XLNet: Generalized Autoregressive Pretraining for Language Understanding

Zhilin Yang, Zhang Dai et al.
CMU, Google Brain

Kyung-Jae Cho VUNO Inc.

vuno

XLNet outperforms BERT

Quoc Le @quocleix · 6월 20일

XLNet: a new pretraining method for NLP that significantly improves upon BERT on 20 tasks (e.g., SQuAD, GLUE, RACE)

arxiv: arxiv.org/abs/1906.08237

github (code + pretrained models): github.com/zihangdai/xlnet

with Zhilin Yang, @ZihangDai, Yiming Yang, Jaime Carbonell, @rsalakhu

트윗 번역하기

Results

As of June 19, 2019, XLNet outperforms BERT on 20 tasks and achieves state-of-the-art results on 18 tasks. Below are some comparison between XLNet-Large and BERT-Large, which have similar model sizes:

Results on Reading Comprehension

Model	RACE accuracy	SQuAD1.1 EM	SQuAD2.0 EM
BERT	72.0	84.1	78.98
XLNet	**81.75**	**88.95**	**86.12**

We use SQuAD dev results in the table to exclude other factors such as using additional training data or other data augmentation techniques. See SQuAD leaderboard for test numbers.

Results on Text Classification

Model	IMDB	Yelp-2	Yelp-5	DBpedia	Amazon-2	Amazon-5
BERT	4.51	1.89	29.32	0.64	2.63	34.17
XLNet	**3.79**	**1.55**	**27.80**	**0.62**	**2.40**	**32.26**

The above numbers are error rates.

Results on GLUE

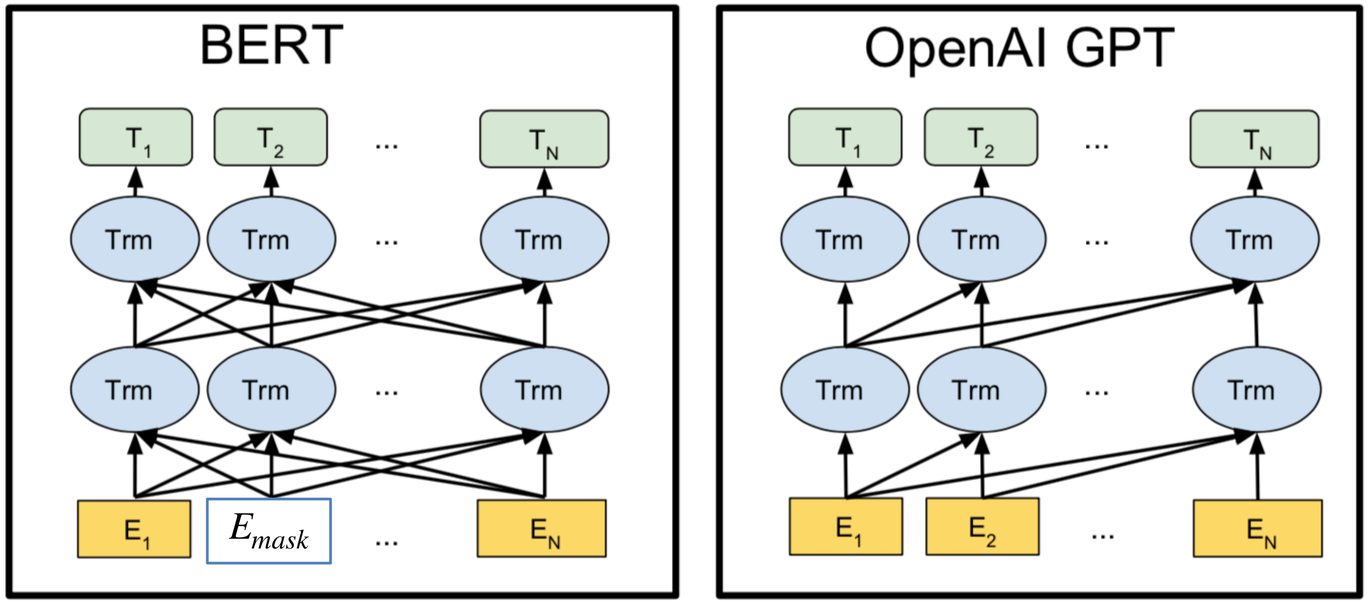
Model	MNLI	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B
BERT	86.6	92.3	91.3	70.4	93.2	88.0	60.6	90.0
XLNet	**89.8**	**93.9**	**91.8**	**83.8**	**95.6**	**89.2**	**63.6**	**91.8**

We use single-task dev results in the table to exclude other factors such as multi-task learning or using ensembles.

23 753 1,848

Existing Pretraining Models

Auto-encoding **Auto-regressive**



- Auto-encoding (e.g., BERT)
  - Randomly set a portion (e.g., 15%) of tokens in x to [MASK] token
  - Training objective is to reconstruct [MASK] token to original token
- Auto-regressive (e.g., GPT, GPT2)
  - Maximize the likelihood under the forward autoregressive factorization (Language Model)

Pros and Cons of two pertaining objectives

	BERT	GPT
Independence Assumption	Factorizes joint conditional probability based on an i**ndependence assumption that all masked tokens are separately reconstructed**	No independent Assumption
Input Noise	Pretrain-finetune discrepancy	No discrepancy
Context Dependency	Bi-directional	Uni-directional

- Independence Assumption
  - Neglects dependency between the masked positions

GPT

$$\max_{\theta} \quad \log p_\theta(\mathbf{x}) = \sum_{t=1}^{T} \log p_\theta(x_t \mid \mathbf{x}_{<t}) = \sum_{t=1}^{T} \log \frac{\exp\left(h_\theta(\mathbf{x}_{1:t-1})^\top e(x_t)\right)}{\sum_{x'} \exp\left(h_\theta(\mathbf{x}_{1:t-1})^\top e(x')\right)},$$

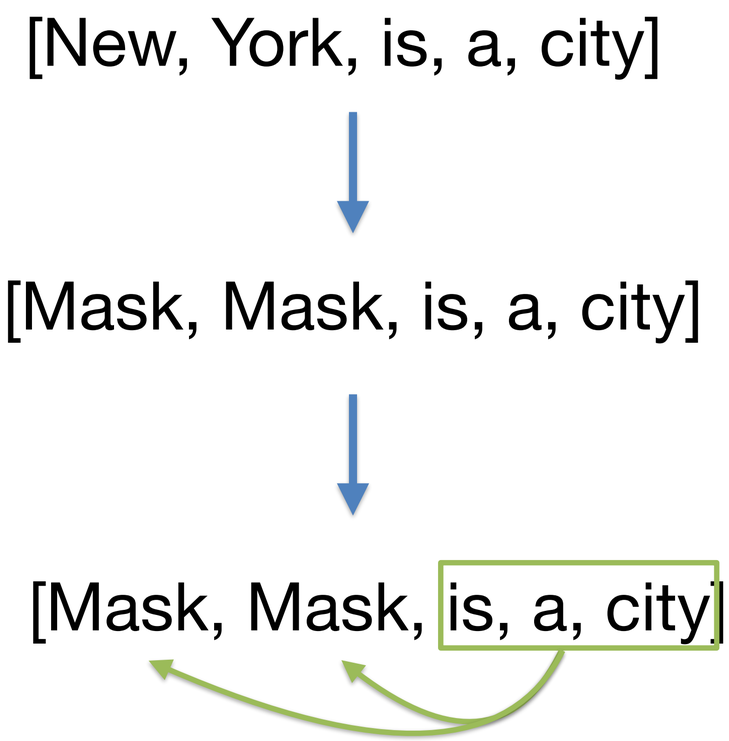
BERT

$$\max_{\theta} \quad \log p_\theta(\bar{\mathbf{x}} \mid \hat{\mathbf{x}}) \approx \sum_{t=1}^{T} m_t \log p_\theta(x_t \mid \hat{\mathbf{x}}) = \sum_{t=1}^{T} m_t \log \frac{\exp\left(H_\theta(\hat{\mathbf{x}})_t^\top e(x_t)\right)}{\sum_{x'} \exp\left(H_\theta(\hat{\mathbf{x}})_t^\top e(x')\right)},$$

VUNO

Independence Assumption

Neglects dependency between the masked positions?



Pros and Cons of two pertaining objectives

	BERT	GPT
Independence Assumption	Factorizes joint conditional probability based on an i**ndependence assumption that all masked tokens are separately reconstructed**	No independent Assumption
Input Noise	Pretrain-finetune discrepancy	No discrepancy
Context Dependency	Bi-directional	Uni-directional

- Input Noise
  - The input to BERT contains artificial symbols like [MASK] that never occur in downstream tasks
- Context Dependency
  - BERT has access to the contextual information on both sides

Proposed Method

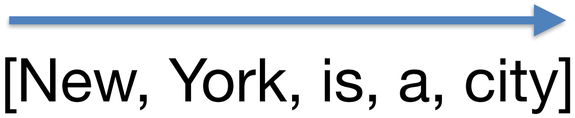
Permutation Language Model

Two-Stream Self Attention

Transformer-XL

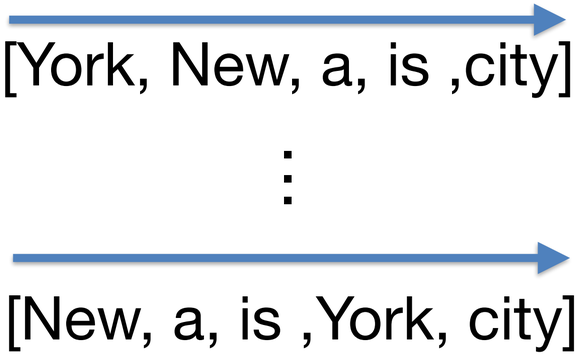
Objective: Permutation Language Model

Language Model



Masked Language Model

Permutation Language Model



Objective: Permutation Language Model

- Perform autoregressive factorization on T! different orders
  - **Address bidirectional context**
- Objective fits into the AR framework, naturally avoids
  - **Independence assumption**
  - **pretrain-finetune discrepancy**

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

2.2 Objective: Permutation Language Modeling

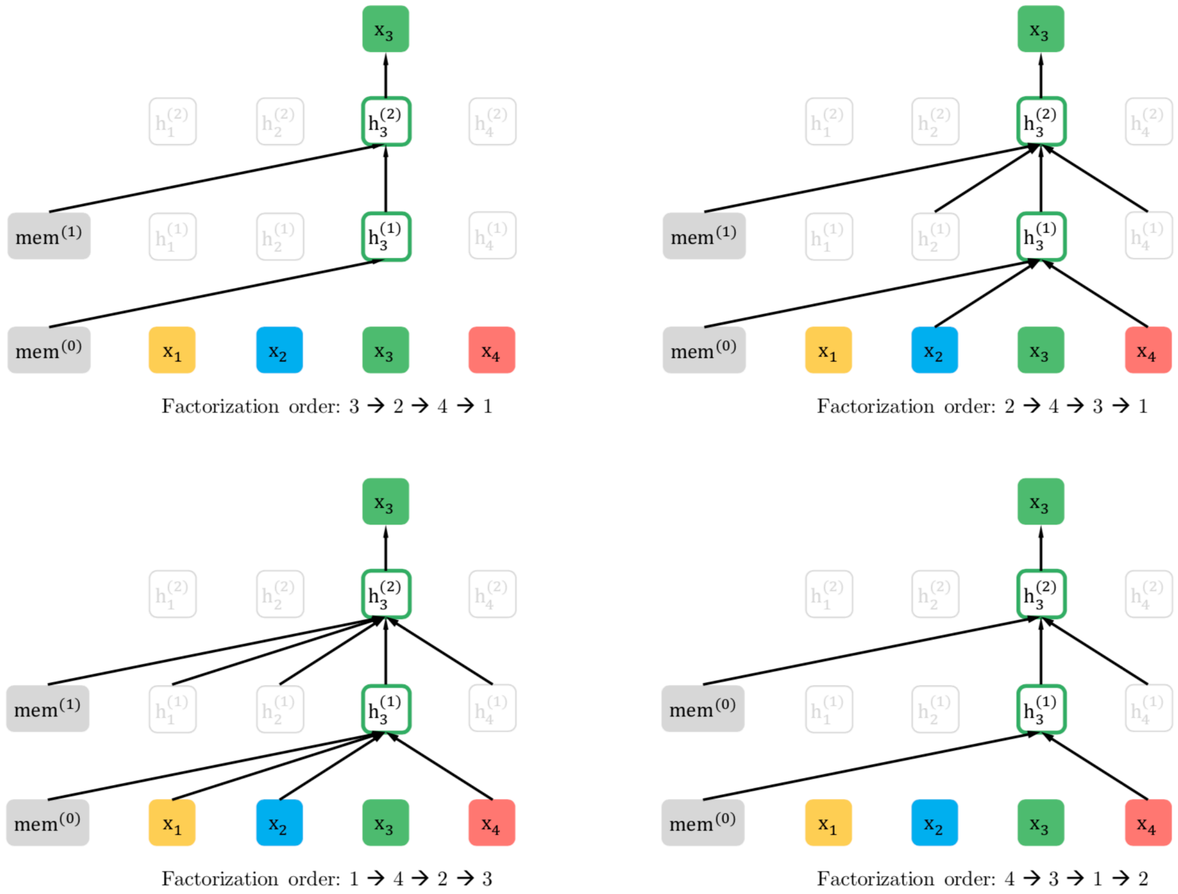


Figure 1: Illustration of the permutation language modeling objective for predicting x_3 given the same input sequence $\mathbf{x}$ but with different factorization orders.

Problem with using standard transformer

Permutation Language Model

$$p_\theta(X_{z_t} = x \mid \mathbf{x}_{z_{<t}}) = \frac{\exp\left(e(x)^\top h_\theta(\mathbf{x}_{\mathbf{z}<t})\right)}{\sum_{x'} \exp\left(e(x')^\top h_\theta(\mathbf{x}_{\mathbf{z}<t})\right)}$$

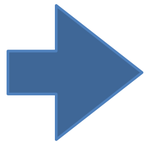
- The representation h_θ does not depend on which position it will predict
  - Consequently, the **same distribution is predicted** regardless of the target position

[New, York, is, a, city]

[is, a, city, New, York]

⋮

[is, a, city, York, New]



log *p* (NEW | is a city) = log *p* (York | is a city)

Two-Stream Self Attention

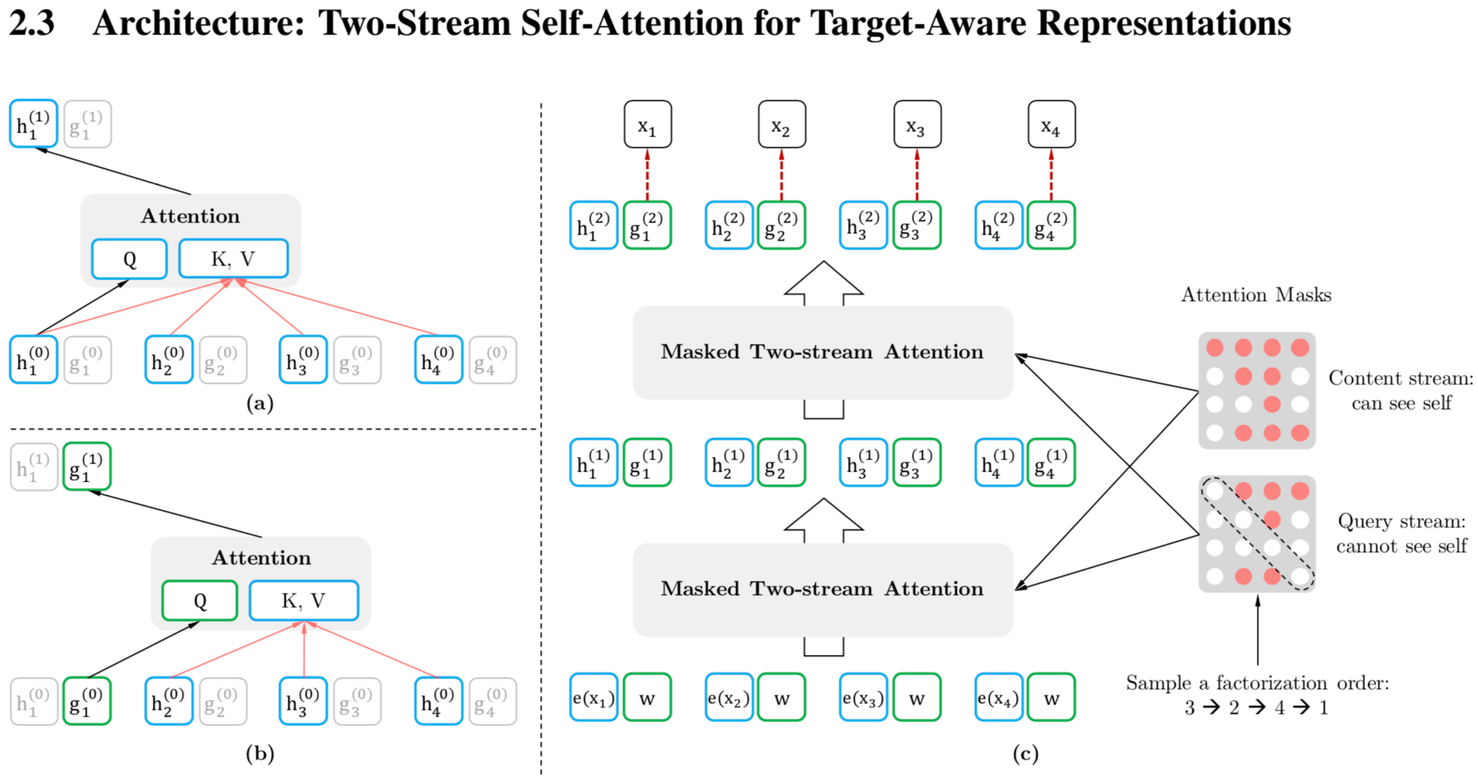
- Re-parameterize the next-token distribution to be **target position aware (Two-stream Self Attention)**

$$\frac{\exp\left(e(x)^\top h_\theta(\mathbf{x}_{\mathbf{z}<t})\right)}{\sum_{x'}\exp\left(e(x')^\top h_\theta(\mathbf{x}_{\mathbf{z}<t})\right)} \quad \Rightarrow \quad \frac{\exp\left(e(x)^\top g_\theta(\mathbf{x}_{\mathbf{z}<t}, z_t)\right)}{\sum_{x'}\exp\left(e(x')^\top g_\theta(\mathbf{x}_{\mathbf{z}<t}, z_t)\right)},$$

- Two-stream Self Attention
  - (1) To predict the token x_{z_t}, $g_\theta(x_{z<t}, z_t)$ **should only use the *position*** z_t and not the content x_t, otherwise the objective becomes trivial
  - (2) To predict the other tokens x_{z_t} with $j > t$, $g_\theta(x_{z<t}, z_t)$ **should also encode the content** x_t to provide full contextual information

- The content representation $h_\theta(\mathbf{x}_{\mathbf{z}\leq t})$, or abbreviated as h_{z_t}, which serves a similar role to the standard hidden states in Transformer. This representation encodes *both* the context and x_{z_t} itself.
- The query representation $g_\theta(\mathbf{x}_{\mathbf{z}<t}, z_t)$, or abbreviated as g_{z_t}, which only has access to the contextual information $\mathbf{x}_{\mathbf{z}<t}$ and the position z_t, but not the content x_{z_t}, as discussed above.

Two-Stream Self Attention



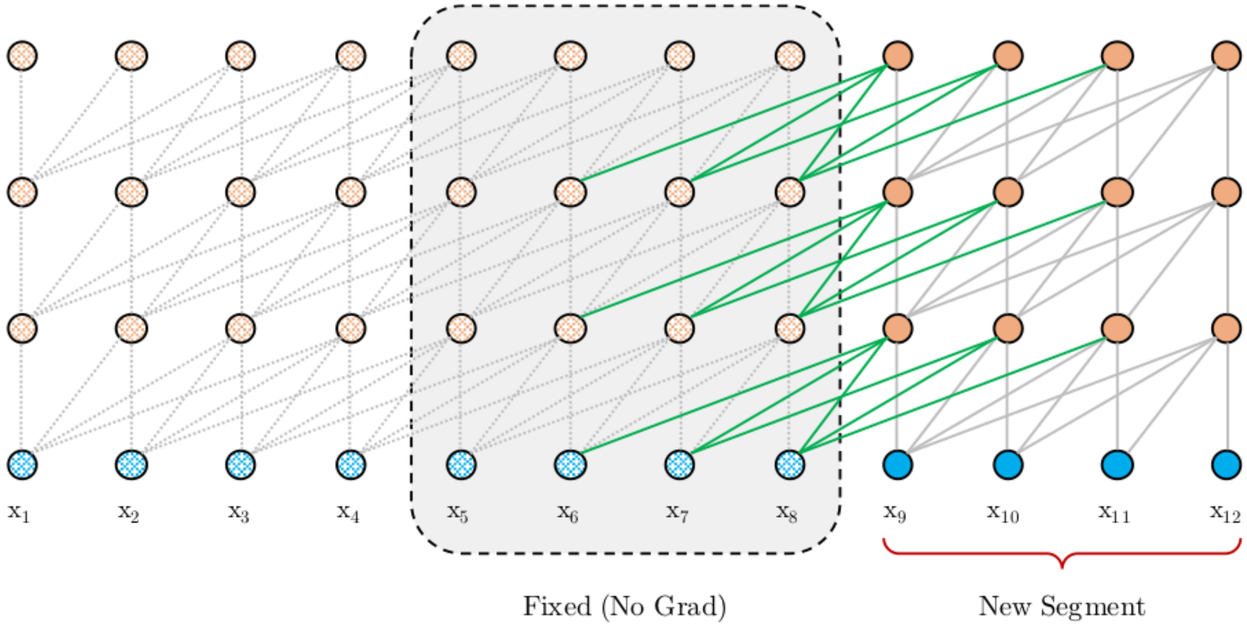
- First layer of query stream is initialized with a trainable vector $g_i^{(0)} = w$
- First layer of content stream is $h_i^0 = e(x_i)$
- For each self-attention layer, the two streams of representations are updated as follows

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

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

Transformer-XL

- Segment-Level Recurrence

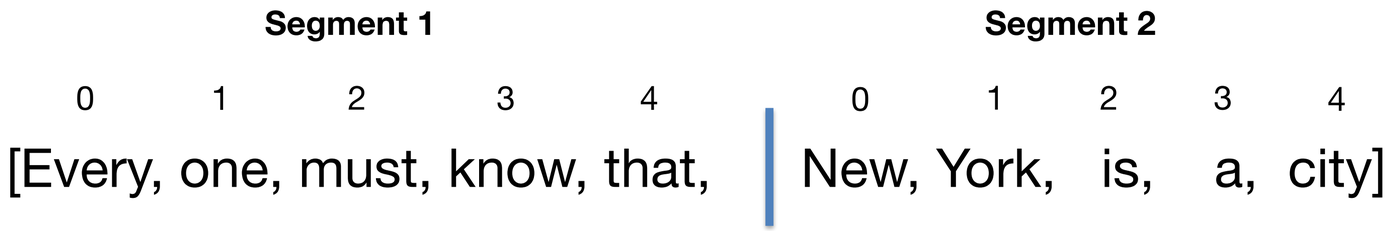


$$\widetilde{\mathbf{h}}_{\tau+1}^{n-1} = [\mathrm{SG}(\mathbf{h}_{\tau}^{n-1}) \circ \mathbf{h}_{\tau+1}^{n-1}],$$
$$\mathbf{q}_{\tau+1}^{n}, \mathbf{k}_{\tau+1}^{n}, \mathbf{v}_{\tau+1}^{n} = \mathbf{h}_{\tau+1}^{n-1}\mathbf{W}_{q}^{\top}, \widetilde{\mathbf{h}}_{\tau+1}^{n-1}\mathbf{W}_{k}^{\top}, \widetilde{\mathbf{h}}_{\tau+1}^{n-1}\mathbf{W}_{v}^{\top},$$
$$\mathbf{h}_{\tau+1}^{n} = \text{Transformer-Layer}\left(\mathbf{q}_{\tau+1}^{n}, \mathbf{k}_{\tau+1}^{n}, \mathbf{v}_{\tau+1}^{n}\right)$$

Segment 1 | **Segment 2**

[Every, one, must, know, that, | New, York, is, a, city]

Relative positional encoding



Use **relative distance** between query vector, q_i and key vector, k_j , i.e. $(i - j)$

Standard Transformer

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

Transfomer-XL

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

VUNO

Pretraining and Implementation

- We use 126GB of plain text in total from BooksCorpus, English Wikipedia, ClueWeb 2012-B, Common Crawl, and Giga5
- XLNet-Large has the same architecture hyper parameters as BERT-Large which results in same model size
- Sequence length: 512
- XLNet-Large was trained on 512 TPU v3 chips for 500K steps with an Adam optimizer
- Linear Learning rate decay and batch size of 2048
- Fine-Tuning procedure follows BERT

Experiments Result

RACE Data set - Question & Answering Task

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.

SQuAD - Reading comprehension Task

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.

Glue Data set - 9 NLP tasks

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.

Text Classification Task

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).

vuno

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

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.

- Three main aspects to study
  - **The effectiveness of permutation LM objective, compared to denoising auto-encoding objective (used by BERT)**
  - The importance of using Transformer-XL
  - The necessity of some implementation details including span-based prediction, the bidirectional input pipeline, and next sentence prediction
- XLNet-Base outperform both BERT and DAE trained on Transformer-XL
- Transformer-XL achieves better performance than BERT

Conclusion

- XLNet is generalized AR pertaining method that uses a permutation language modeling objective to combine the advantages of AR and AE methods
- XLNet achieves SOTA various tasks with substation improvement
- In the future, we envision applications of XLNet to a wider set of tasks such as vision and reinforcement learning

Putting the world’s medical data to work

hello@vuno.co