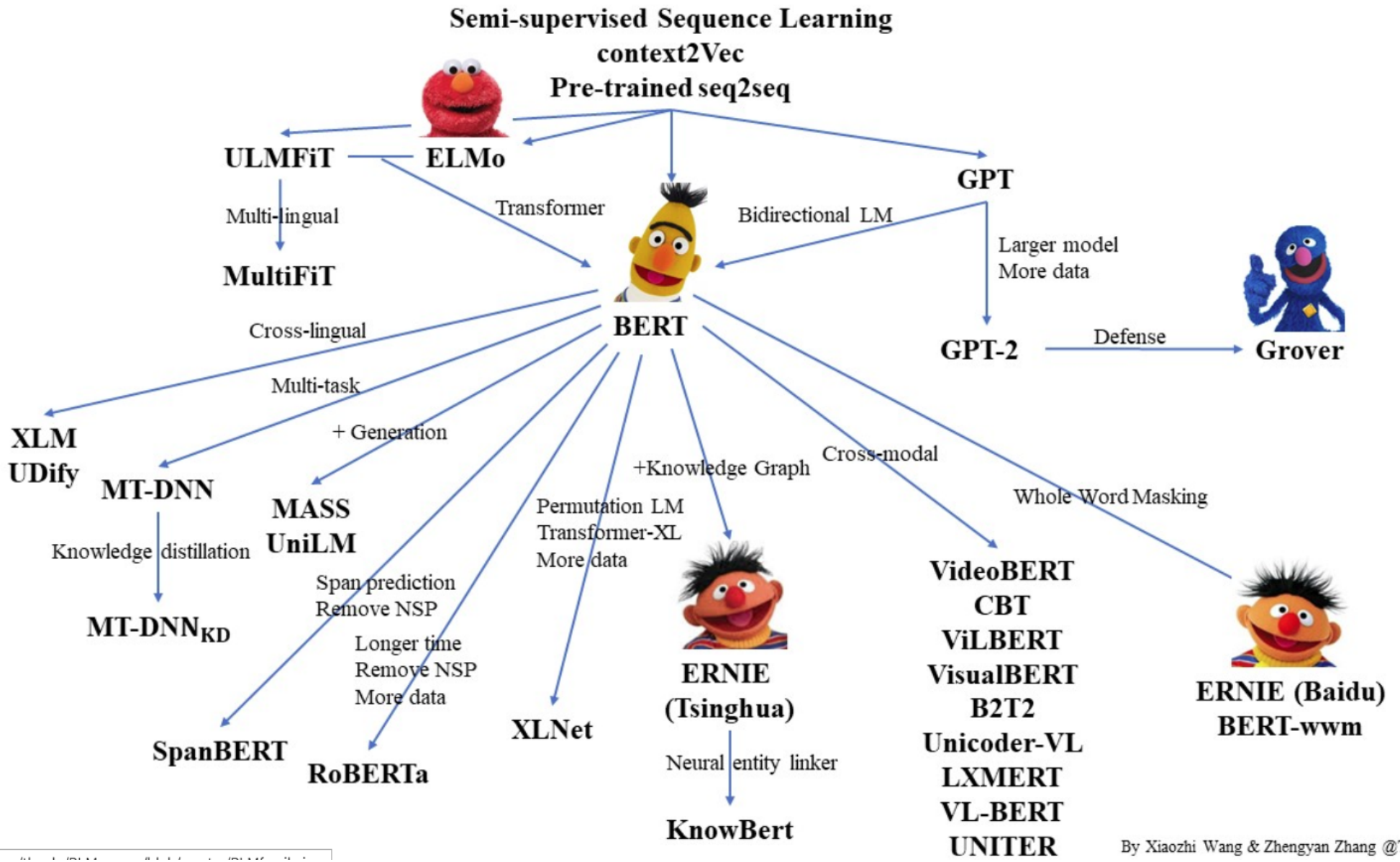
# *Improved Masked Language Models*

Nikolay Arefyev
*CMC MSU Department of Algorithmic Languages &*
*Samsung Moscow Research Center*


Semi-supervised Sequence Learning
context2Vec
Pre-trained seq2seq
ULMFiT
ELMo
GPT
Multi-lingual
MultiFiT
Transformer
Bidirectional LM
Larger model
More data
BERT
GPT-2
Defense
Grover
Cross-lingual
XLM
UDify
Multi-task
MT-DNN
Knowledge distillation
MT-DNN$_{KD}$
+ Generation
MASS
UniLM
Span prediction
Remove NSP
SpanBERT
Longer time
Remove NSP
More data
RoBERTa
Permutation LM
Transformer-XL
More data
XLNet
+Knowledge Graph
ERNIE
(Tsinghua)
Neural entity linker
KnowBert
Cross-modal
VideoBERT
CBT
ViLBERT
VisualBERT
B2T2
Unicoder-VL
LXMERT
VL-BERT
UNITER
Whole Word Masking
ERNIE (Baidu)
BERT-wwm
By Xiaozhi Wang & Zhengyan Zhang @THUNLP

# RoBERTa

- RoBERTa: **R**obustly **o**ptimized **BERT a**pproach, 2019
- Was not accepted to ICLR-2020 due to the limited novelty and despite of SOTA results on GLUE, SQuAD, RACE
- Google sub-optimally selected optimization hyperparameters / decisions and trained BERT poorly, so Facebook fixed it
  => much better than BERT, similar or little better results than XLNet (were all that XLNet tricks useful?)

**RoBERTa: A Robustly Optimized BERT Pretraining Approach**

**Yinhan Liu**[*§] **Myle Ott**[*§] **Naman Goyal**[*§] **Jingfei Du**[*§] **Mandar Joshi**[†]
**Danqi Chen**[§] **Omer Levy**[§] **Mike Lewis**[§] **Luke Zettlemoyer**[†§] **Veselin Stoyanov**[§]

[†] Paul G. Allen School of Computer Science & Engineering,
University of Washington, Seattle, WA
{mandar90,lsz}@cs.washington.edu

[§] Facebook AI
{yinhanliu,myleott,naman,jingfeidu,
danqi,omerlevy,mikelewis,lsz,ves}@fb.com

# Dynamic Masking

- BERT: data was duplicated 10 times and perturbed (masked, etc.) before training for 40 epochs

  => inputs/targets are seen 4 times

- RoBERTa: randomly perturbs inputs during training.
  - No need to duplicate large datasets
  - More diverse data when training more epochs
  - Natural (like dropout)
  - Similar or better results

| Masking | SQuAD 2.0 | MNLI-m | SST-2 |
|---|---|---|---|
| reference | 76.3 | 84.3 | 92.8 |
| *Our reimplementation:* | | | |
| static | 78.3 | 84.3 | 92.5 |
| dynamic | 78.7 | 84.0 | 92.9 |

Table 1: Comparison between static and dynamic masking for $\text{BERT}_{\text{BASE}}$. We report F1 for SQuAD and accuracy for MNLI-m and SST-2. Reported results are medians over 5 random initializations (seeds). Reference results are from Yang et al. (2019).

Figures from Liu et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach, 2019

# Inputs and NSP

- **segment-pair+NSP (BERT)**: a pair of text fragments, combined length is <=512
- **sentence-pair+NSP**: a pair of sentences, much shorter than 512 => dynamic batch size to obtain same number of tokens in a batch
- **full-sentences**: a sequence of whole sentences taken continuously from the corpus while length is <=512. Insert special symbol for document boundaries.
- **doc-sentences**: similarly to previous, but examples do not cross document boundaries. Increase batch size dynamically to obtain same number of tokens in a batch as in full-sentences.

Figures from Liu et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach, 2019

# Inputs and NSP

- Doc-sentences is best. But results in variable size batch.

  => Full-sentences is used for all other experiments.

Sentences are worse than text fragments (cannot learn long-range deps?)

Removing NSP helps (contrary to BERT paper)

| Model | SQuAD 1.1/2.0 | MNLI-m | SST-2 | RACE |
|---|---|---|---|---|
| *Our reimplementation (with NSP loss):* | | | | |
| SEGMENT-PAIR | 90.4/78.7 | 84.0 | 92.9 | 64.2 |
| SENTENCE-PAIR | 88.7/76.2 | 82.9 | 92.1 | 63.0 |
| *Our reimplementation (without NSP loss):* | | | | |
| FULL-SENTENCES | 90.4/79.1 | 84.7 | 92.5 | 64.8 |
| DOC-SENTENCES | 90.6/79.7 | 84.7 | 92.7 | 65.6 |
| $\text{BERT}_{\text{BASE}}$ | 88.5/76.3 | 84.3 | 92.8 | 64.3 |
| $\text{XLNet}_{\text{BASE}}$ (K = 7) | –/81.3 | 85.8 | 92.7 | 66.1 |
| $\text{XLNet}_{\text{BASE}}$ (K = 6) | –/81.0 | 85.6 | 93.4 | 66.7 |

Table 2: Development set results for base models pretrained over BOOKCORPUS and WIKIPEDIA. All models are trained for 1M steps with a batch size of 256 sequences. We report F1 for SQuAD and accuracy for MNLI-m, SST-2 and RACE. Reported results are medians over five random initializations (seeds). Results for $\text{BERT}_{\text{BASE}}$ and $\text{XLNet}_{\text{BASE}}$ are from Yang et al. (2019).

Figures from Liu et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach, 2019

# Inputs and NSP

Compare different batch sizes for the fixed number of epochs (different number of steps).

=> bsz=2K (8x larger than in BERT) is better even with 8x less training steps; bsz=8K is worse (too few steps?)

| bsz | steps | lr | ppl | MNLI-m | SST-2 |
|---|---|---|---|---|---|
| 256 | 1M | 1e-4 | 3.99 | 84.7 | 92.7 |
| 2K | 125K | 7e-4 | **3.68** | **85.2** | **92.9** |
| 8K | 31K | 1e-3 | 3.77 | 84.6 | 92.8 |

Table 3: Perplexity on held-out training data (*ppl*) and development set accuracy for base models trained over BOOKCORPUS and WIKIPEDIA with varying batch sizes (*bsz*). We tune the learning rate (*lr*) for each setting. Models make the same number of passes over the data (epochs) and have the same computational cost.

Figures from Liu et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach, 2019

# Subwords

- BERT: 30K character level BPE (WordPiece) vocabulary, heuristic preprocessing and tokenization.
- RoBERTa: 50K byte-level BPE, no preprocessing or tokenization (much simpler and more universal, but slightly worse results for some tasks in the prelimiary experiments)
  - Too many unicode characters (~140K), a significant part of which occurs in a diverse corpora, can occupy a large part of the vocabulary leaving too few space for words and large parts of words. The model can degenerate to character level model.
  - Byte-level BPE starts from bytes, not unicode symbols. With 50K vocabulary it can encode any input string without using <unk> token.
  - Still looks at unicode character categories to prevent merging different categories (otherwise, wastes vocabulary: *dog. dog! dog?*)

Figures from Liu et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach, 2019

# Data

BERT → RoBERTa: 16GB → **160GB**

- 16GB from BERT: BookCropus + En Wikipedia
- 76GB CC-News – En part of CommonCrawl News
- 38GB OpenWebText – crawled URLs from Reddit with >=3 upvotes
- 31GB Stories – subset of CommonCrawl matching story-like style of Winograd schemas

Training large models: 4days on 64 TPU chips[13] → 1 day on 1024 V100 GPUs

native advertising in the paper

[13] https://cloudplatform.googleblog.com/2018/06/Cloud-TPU-now-offers-preemptible-pricing-and-global-availability.html

Figures from Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018

# RoBERTa

- BERT → **RoBERTa**:
  - Adam: beta1=0.9, beta2=0.999 → **0.98**, eps=1e-6*

    ? improves stability when training with “large batch sizes” [1] / “larger learning rates, but is not directly related to batch size” [2]
  - To speed up: seqlen 128 for the first 90% steps, 512 for the final 10% steps to train all positional embeddings → **512 only**
  - 1M → **500K** steps, bs=256 → **8K,** 32K/128K → **4M** topkens/batch
  - lr warmup 10K → **30K**, linear lr decay
  -

* “While most deep learning platforms use a default value of eps=1e-8, we found that BERT would often fail to train with eps=1e-8 or 1e-7, especially for the larger models. An epsilon of 1e-6 was the smallest value that worked consistently” [2]


- [1] Liu et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach, 2019
- [2]https://openreview.net/forum?id=SyxS0T4tvS

# Dev set results

- "We note that even our longest-trained model does not appear to overfit our data and would likely benefit from additional training." (money finished)

Same arch, similar data: large improvement

More data helps only when trained longer

| Model | data | bsz | steps | SQuAD (v1.1/2.0) | MNLI-m | SST-2 |
|---|---|---|---|---|---|---|
| RoBERTa | | | | | | |
| with BOOKS + WIKI | 16GB | 8K | 100K | 93.6/87.3 | 89.0 | 95.3 |
| + additional data (§3.2) | 160GB | 8K | 100K | 94.0/87.7 +0.3 | 89.3 +0.3 | 95.6 +0.3 |
| + pretrain longer | 160GB | 8K | 300K | 94.4/88.7 | 90.0 | 96.1 |
| + pretrain even longer | 160GB | 8K | 500K | **94.6/89.4** +2.1 | **90.2** +1.2 | **96.4** +1.1 |
| $\text{BERT}_{\text{LARGE}}$ | | | | | | |
| with BOOKS + WIKI | 13GB | 256 | 1M | 90.9/81.8 | 86.6 | 93.7 |
| $\text{XLNet}_{\text{LARGE}}$ | | | | | | |
| with BOOKS + WIKI | 13GB | 256 | 1M | 94.0/87.8 | 88.4 | 94.4 |
| + additional data | 126GB | 2K | 500K | 94.5/88.8 | 89.8 | 95.6 |

SQUAD v2.0: Clf head arch. differs from BERT!

Table 4: Development set results for RoBERTa as we pretrain over more data (16GB → 160GB of text) and pretrain for longer (100K → 300K → 500K steps). Each row accumulates improvements from the rows above. RoBERTa matches the architecture and training objective of $\text{BERT}_{\text{LARGE}}$. Results for $\text{BERT}_{\text{LARGE}}$ and $\text{XLNet}_{\text{LARGE}}$ are from Devlin et al. (2019) and Yang et al. (2019), respectively. Complete results on all GLUE tasks can be found in the Appendix.

Figures from Liu et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach, 2019

# Dev set results

- Hyperparameter selection for GLUE: BERT → **RoBERTa**
  - Bs: 32 → **16, 32**
  - Max learning rate (2,3,4,5)e-5 → (**1,2,3)e-5**, linear warup 6% + linear decay
  - Epochs: 3 → **10**, early stopping on dev set

| | MNLI | QNLI | QQP | RTE | SST | MRPC | CoLA | STS |
|---|---|---|---|---|---|---|---|---|
| RoBERTa$_{\text{BASE}}$ | | | | | | | | |
| + all data + 500k steps | 87.6 | 92.8 | 91.9 | 78.7 | 94.8 | 90.2 | 63.6 | 91.2 |
| RoBERTa$_{\text{LARGE}}$ | | | | | | | | |
| with BOOKS + WIKI | 89.0 | 93.9 | 91.9 | 84.5 | 95.3 | 90.2 | 66.3 | 91.6 |
| + additional data (§3.2) | 89.3 | 94.0 | 92.0 | 82.7 | 95.6 | **91.4** | 66.1 | 92.2 |
| + pretrain longer 300k | 90.0 | 94.5 | **92.2** | 83.3 | 96.1 | 91.1 | 67.4 | 92.3 |
| + pretrain longer 500k | **90.2** | **94.7** | **92.2** | **86.6** | **96.4** | 90.9 | **68.0** | **92.4** |

Table 8: Development set results on GLUE tasks for various configurations of RoBERTa.

Figures from Liu et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach, 2019

# GLUE test results

- Single-task finetuning (unlike some other top models); for RTE, STS, MRPC start fine-tuning from model fine-tuned on MNLI
- Ensemble 5-7 models per task

| | MNLI | QNLI | QQP | RTE | SST | MRPC | CoLA | STS | WNLI | Avg |
|---|---|---|---|---|---|---|---|---|---|---|
| *Single-task single models on dev* | | | | | | | | | | |
| $\text{BERT}_{\text{LARGE}}$ | 86.6/- | 92.3 | 91.3 | 70.4 | 93.2 | 88.0 | 60.6 | 90.0 | - | - |
| $\text{XLNet}_{\text{LARGE}}$ | 89.8/- | 93.9 | 91.8 | 83.8 | 95.6 | 89.2 | 63.6 | 91.8 | - | - |
| RoBERTa | **90.2/90.2** | **94.7** | **92.2** | **86.6** | **96.4** | **90.9** | **68.0** | **92.4** | **91.3** | - |
| *Ensembles on test (from leaderboard as of July 25, 2019)* | | | | | | | | | | |
| ALICE | 88.2/87.9 | 95.7 | **90.7** | 83.5 | 95.2 | 92.6 | **68.6** | 91.1 | 80.8 | 86.3 |
| MT-DNN | 87.9/87.4 | 96.0 | 89.9 | 86.3 | 96.5 | 92.7 | 68.4 | 91.1 | 89.0 | 87.6 |
| XLNet | 90.2/89.8 | 98.6 | 90.3 | 86.3 | **96.8** | **93.0** | 67.8 | 91.6 | **90.4** | 88.4 |
| RoBERTa | **90.8/90.2** | **98.9** | 90.2 | **88.2** | 96.7 | 92.3 | 67.8 | **92.2** | 89.0 | **88.5** |

Table 5: Results on GLUE. All results are based on a 24-layer architecture. $\text{BERT}_{\text{LARGE}}$ and $\text{XLNet}_{\text{LARGE}}$ results are from Devlin et al. (2019) and Yang et al. (2019), respectively. RoBERTa results on the development set are a median over five runs. RoBERTa results on the test set are ensembles of *single-task* models. For RTE, STS and MRPC we finetune starting from the MNLI model instead of the baseline pretrained model. Averages are obtained from the GLUE leaderboard.

Figures from Liu et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach, 2019

# Other datasets

- SOTA among models w/o data augmentation

| Model | SQuAD 1.1 EM | SQuAD 1.1 F1 | SQuAD 2.0 EM | SQuAD 2.0 F1 |
|---|---|---|---|---|
| *Single models on dev, w/o data augmentation* | | | | |
| $\text{BERT}_{\text{LARGE}}$ | 84.1 | 90.9 | 79.0 | 81.8 |
| $\text{XLNet}_{\text{LARGE}}$ | **89.0** | 94.5 | 86.1 | 88.8 |
| RoBERTa | 88.9 | **94.6** | **86.5** | **89.4** |
| *Single models on test (as of July 25, 2019)* | | | | |
| $\text{XLNet}_{\text{LARGE}}$ | | | 86.3† | 89.1† |
| RoBERTa | | | 86.8 | 89.8 |
| XLNet + SG-Net Verifier | | | **87.0**† | **89.9**† |

Table 6: Results on SQuAD. † indicates results that depend on additional external training data. RoBERTa uses only the provided SQuAD data in both dev and test settings. $\text{BERT}_{\text{LARGE}}$ and $\text{XLNet}_{\text{LARGE}}$ results are from Devlin et al. (2019) and Yang et al. (2019), respectively.

| Model | Accuracy | Middle | High |
|---|---|---|---|
| *Single models on test (as of July 25, 2019)* | | | |
| $\text{BERT}_{\text{LARGE}}$ | 72.0 | 76.6 | 70.1 |
| $\text{XLNet}_{\text{LARGE}}$ | 81.7 | 85.4 | 80.2 |
| RoBERTa | **83.2** | **86.5** | **81.3** |

Table 7: Results on the RACE test set. $\text{BERT}_{\text{LARGE}}$ and $\text{XLNet}_{\text{LARGE}}$ results are from Yang et al. (2019).

Figures from Liu et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach, 2019

# RoBERTa (summary)

BERT → RoBERTa

- 10x larger dataset
- 32x larger batch size and longer training
  - but 2x less steps
- Dynamic masking
- Removed Next sentence prediction loss and segment-pairs input format
- 30k WordPiece → 50k byte-level BPE vocabulary
- Careful selection of optimization hyperparameters for pre-training and fine-tuning

# XLM-R (XLM-RoBERTa)

- A single RoBERTa model trained on **2.5TB** of **filtered** texts from CommonCrawl on **100 languages**
- Shows zero-shot cross-lingual transfer ability!
- Takes best from XLM (multilingual) and RoBERTa (English) models

**Unsupervised Cross-lingual Representation Learning at Scale**

**Alexis Conneau*** **Kartikay Khandelwal***

**Naman Goyal** **Vishrav Chaudhary** **Guillaume Wenzek** **Francisco Guzmán**

**Edouard Grave** **Myle Ott** **Luke Zettlemoyer** **Veselin Stoyanov**

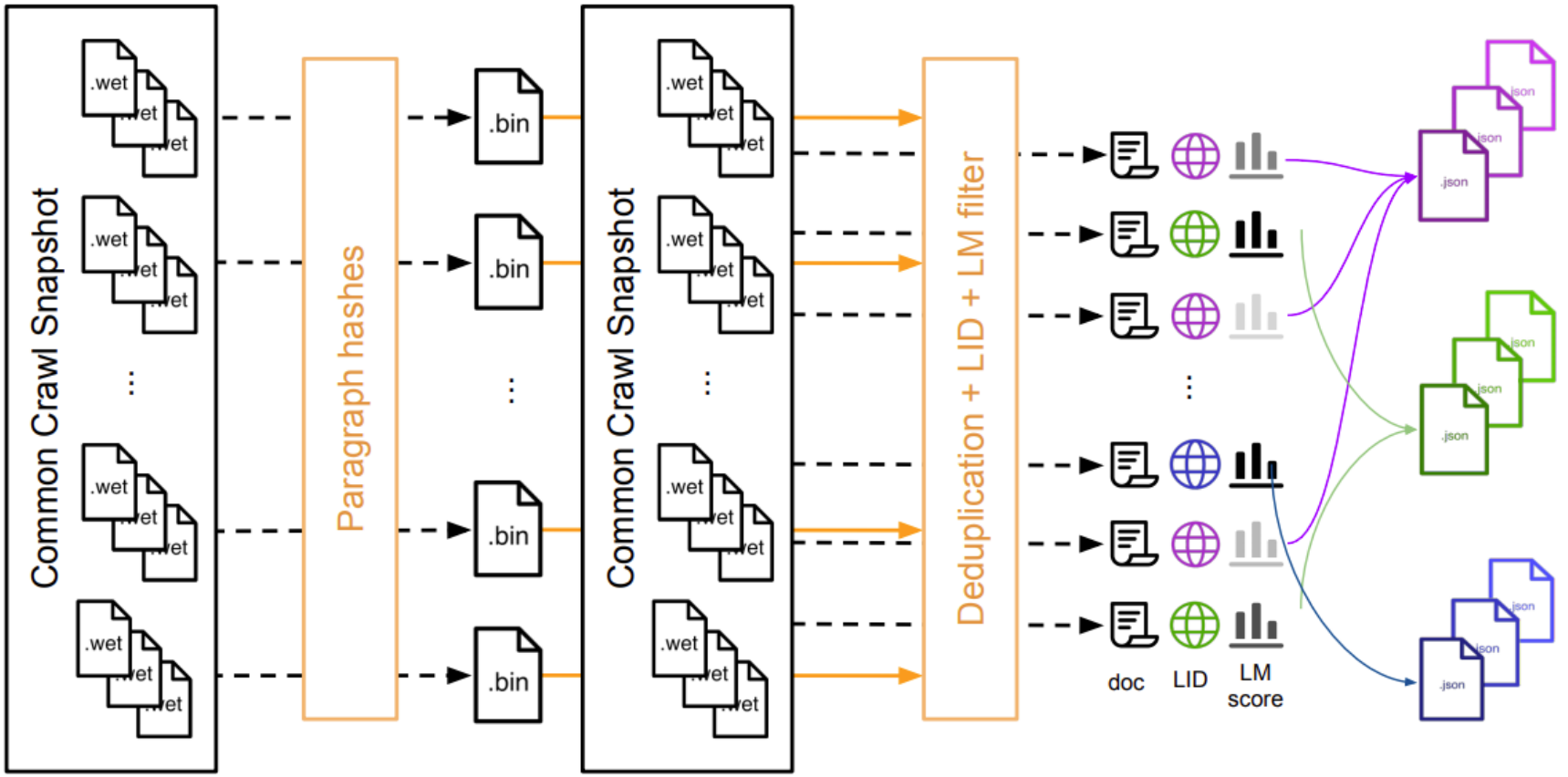
**Facebook AI**

Figures from Conneau et al. Unsupervised Cross-lingual Representation Learning at Scale, 2020

# XLM-R data

Build clean CommonCrawl in 100 langs with CCNet:

- Partially deduplicate paragraphs (also removes boilerplate)
- Run lang id and filter by language
- Filter by perplexity (for each language a Kneser-Ney 5-gram LM is trained on Wikipedia in this language)




Wenzek et al. CCNet: Extracting High Quality Monolingual Datasets from Web Crawl Data, 2019

# XLM-R data

- Upsampling low-resource languages
  - Alleviates bias towards high-resource languages
  - Prevents words in low-resource languages from being split into characters
  - Alpha=0.3 (0.5 in XLM):
    - Ex: English / Uzbek = 300GB / 0.7GB = 430;  $430^{0.3}=6.16$
  - N is number of languages, $n_i$ is number of sentences in language *i*

$$q_i = \frac{p_i^\alpha}{\sum_{j=1}^{N} p_j^\alpha} \quad \text{with} \quad p_i = \frac{n_i}{\sum_{k=1}^{N} n_k}$$

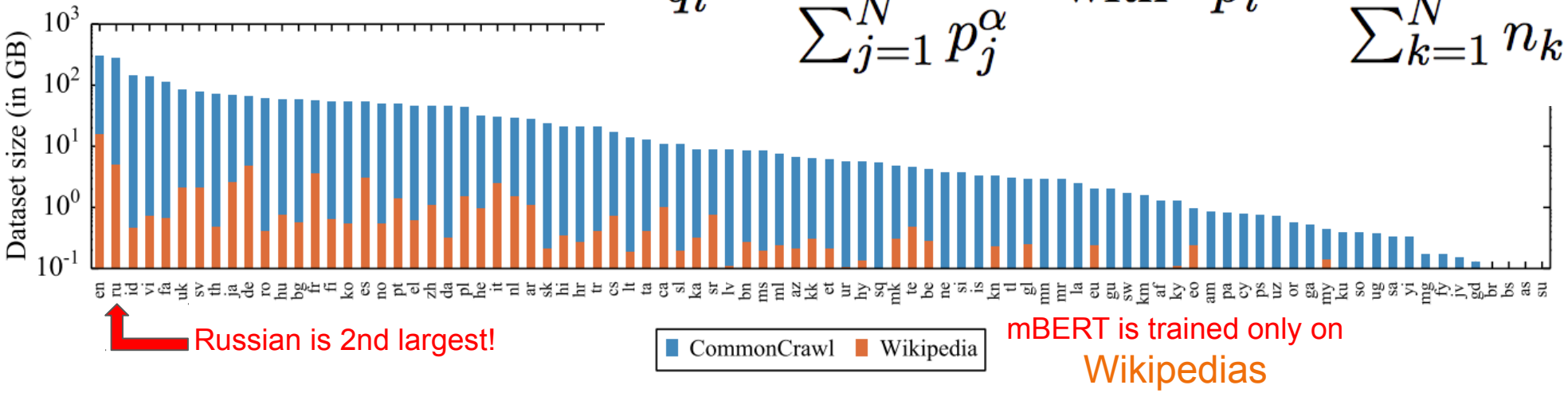



Figure 1: Amount of data in GiB (log-scale) for the 88 languages that appear in both the Wiki-100 corpus used for mBERT and XLM-100, and the CC-100 used for XLM-R. CC-100 increases the amount of data by several orders of magnitude, in particular for low-resource languages.

Figures from Conneau et al. Unsupervised Cross-lingual Representation Learning at Scale, 2020; Lample et al. Cross-lingual Language Model Pretraining, 2019

# The curse of multilinguality

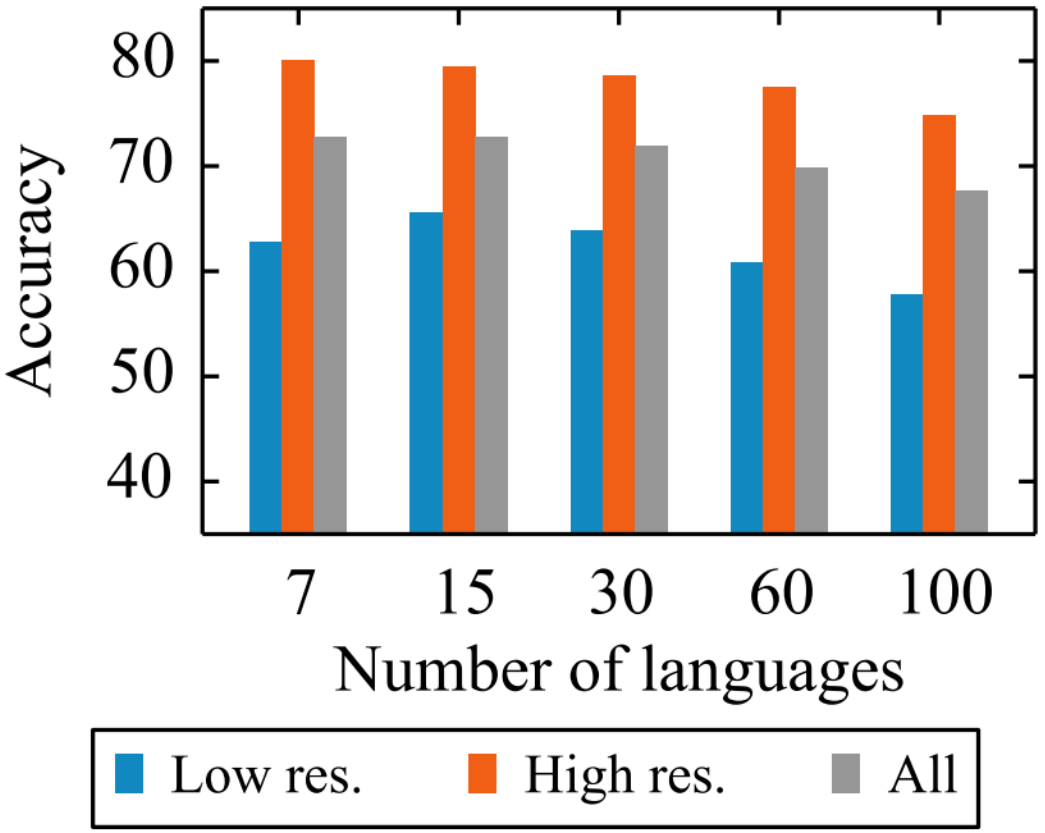
With fixed capacity as we add more langs, the quality:

- When 93 langs added, acc. for 7 eval langs: 0.718 → 0.677
- for high-res. langs decrease (capacity dilution)
- for low-res. – first increases (due to cross-ling. transfer), then decreases (due to capacity dilution)

Increase capacity with number of langs?

For ablations: base model, pre-training on Wikipedia, fine-tuning on XNLI.

Accuracy on XNLI (zero-shot from en?)
High res.: (English + French) / 2
Low res.: (Swahili + Urdu) / 2
All: + German, Russian, Chinese

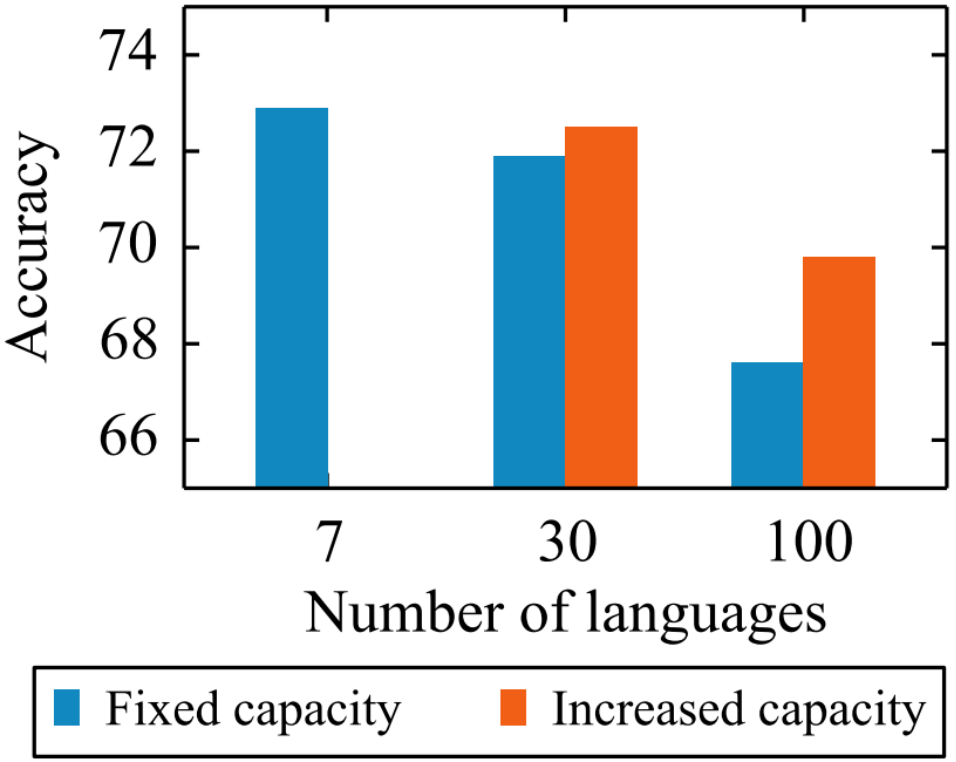



Figures from Conneau et al. Unsupervised Cross-lingual Representation Learning at Scale, 2020

# The curse of multilinguality

Increasing capacity via hidden size: 768/960/1152 for 7/30/100 langs

- 23 langs added to 7 eval. langs almost don't hurt
- 93 langs added to 7 eval. langs still hurt a lot (vocab. is still 150K)

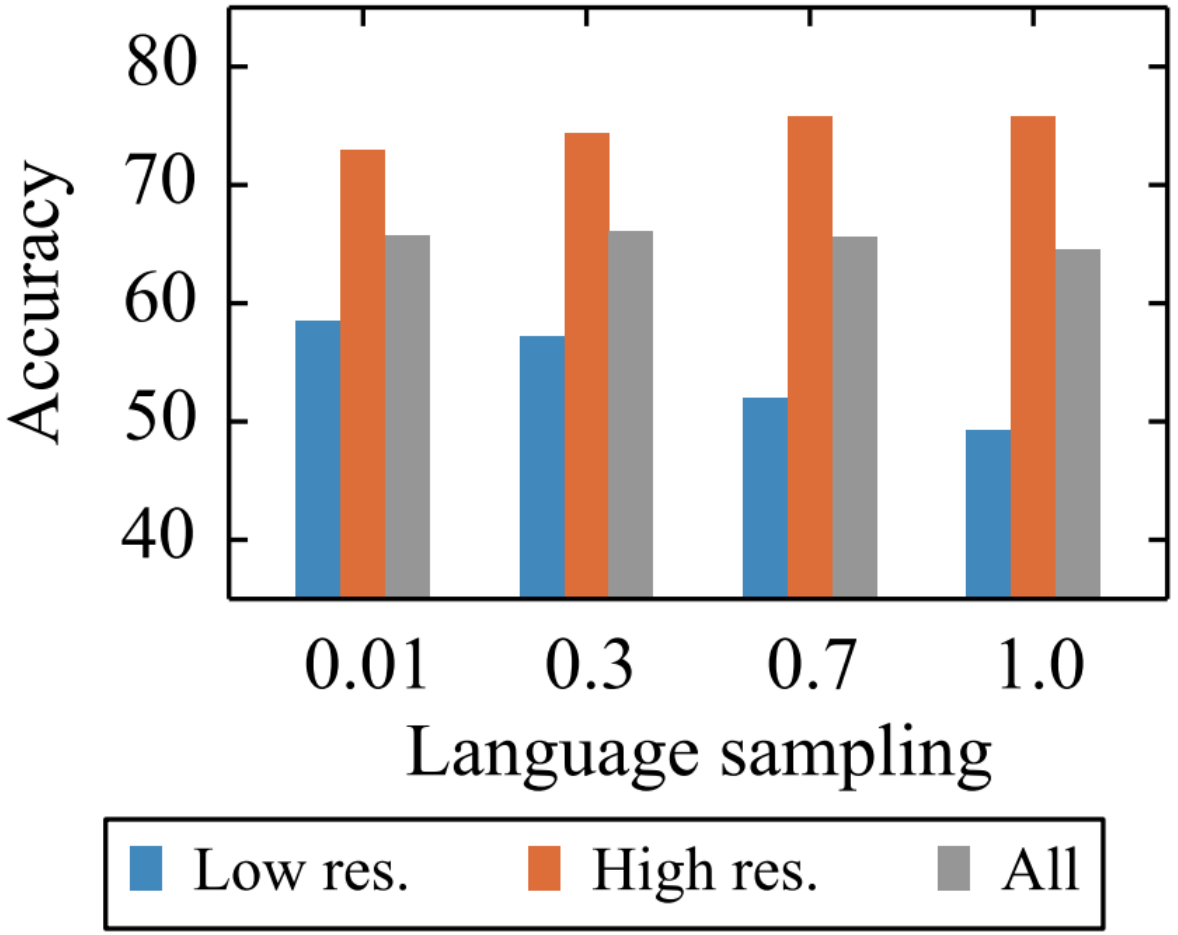



Figures from Conneau et al. Unsupervised Cross-lingual Representation Learning at Scale, 2020

# The curse of multilinguality

XLM-100 on pre-trained Wikipedia

- Upsampling helps low-res. langs a lot, but hurts high-res. langs a little
- alpha=0.3 results in best average accuracy on 7 eval. langs

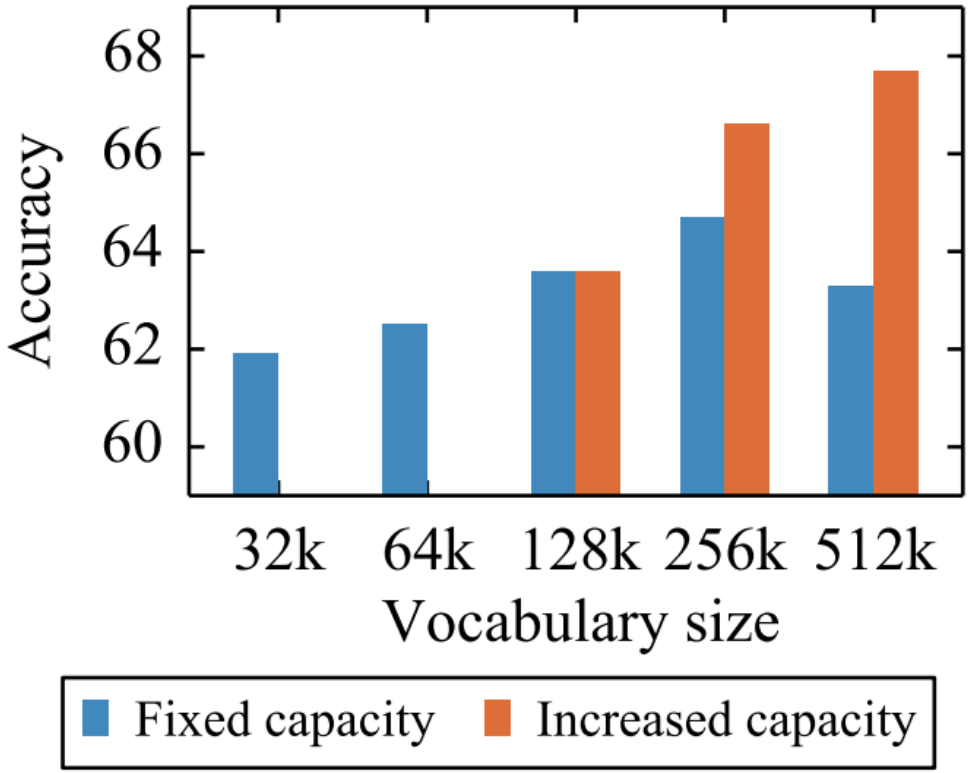



Figures from Conneau et al. Unsupervised Cross-lingual Representation Learning at Scale, 2020

# The curse of multilinguality

XLM-100 on pre-trained Wikipedia

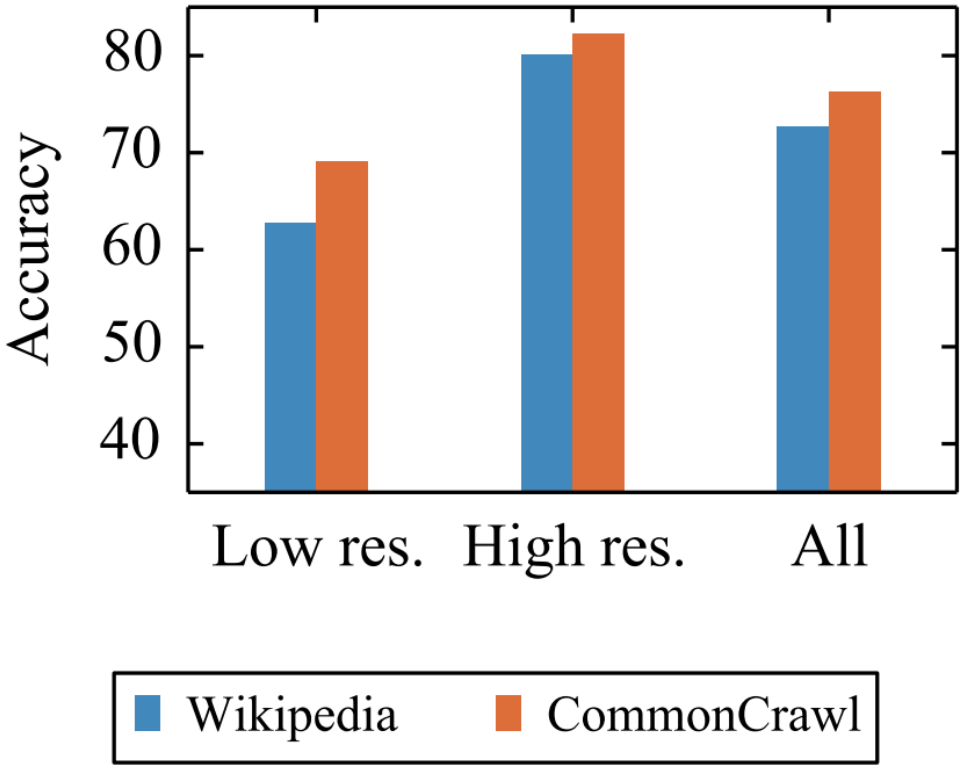
- With fixed capacity, increasing vocab. size (while decreasing hidden size)
  - helps up to certain size 256K, then hidden size becomes too small
    => vocabulary size may be more important than hidden size!
- With fixed hidden size (base model): larger the vocab – better the results
  - +3% when increasing from 128K to 512K
  - further increase of vocab. requires softmax optimization (adaptive softmax?)

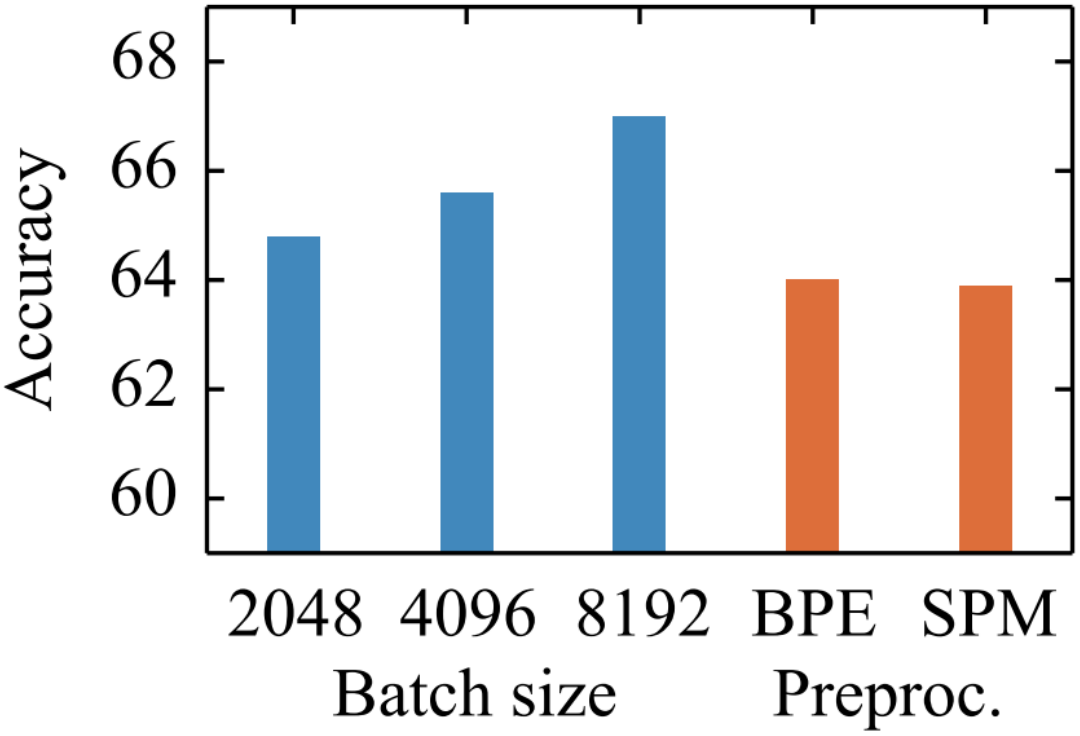



Figures from Conneau et al. Unsupervised Cross-lingual Representation Learning at Scale, 2020

# The curse of multilinguality

- Wikipedia → CommonCrawl helps very much for low-res langs
- Similarly to RoBERTa, larger batches and longer training helps
  - early-stopping of fine-tuning by MLM validation perplexity hurts for final accuracy (XLM underfitted); final accuracy improves even when valid perplexity stops improving!
- SentencePiece is on par with BPE, but words on raw texts without any language-specific pre-processing




Figures from Conneau et al. Unsupervised Cross-lingual Representation Learning at Scale, 2020

# XLM-R large

- As BERT/RoBERTa large: 24 layers of Transformer, hidden size 1024
  - 270M/550M in base/large model
  - about half of parameters are subword embeddings
- Vocabulary: 250K subwords shared between 100 languages
  - Sentence Piece with a unigram LM directly on raw text
  - MLM pre-trained with full softmax
- 1.5M updates, bs=8192
- 500 V100 GPUs
  - How many days?
  - Compared to RoBERTa: 2x less GPUs, 3x more updates, same bs => about 1 week?

Figures from Conneau et al. Unsupervised Cross-lingual Representation Learning at Scale, 2020

# XLM-R Substitution examples

- Can generate reasonable lexical substitutes in **Russian** without any fine-tuning!

**Context**
Угощение на этот раз было куда более демократичное – варёные раки, **вобла** и прочие подобные закуски.
<s>/__У/го/щение/__на/__этот/__раз/__было/__куда/__более/__демократичн/ое/__–/__вар/ё/ные/__рак/и/,/<mask>/__и/__прочие/__подобные/__закуски/./</s>

| **Patterns** | **Substitutes** |
|---|---|
| <mask><br><mask><mask> | сыр, яйца, хлеб, суп, рис, салат, чай, плов<br>картофель, грибы, креветки, пироги, макароны, рыба, мясо, роллы |
| <mask><mask> **и** T<br><mask><mask> **или** T | рыба, креветки, утки, рыбу, мясо, грибы, кабачки, курица<br>рыба, креветки, утки, рыбу, курица, кабачки, мясо, голубцы |
| T **или** <mask><mask> | креветки, рыба, утки, кабачки, курица, рыбу, голубцы, грибы |
| <mask> **или** T | яйца, сыр, рис, рыба, плов, хлеб, молоко, суп |
| <mask><mask><mask> **или** T | морковь, картошка, печенье, мороженое, лосось, морепродукты, курицы, сливки |

Results from Arefyev et al. Word2vec not dead: predicting hypernyms of co-hyponyms is better than reading definitions, 2020

# XLM-R Substitution examples

**Context**
Смелость и решительность башкирской лошади, легкость в управлении, а также способность долго передвигаться резвым **галопом** и рысью позволяли всаднику эффективно вести прицельный огонь и рубить саблей.
*(The courage and determination of the Bashkir horse, the ease of control, and the ability to move for a long time with a quick **gallop** and trot allowed the rider to effectively conduct aimed fire and chop with a saber)*

| **Patterns** | **Substitutes** |
|---|---|
| <mask><br><mask><mask> | путем, видом, трудом, образом, хода, лицом, способом<br>шагом, ходом, конем, движением, ветром, ходам, верхом |
| <mask><mask> **и** T<br><mask><mask> **или** T | шагом, движением, ходом, ходам, ходом, движениям, руслом, темпом<br>шагом, ходом, движением, ходом, ходам, маршем, руслом, темпом |
| T **или** <mask><mask> | верхом, шагом, впереди, пешком, ходом, конем, ходом, бегом |
| <mask> **или** T | путем, хода, трудом, образом, ход, способом, видом, вперед |
| <mask><mask><mask> **или** T | бегом, ездом, ходом, прыжком, шасси, рычанием, ходом, бегством |

Results from Arefyev et al. Word2vec not dead: predicting hypernyms of co-hyponyms is better than reading definitions, 2020

# XNLI accuracy (on 6/15 langs)

4 high-res | 2 high-res | Avg across 15

| Model | D | #M | #lg | en | fr | es | de | sw | ur | Avg |
|---|---|---|---|---|---|---|---|---|---|---|
| *Fine-tune multilingual model on English training set (Cross-lingual Transfer)* | | | | | | | | | | |
| Lample and Conneau (2019) | Wiki+MT | N | 15 | 85.0 | 78.7 | 78.9 | 77.8 | 68.4 | 67.3 | 75.1 |
| Huang et al. (2019) | Wiki+MT | N | 15 | 85.1 | 79.0 | 79.4 | 77.8 | 69.7 | 66.7 | 75.4 |
| Devlin et al. (2018) | Wiki | N | 102 | 82.1 | 73.8 | 74.3 | 71.1 | 50.4 | 58.0 | 66.3 |
| Lample and Conneau (2019) | Wiki | N | 100 | 83.7 | 76.2 | 76.6 | 73.7 | 58.0 | 62.4 | 71.3 |
| Lample and Conneau (2019) | Wiki | 1 | 100 | 83.2 | 76.7 | 77.7 | 74.0 | 58.2 | 62.4 | 70.7 |
| **XLM-R$_{Base}$** | CC | 1 | 100 | 85.8 | 79.7 | 80.7 | 78.7 | 66.5 | 68.3 | 76.2 |
| **XLM-R** | CC | 1 | 100 | **89.1** | **84.1** | **85.1** | **83.9** | **73.9** | **73.8** | **80.9** |
| *Translate everything to English and use English-only model (TRANSLATE-TEST)* | | | | | | | | | | |
| BERT-en | Wiki | 1 | 1 | 88.8 | 81.4 | 82.3 | 80.1 | 65.8 | 65.8 | 76.2 |
| RoBERTa | Wiki+CC | 1 | 1 | **91.3** | 82.9 | 84.3 | 81.2 | 66.7 | 66.8 | 77.8 |
| *Fine-tune multilingual model on each training set (TRANSLATE-TRAIN)* | | | | | | | | | | |
| Lample and Conneau (2019) | Wiki | N | 100 | 82.9 | 77.6 | 77.9 | 77.9 | 66.5 | 62.4 | 74.2 |
| *Fine-tune multilingual model on all training sets (TRANSLATE-TRAIN-ALL)* | | | | | | | | | | |
| Lample and Conneau (2019)† | Wiki+MT | 1 | 15 | 85.0 | 80.8 | 81.3 | 80.3 | 72.8 | 68.5 | 77.8 |
| Huang et al. (2019) | Wiki+MT | 1 | 15 | 85.6 | 81.1 | 82.3 | 80.9 | 73.8 | 69.6 | 78.5 |
| Lample and Conneau (2019) | Wiki | 1 | 100 | 84.5 | 80.1 | 81.3 | 79.3 | 69.2 | 67.7 | 76.9 |
| **XLM-R$_{Base}$** | CC | 1 | 100 | 85.4 | 81.4 | 82.2 | 80.3 | 73.1 | 73.0 | 79.1 |
| **XLM-R** | CC | 1 | 100 | **89.1** | **85.1** | **86.6** | **85.7** | **78.0** | **78.1** | **83.6** |

swap

0-shot from en,

+14% cmp. to mBERT

Best for en, worst for other langs

Best for all langs except en

Figures from Conneau et al. Unsupervised Cross-lingual Representation Learning at Scale, 2020

# XLM-R (Summary)

- Pre-training a single MLM on 100 languages:
  - requires high capacity (large vocab., hidden size)
  - requires long training with low-res. langs upsampling
- Fine-tuning multilingual MLM:
  - enables zero-shot cross-lingual transfer
  - but better translate (MT) into all target languages and fine-tune on multilingual train set

# XLNet

- Combines best from LM and MLM pre-training
- An alternative to MLM: deeply bidirectional model, which is still autoregressive!
- Improves backbone architecture by integrating ideas from Transformer-XL

---

## XLNet: Generalized Autoregressive Pretraining for Language Understanding

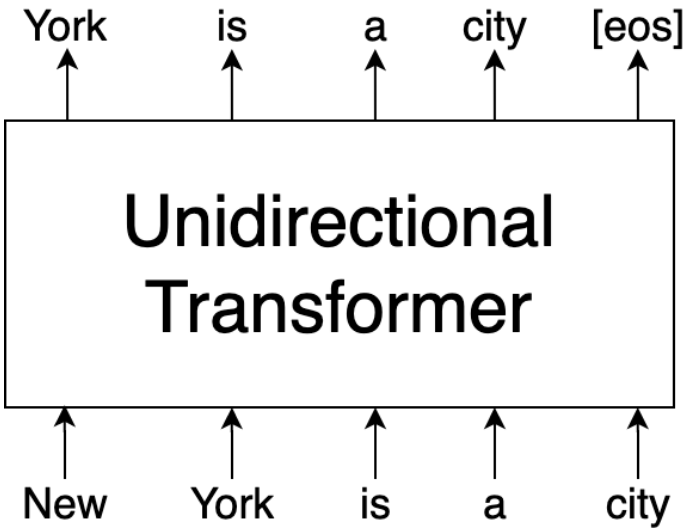
---

**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 AI Brain Team
{zhiliny,dzihang,yiming,jgc,rsalakhu}@cs.cmu.edu, qvl@google.com

# LM vs. MLM pre-training

Language model

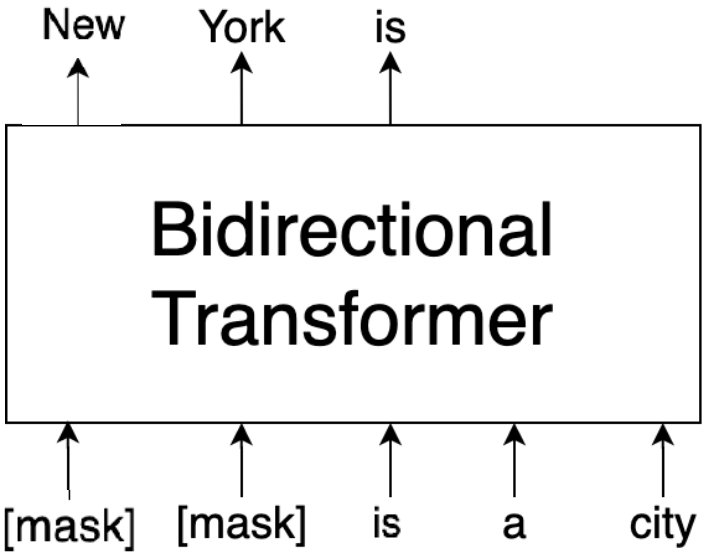
**Auto-Regressive (AR)**




$$\max_{\theta} \sum_{t=1}^{T} \log p_\theta(x_t | \mathbf{x}_{<t})$$

Masked Language model

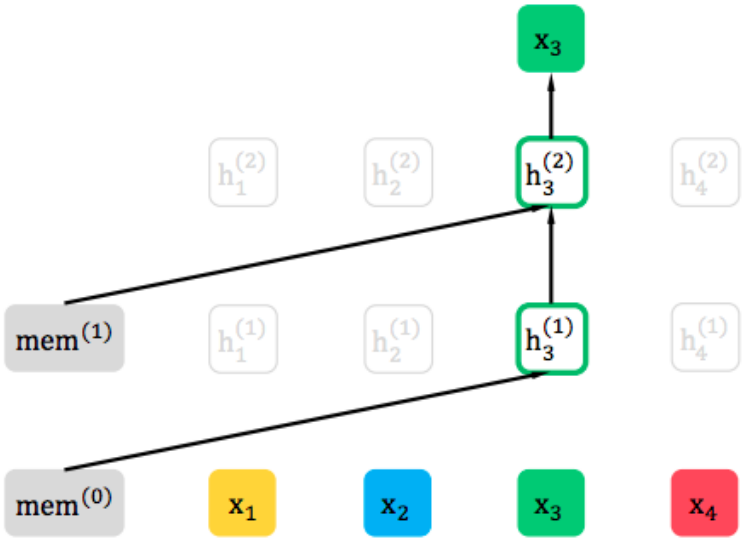
**Denoising Auto-Encoder (AE)**




$$\max_{\theta} \sum_{t=1}^{T} \mathrm{m}_t \log p_\theta(x_t | \tilde{\mathbf{x}})$$

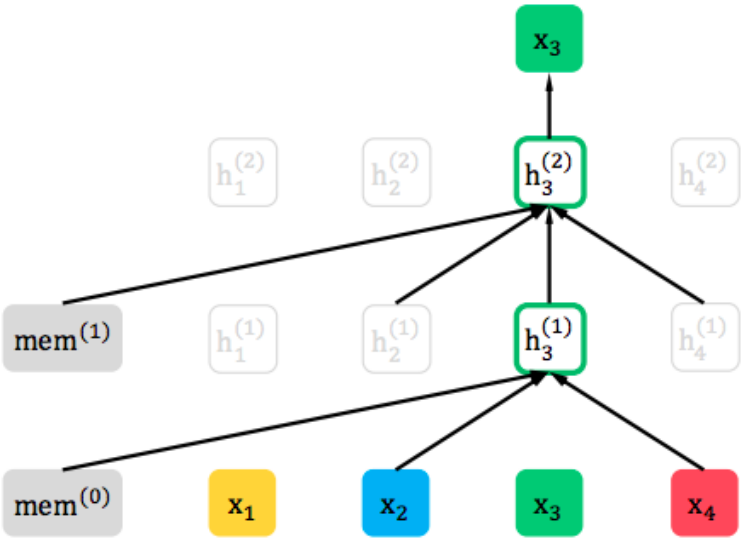
- Natural auto-regressive dependence ⟺ Masked tokens are treated as independent given context (predicted all at once)
- Similar distributions of inputs during pre-training and fine-tuning ⟺ Discrepancy between distributions of inputs during pre-training and fine-tuning ( [mask] token, random replacement)
- No Bidirectional Context ⟺ Natural Bidirectional Context

# Permutation language model



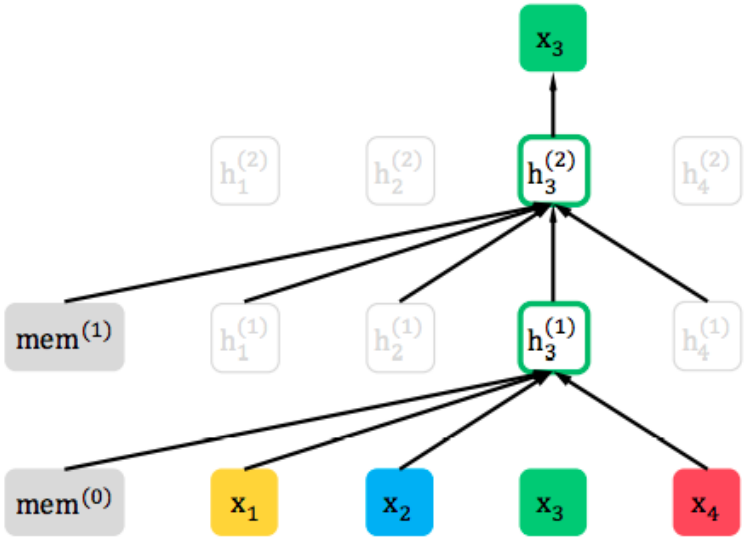

Factorization order: 3 → 2 → 4 → 1

$$P(X)=\boldsymbol{P}(\boldsymbol{x_3})P(x_2|x_3)P(x_4|x_3,x_2)P(x_1|x_3,x_2,x_4)$$



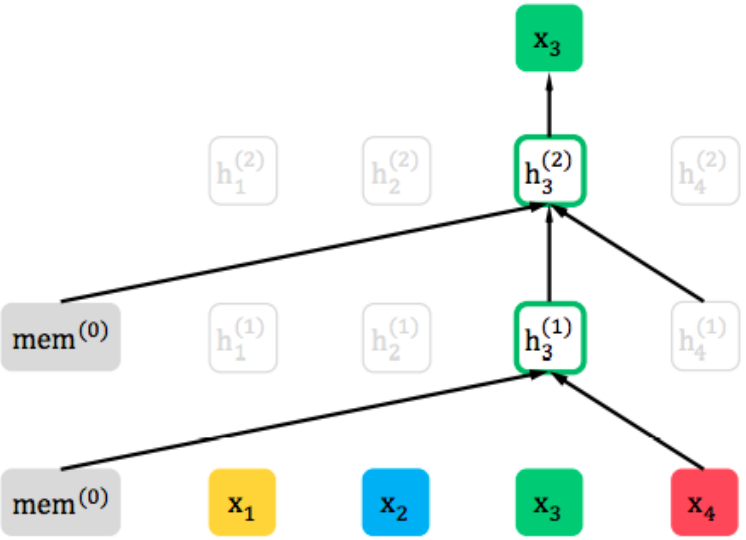

Factorization order: 2 → 4 → 3 → 1

$$P(X)=P(x_2)P(x_4|x_2)\boldsymbol{P}(\boldsymbol{x_3}|\boldsymbol{x_2},\boldsymbol{x_4})P(x_1|x_2,x_4,x_3)$$




Factorization order: 1 → 4 → 2 → 3

$$P(X)=P(x_1)P(x_4|x_1)P(x_2|x_1,x_4)\boldsymbol{P}(\boldsymbol{x_3}|\boldsymbol{x_1},\boldsymbol{x_4},\boldsymbol{x_2})$$




Factorization order: 4 → 3 → 1 → 2

$$P(X)=P(x_4)\boldsymbol{P}(\boldsymbol{x_3}|\boldsymbol{x_4})P(x_1|x_4,x_3)P(x_2|x_4,x_3,x_1)$$

Figures from Yang et al. XLNet: Generalized Autoregressive Pretraining for Language Understanding, 2020

# Permutation language model

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

- Maximize the expected log likelihood of training sequences w.r.t. all possible permutations of factorization order.
- Each position doesn't know in advance which other positions will be available. Thus, each position learns to utilize information from all other positions (bidirectional context).
- In practice, loss is calculated only for the last 14-17% of the positions in a permutation.

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

Figures from Yang et al. XLNet: Generalized Autoregressive Pretraining for Language Understanding, 2020

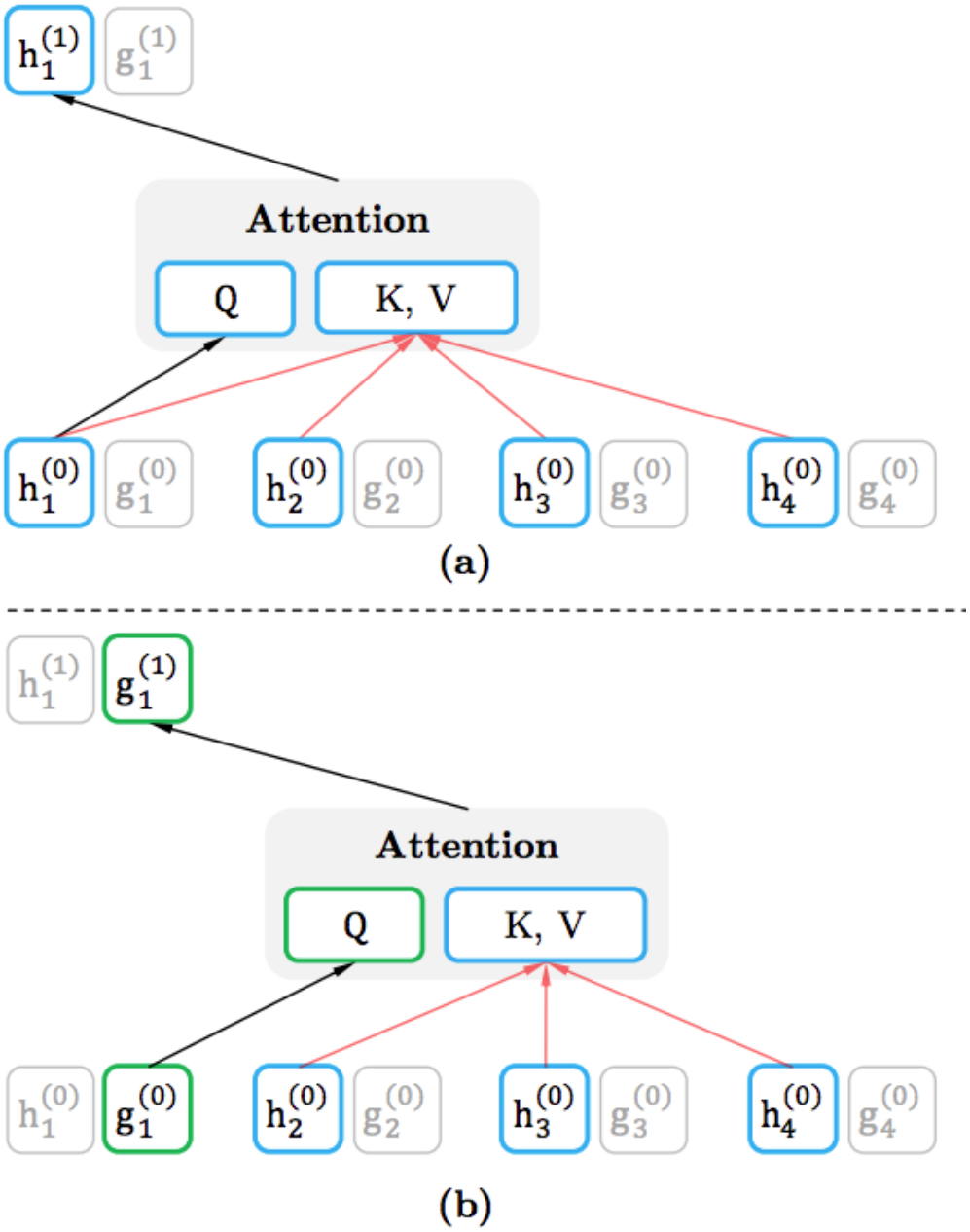
# Two streams

- In a traditional LM we always predict the content of the next position. The model naturally doesn’t see this content.
- In a MLM we predict the content at the current position. The model (usually) doesn’t see this content because it is masked.
- In a permutational LM we shall specify the position for which we want the content to be predicted, but without showing the content at that position!

Figures from Yang et al. XLNet: Generalized Autoregressive Pretraining for Language Understanding, 2020

# Two streams

- Content stream *h* is similar to transformer's decoder
- Query stream *g* doesn't see content at the current position, only positional information




$$g_i^{(0)} = w$$ a trainable vector (independent of $x_i$)

$$h_i^{(0)} = e(x_i)$$

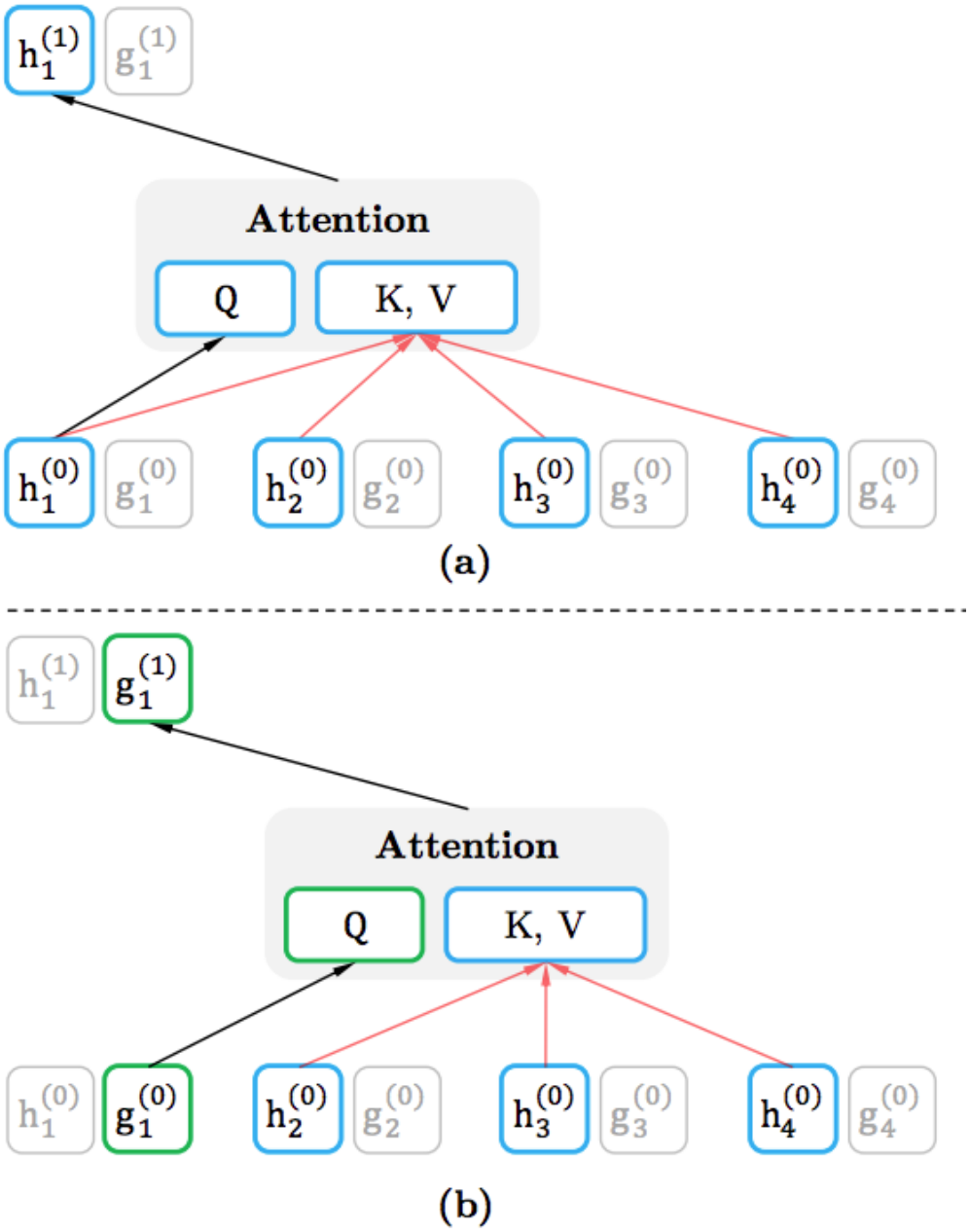
$$g_{z_t}^{(m)} \leftarrow \text{Attention}(\mathrm{Q} = g_{z_t}^{(m-1)}, \mathrm{KV} = \mathbf{h}_{\mathbf{z}_{<t}}^{(m-1)}; \theta)$$

$$h_{z_t}^{(m)} \leftarrow \text{Attention}(\mathrm{Q} = h_{z_t}^{(m-1)}, \mathrm{KV} = \mathbf{h}_{\mathbf{z}_{\leq t}}^{(m-1)}; \theta)$$

Figures from Yang et al. XLNet: Generalized Autoregressive Pretraining for Language Understanding, 2020

# Two streams

- Content stream *h* is similar to transformer's decoder
- Query stream *g* doesn't see content at the current position, only positional information




$$g_i^{(0)} = w$$ a trainable vector (independent of $x_i$)

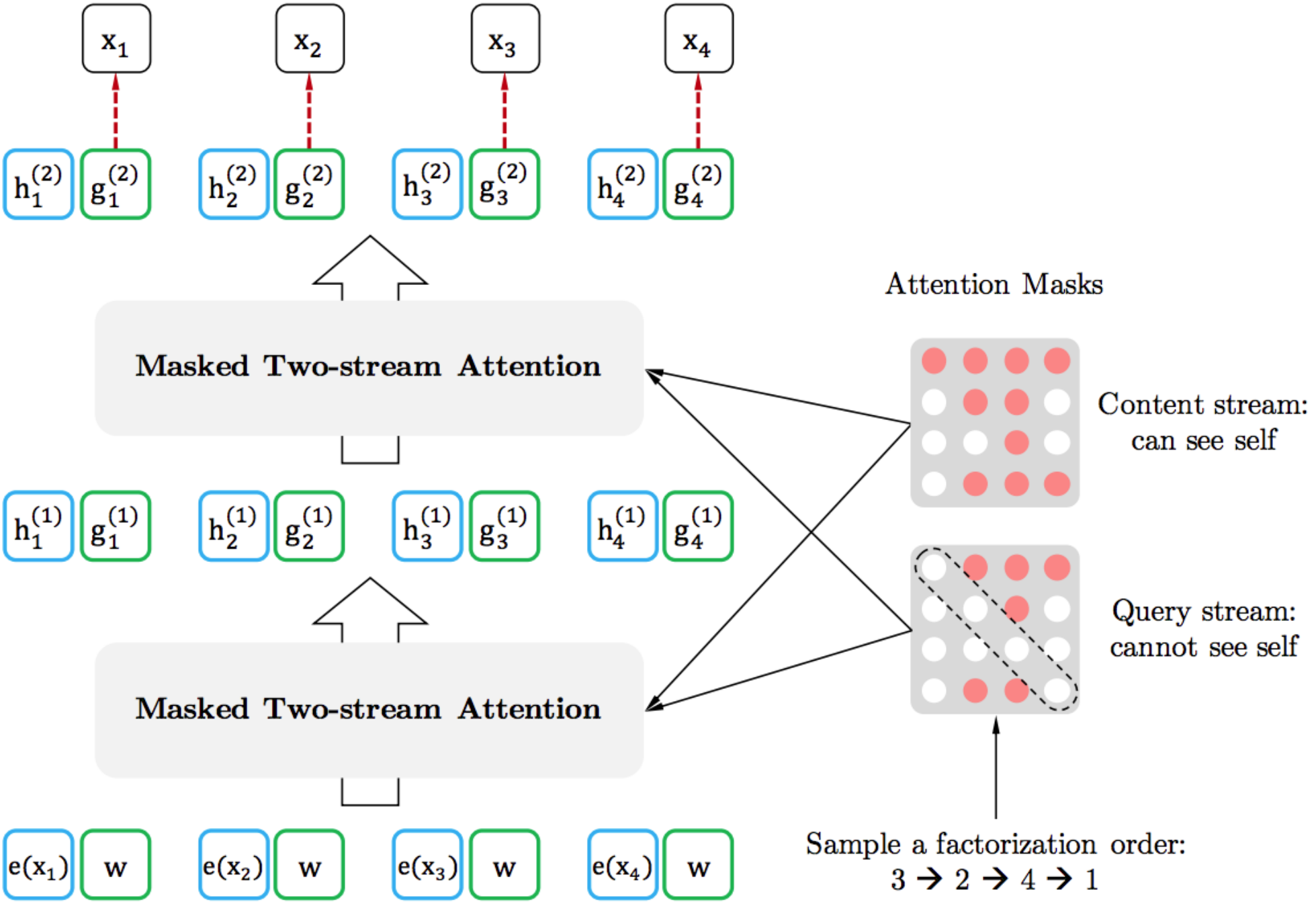
$$h_i^{(0)} = e(x_i)$$

$$g_{z_t}^{(m)} \leftarrow \text{Attention}(\mathrm{Q} = g_{z_t}^{(m-1)}, \mathrm{KV} = \mathbf{h}_{\mathbf{z}_{<t}}^{(m-1)}; \theta)$$

$$h_{z_t}^{(m)} \leftarrow \text{Attention}(\mathrm{Q} = h_{z_t}^{(m-1)}, \mathrm{KV} = \mathbf{h}_{\mathbf{z}_{\leq t}}^{(m-1)}; \theta)$$

Figures from Yang et al. XLNet: Generalized Autoregressive Pretraining for Language Understanding, 2020

# Two streams




Figures from Yang et al. XLNet: Generalized Autoregressive Pretraining for Language Understanding, 2020

# Other details

- Improvements from Transformer-XL
  - Relative positional embeddings
  - Segment recurrence
- Drops query stream when fine-tuning
- Similarly to BERT, segment-pair inputs from same or different documents, but no NSP.
- SentencePiece vocabulary of 32K subwords
- Following BERT, XLNet has 110M / 340M parameters for base / large
- Trained on 126GB of texts
  - bs=2048, 500K steps
  - 512 TPU v3 chips, 2.5 days
  - Underfitted the data, but continuing training didn’t help for downstream tasks.

Figures from Yang et al. XLNet: Generalized Autoregressive Pretraining for Language Understanding, 2020

# XLNet vs. BERT

- Same training data as in BERT: Wikipedia + BooksCorpus
- Same hyperparameters for pretraining as in BERT:
  - Model size: L = 24, H = 1024, A = 16
  - Batch size: 256
  - Number of steps: 1M
  - …