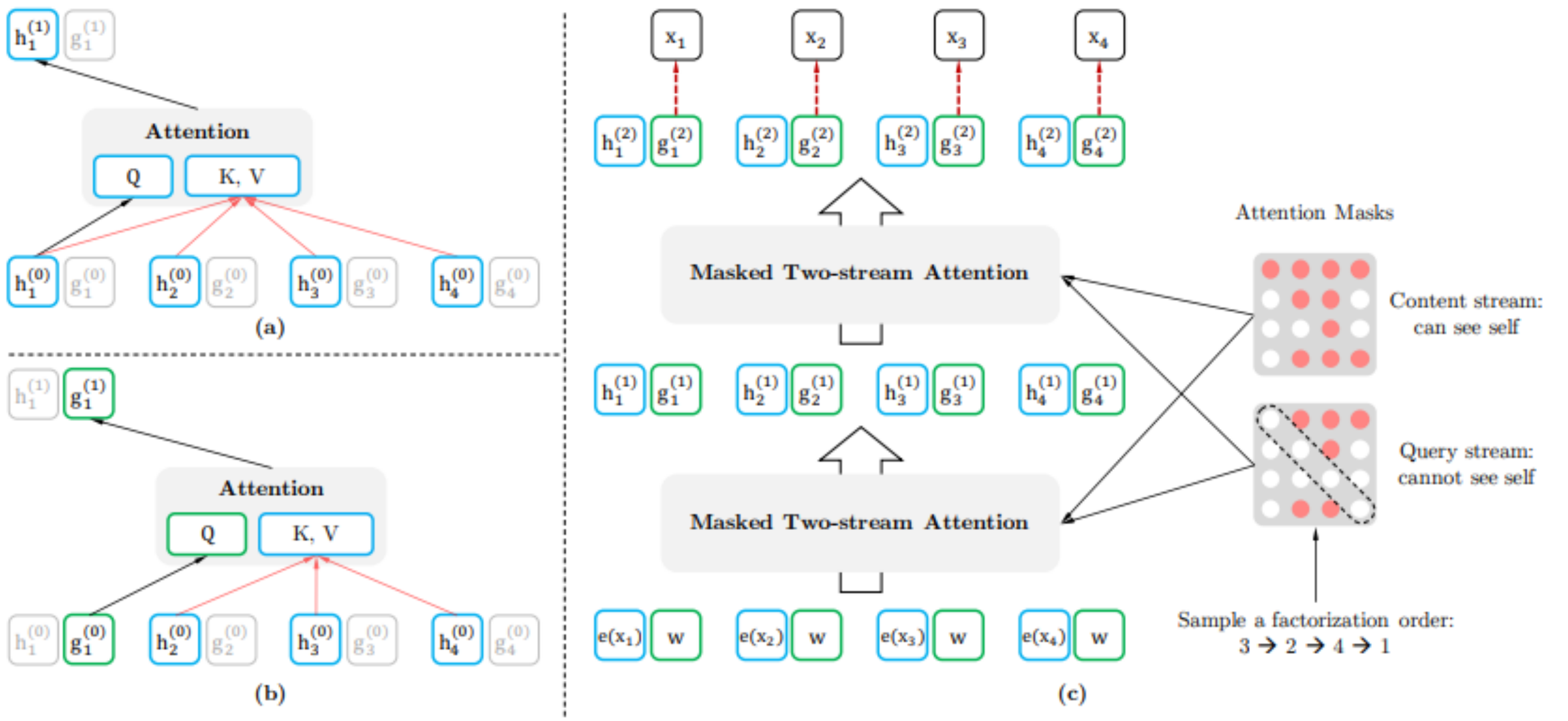
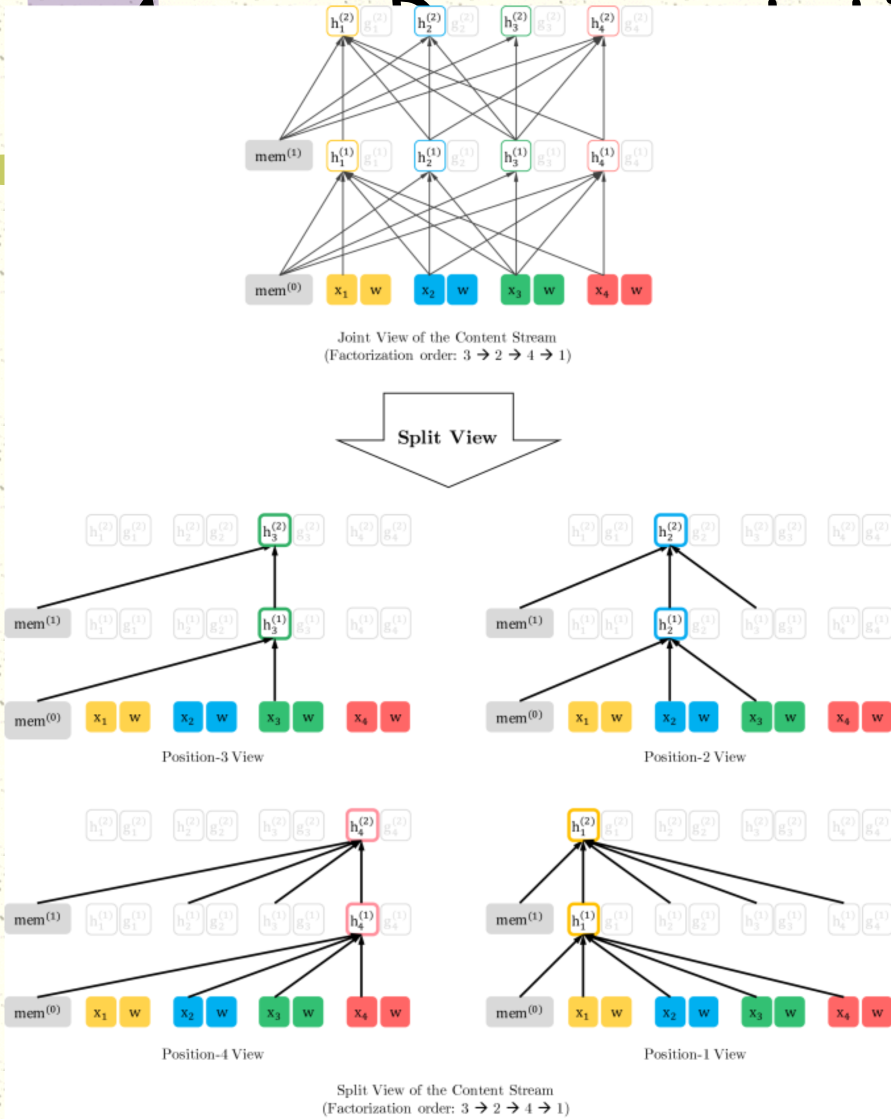
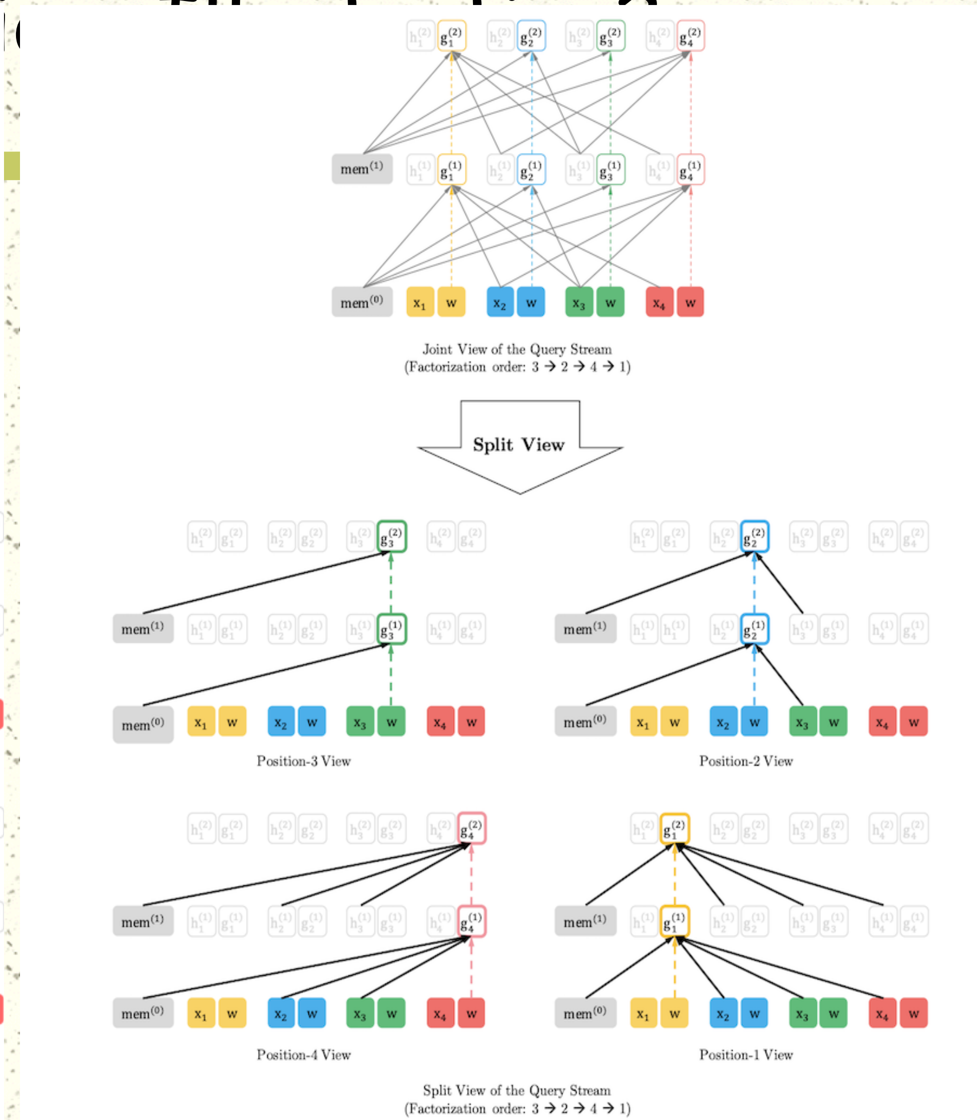


Figure 2: (a): Content stream attention, which is the same as the standard self-attention. (b): Query stream attention, which does not have access information about the content x_{z_t}. (c): Overview of the permutation language modeling training with two-stream attention.

2.3 Two-Stream Self-Attention for Target-





2.3 Two-Stream Self-Attention for Target-Aware Representation: Partial Prediction

- Several benefits, but optimization problem due to permutations
  - Only predict the last tokens in a factorization order
  - Split z into a non-target subsequence $Z_{\leq c}$ and a target subsequence $Z_{>c}$(c: cutting point)
  - $\max_{\theta} \mathbb{E}_{\mathbf{z}\sim \mathcal{Z}_{\mathrm{T}}}\left[\log p_{\theta}\left(\mathrm{x}_{Z_{>c}} \middle| \mathrm{x}_{\mathbf{z}\leq c}\right)\right]$= $\max_{\theta} \mathbb{E}_{\mathbf{z}\sim \mathcal{Z}_{\mathrm{T}}}\left[\sum_{t=c+1}^{T} \log p_{\theta}\left(x_{z_t} \middle| \mathrm{x}_{\mathbf{z}<t}\right)\right]$
  - $\mathrm{x}_{\mathbf{z}>c}$is chosen as a target because it possesses the longest context in the sequence given the current factorization order Z
  - $3 \rightarrow 2 \rightarrow 4 \rightarrow 1\ order$
    - $p(\mathrm{x}_3)p(\mathrm{x}_2|\mathrm{x}_3)p(\mathrm{x}_4|\mathrm{x}_2,\mathrm{x}_3)p(\mathrm{x}_1|\mathrm{x}_3,\mathrm{x}_2,\mathrm{x}_4) \rightarrow p(\mathrm{x}_4|\mathrm{x}_2,\mathrm{x}_3)p(\mathrm{x}_1|\mathrm{x}_3,\mathrm{x}_2,\mathrm{x}_4)$

2.4 Incorporating Ideas from Transformer-XL

- Relative positional encoding scheme
  - 여러 segment에 대해 recurrent모델링하는 경우 absolute positional encoding은 더이상 사용 불가
- Segment recurrent mechanism
  - Long sentence A divides $\tilde{x} = s_{1:T}, x = s_{T+1:2T}$and let $\tilde{z} =$ *perm[1,…,T], z=perm[T+1,…,2T]*
  - Process the first segment based on $\tilde{z}$ then cache the obtained content representations $\tilde{h}^{(m)}$ each layer m
  - $h_{z_t}{}^{(m)} \leftarrow Attention(Q = h_{z_t}{}^{(m-1)}, \mathrm{KV} = \left[\tilde{h}^{(m-1)}, h^{(m-1)}{}_{z \le t}\right]; \theta)$

2.5 Modeling Multiple Segments

- XLNet Pre-training
  - 임의로 두 segment를 뽑아 하나로 concatenation하여 PLM
  - 동일한 context에 속해 있는 memory는 재사용
  - Bert와 유사하게 [A, SEP, B, SEP, CLS] 형태(A, B는 segment)의 입력
- Relative segment encodings
  - BERT: word embedding + abs segment embedding
  - XLNet: relative segment embedding
    - Sequence의 위치 i, j에 대하여 어떠한 segment에서 왔는지는 생각 안 하고 s vector를 사용하여 동일한 segment인지만 고려
    - Finetuning 단계에서 두 개가 넘는 segment입력을 받을 수 있음
    - Inductive bias of relative encoding의 일반화?

2.6.1 Comparing with BERT: concepts

- Sentence: New York is a city
- BERT objective(auto-encoding)

$$\mathcal{J}_{BERT} = logp(\text{New}|\text{is a city}) + logp(\text{York}|\text{is a city})$$

- New와 York은 독립 사건

- XLNet objective(auto-regressive)

$$\mathcal{J}_{XLNet} = logp(\text{New}|\text{is a city}) + logp(\text{York}|\text{New, is a city})$$
$$\mathcal{J}_{XLNet} = logp(\text{New}|\text{York, is a city}) + logp(\text{York}|\text{is a city})$$

- New 와 York사이의 dependency 모델링
- Joint Probability를 factorization하여 bidirectional context 모델링
- 더 많은 dependency학습 가능("denser" effective training signals)

2.6.1 Comparing with BERT: formal expressions

- Given sequence x=[$x_1, x_2, \ldots, x_T$], target-context pair of interest $\mathcal{L} = \{(x, \mathcal{U})\}$
  $\mathcal{L} = \{(x = York|\mathcal{U} = \{\text{New}\}), (x = \text{York}|\mathcal{U} = \{\text{city}\}), (x = york, \mathcal{U} = \{\text{New}, \text{city}\}), \ldots\}$
- $\mathcal{T}$ is given target tokens, $\mathcal{N}$=x\\$\mathcal{T}$ is non-target tokens
  - Both BERT & XLNet maximize $logp(\mathcal{T}|\mathcal{N})$

$$\mathcal{J}_{BERT} = \sum_{x \in \mathcal{T}} logp(x|\mathcal{N}); \quad \mathcal{J}_{XLNet} = \sum_{x \in \mathcal{T}} logp(x|\mathcal{N} \cup \mathcal{T}_{<x})$$

  - Both has multiple loss terms $logp(x|\mathcal{V}_x), \mathcal{V}_x = \mathcal{N}$ or $\mathcal{N} \cup \mathcal{T}_{<x}$
- Two cases by definition
  - If $\mathcal{U} \in \mathcal{N}$, the dependency $(x, \mathcal{U})$ covered by both
  - If $\mathcal{U} \in \mathcal{N} \cup \mathcal{T}_{<x}$ and $\mathcal{N} \cup \mathcal{T}_{<x} \neq \emptyset$, the dependency $(x, \mathcal{U})$ covered only be XLNet

2.6.2 Comparing with language modeling

- Conventional AR language modeling
  - Context: "Thom Yorke is the singer of Radiohead"
  - Question: "Who is the singer of Radiohead"
  - Answer: "Thom York"

  - AR은 뒤에 나오는 문맥과의 dependency 학습이 어려움
- Formal expression
  - Consider a context-target pair $(x, \mathcal{U})$
  - If $\mathcal{U} \cap \mathcal{T}_{<x} \neq \emptyset$, where $\mathcal{T}_{<x}$ denotes tokens prior to x in the original sentence, AR cannot able to cover the dependency

  - XLNet은 모든 dependency 학습 가능

3.1 Pretraining and Implementation

- Dataset
  - BookCorpus + English Wikipedia = 13GB
  - Giga5=16GB
  - Clue Web2012-B+Common Crawl dataset = 19GB, 78GB
- Tokenization
  - Using sentence piece, it sums total 32.89B subword tokens
- Pretraining
  - Sequence length = 512, memory length = 384
  - XLNet large= 512 TPU v3 chips for 500k steps with Adam optimizer, batch size 2048, about 2.5 days

3.2-3.3 Race & SQuAD Dataset

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.

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.

3.4-3.5 Text Classification & GLUE dataset

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).

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	-
XLNet	**89.8/-**	**93.9**	**91.8**	**83.8**	**95.6**	**89.2**	**63.6**	**91.8**	-
Single-task single models on test									
BERT [10]	86.7/85.9	91.1	89.3	70.1	94.9	89.3	60.5	87.6	65.1
Multi-task ensembles on test (from leaderboard as of June 19, 2019)									
Snorkel* [29]	87.6/87.2	93.9	89.9	80.9	96.2	91.5	63.8	90.1	65.1
ALICE*	88.2/87.9	95.7	**90.7**	83.5	95.2	92.6	**68.6**	91.1	80.8
MT-DNN* [18]	87.9/87.4	96.0	89.9	**86.3**	96.5	92.7	68.4	91.1	89.0
XLNet*	**90.2/89.7**†	**98.6**†	90.3†	**86.3**	**96.8**†	**93.0**	67.8	**91.6**	**90.4**

Table 4: Results on GLUE. ∗ indicates using ensembles, and † denotes single-task results in a multi-task row. All results are based on a 24-layer architecture with similar model sizes (aka BERT-Large). See the upper-most rows for direct comparison with BERT and the lower-most rows for comparison with state-of-the-art results on the public leaderboard.

3.6-3.7 ClueWeb09-B dataset & Ablation study

Model	NDCG@20	ERR@20
DRMM [12]	24.3	13.8
KNRM [8]	26.9	14.9
Conv [8]	28.7	18.1
BERT†	30.53	18.67
XLNet	**31.10**	**20.28**

Table 5: Comparison with state-of-the-art results on the test set of ClueWeb09-B, a document ranking task. † indicates our implementations.

#	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.

4. Conclusion

- XLNet is a generalized AR pretraining method
- Permutation language modeling
  - Combine the advantages of AR and AE methods
- XLNet architecture
  - Transformer XL & two-stream attention mechanism
- XLNet achieves SOTA results from various tasks