Optimizing Inference Performance of Transformers on CPUs

Dave Dice Alex Kogan

Oracle Labs

Why Transformers?

[cs.CL] 6 Dec 2017

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best

Why Transformers?

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

Abstract

We introduce a new language representation model called **BERT**, which stands for **B**idirectional **E**ncoder **R**epresentations from **T**ransformers. Unlike recent language representation models (Peters et al., 2018a; Radford et al., 2018), BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both

[cs.

There are two existing strategies for applying pre-trained language representations to downstream tasks: *feature-based* and *fine-tuning*. The feature-based approach, such as ELMo (Peters et al., 2018a), uses task-specific architectures that include the pre-trained representations as additional features. The fine-tuning approach, such as the Generative Pre-trained Transformer (OpenAI

You Need

i Parmar* Jakob Uszkoreit*
le Research Google Research
oogle.com usz@google.com

Łukasz Kaiser*
Google Brain
lukaszkaiser@google.com

Illia Polosukhin* ‡
illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best

Why Transformers?

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.

Abstract

We introduce a new language representation model called **BERT**, which stands for **B**idirectional **E**ncoder **R**epresentations from **T**ransformers. Unlike recent language representation models (Peters et al., 2018a; Radford et al., 2018), BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both

[cs.

There are two exis
ing pre-trained langu
stream tasks: *fea*
feature-based
et al., 2018a), us
include the pre-train
tional features. The fine
the Generative Pre-trained

Abstract

Language model pretraining has led to significant performance gains but careful comparison between different approaches is challenging. Training is computationally expensive, often done on private datasets of different sizes, and, as we will show, hyperparameter choices have significant impact on the final re-

Illia Polosukhin
illia.polosukhin@gmail.com

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

We present a replication study of BERT pre-training (Devlin et al., 2019), which includes a careful evaluation of the effects of hyperparmeter tuning and training set size. We find that BERT was significantly undertrained and propose an improved recipe for training BERT models, which we call RoBERTa, that can match or exceed the performance of all of the post-BERT methods.

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best

Why Transformers?

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin **Ming-Wei Chang** **Kenton Lee** **Kristina Toutanova**
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.

There are two exis
ained langu

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

Abstract

Language model pretraining has led to significant performance gains but careful comparison between different approaches is challenging. Training is computationally expensive, often done on private datasets of different sizes, and, as we will show, hyperparameter choices have significant impact on the final re-

We present a replication study of BERT pretraining (Devlin et al., 2019), which includes a careful evaluation of the effects of hyperparmeter tuning and training set size. We find that BERT was significantly undertrained and propose an improved recipe for training BERT models, which we call RoBERTa, that can match or exceed the performance of all of the post-BERT methods.

ALBERT: A LITE BERT FOR SELF-SUPERVISED LEARNING OF LANGUAGE REPRESENTATIONS

Zhenzhong Lan[1] **Mingda Chen**[2*] **Sebastian Goodman**[1] **Kevin Gimpel**[2]
Piyush Sharma[1] **Radu Soricut**[1]
[1]Google Research [2]Toyota Technological Institute at Chicago
{lanzhzh, seabass, piyushsharma, rsoricut}@google.com
{mchen, kgimpel}@ttic.edu

ABSTRACT

Increasing model size when pretraining natural language representations often results in improved performance on downstream tasks. However, at some point further model increases become harder due to GPU/TPU memory limitations and longer training times. To address these problems, we present two parameter-

Abstract

tion models are based on complex recurrent or
at include an encoder and a decoder. The best

Why Transformers?

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin **Ming-Wei Chang** **Kenton Lee** **Kristina Toutanova**
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.

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,
danqi,omerlevy,mikele

Abstract

There are two exis
Language model pret
nificant performance
parison b
ween differe
aining is con
e on private
we will show
nificant impac
lia Polosuk
olosukhi

ALBERT: A LITE BERT FOR SELF-SUPERVISED LEARNING OF LANGUAGE REPRESENTATIONS

Zhenzhong Lan[1] **Mingda Chen**[2*] **Sebastian Goodman**[1] **Kevin Gimpel**[2]
Piyush Sharma[1] **Radu Soricut**[1]
[1]Google Research [2]Toyota Technological Institute at Chicago
{lanzhzh, seabass, piyushsharma, rsoricut}@google.com
{mchen, kgimpel}@ttic.edu

ABSTRACT

Increasing model size when pretraining natural language representations often results in improved performance on downstream tasks. However, at some point further model increases become harder due to GPU/TPU memory limitations and longer training times. To address these problems, we present two parameter-

Abstra
tion mode
at include

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

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

Abstract

Transformers have a potential of learning longer-term dependency, but are limited by a fixed-length context in the setting of language modeling. We propose a novel neural architecture *Transformer-XL* that enables learn-

Term Memory (LSTM) networks (Hochreiter and Schmidhuber, 1997), have been a standard solution to language modeling and obtained strong results on multiple benchmarks. Despite the wide adaption, RNNs are difficult to optimize due to gradient vanishing and explosion (Hochre-

Why Transformers?

Can be pre-trained on huge amounts of unlabeled data

- E.g., all of Wikipedia, book corpus, etc.

Fine-tuned later to a specific task

- E.g., question-answering, sentiment analysis, text-classification
- with just a small amount of labeled, domain-specific data

Feature millions/billions of parameters

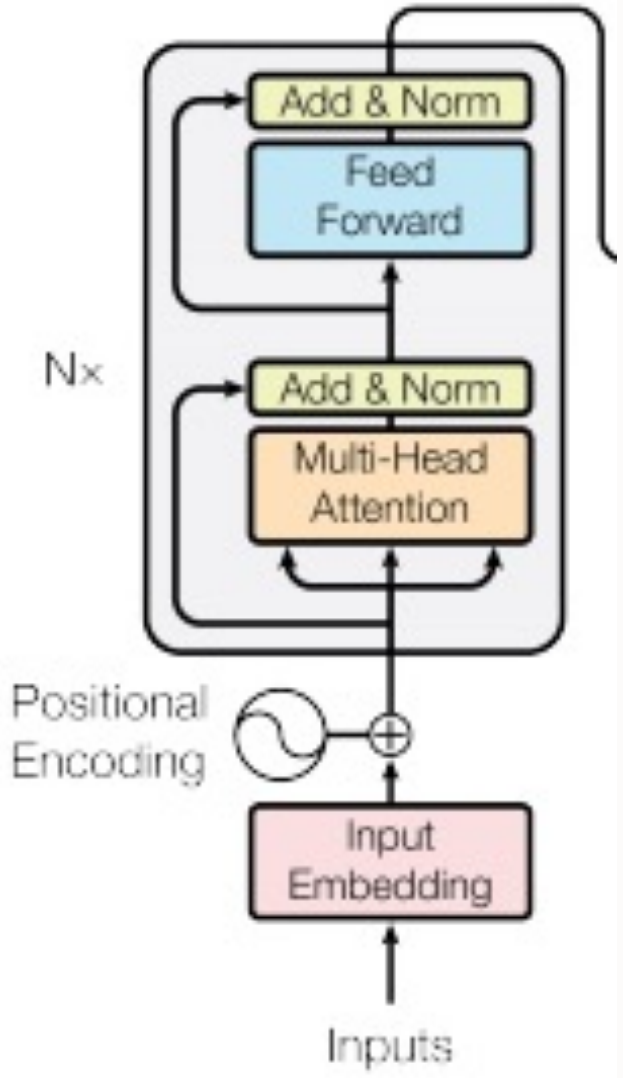
➔ Lots of attention on optimizing the training process

- (relatively) less attention on optimizing inference

Why CPUs?

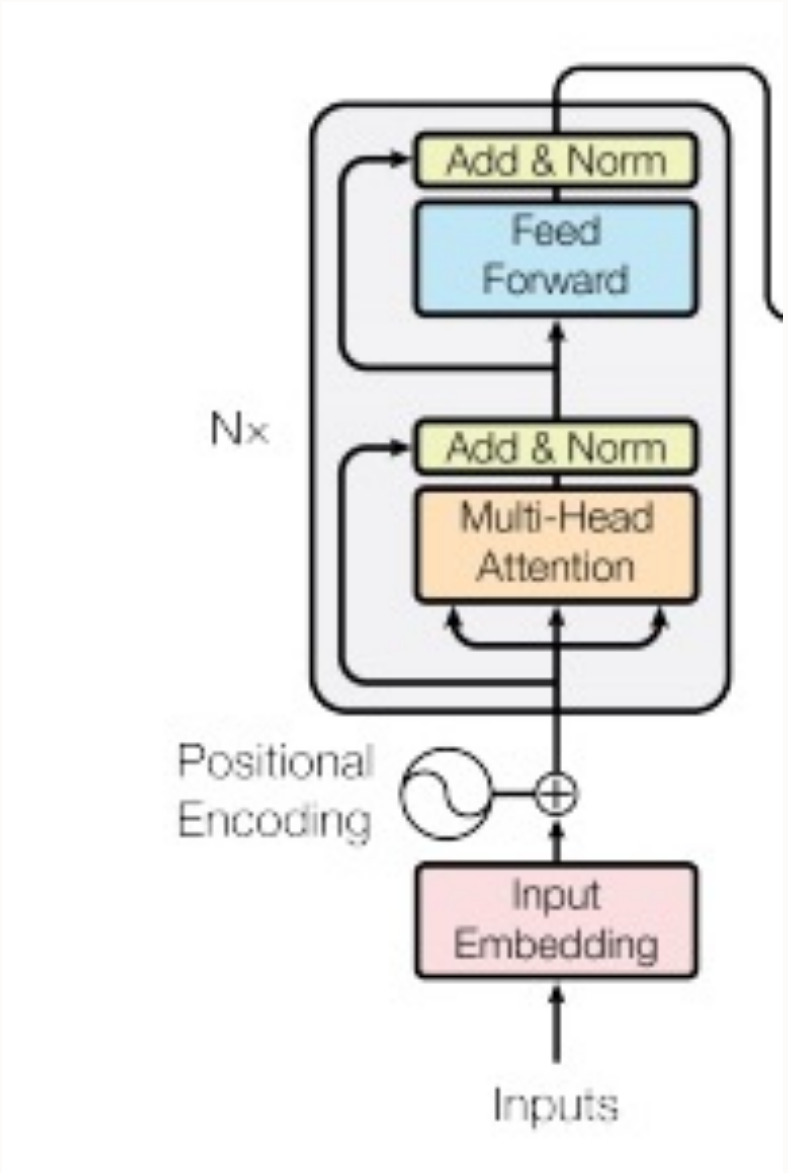
- Commodity hardware
- Cost-effective choice for real-time (aka batch=1) inference
- Preferred choice for inference deployment in industry

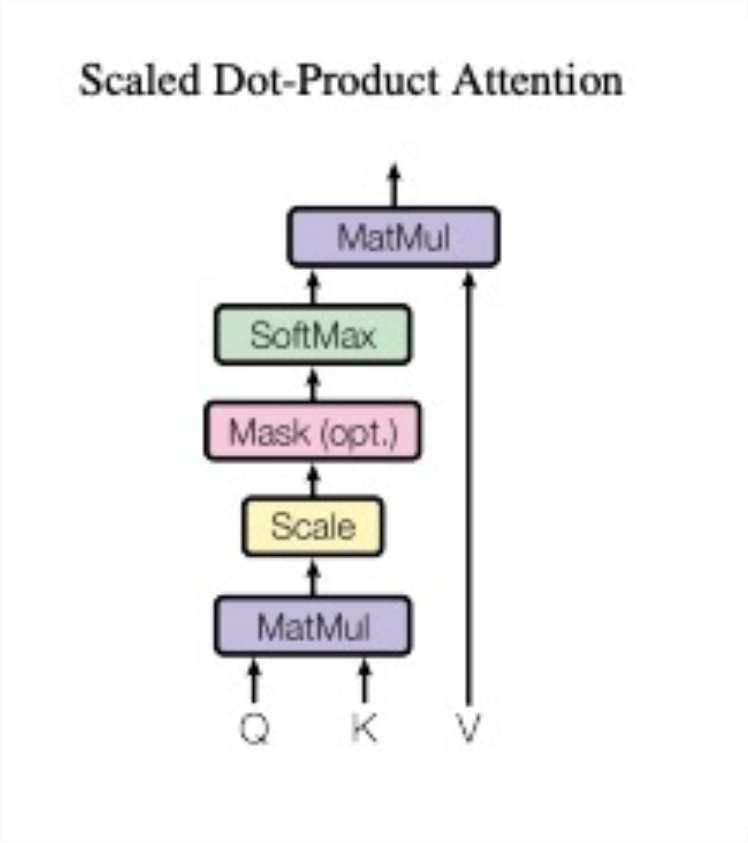
Transformer Architecture

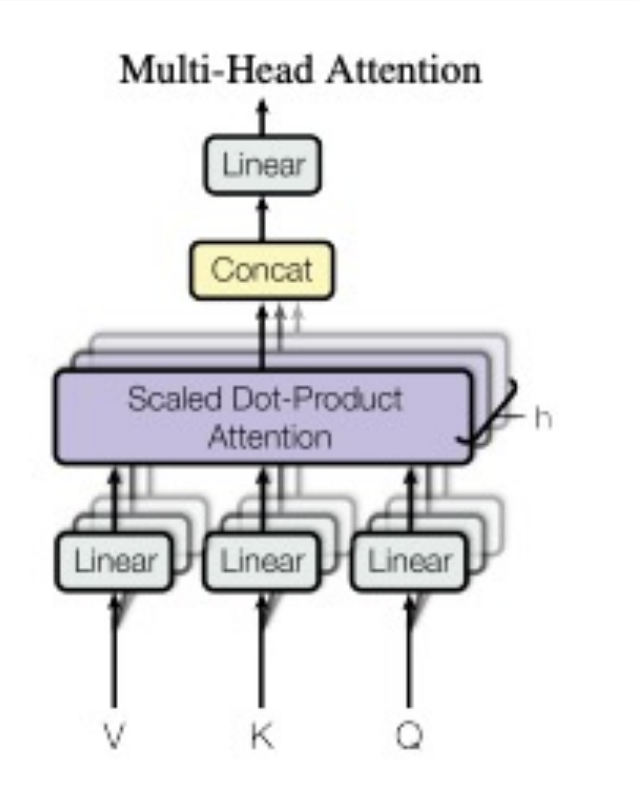


From “Attention is all you need” by Vaswani et el., 2017

Transformer Architecture







From “Attention is all you need” by Vaswani et el., 2017

Our setup

OCI BM.Standard2.52 instance

- Intel Skylake-gen processor with 26 hyper-threaded cores
- AVX512 support

Pytorch (v1.6)

oneDNN math library

- enhanced integration of oneDNN with Pytorch

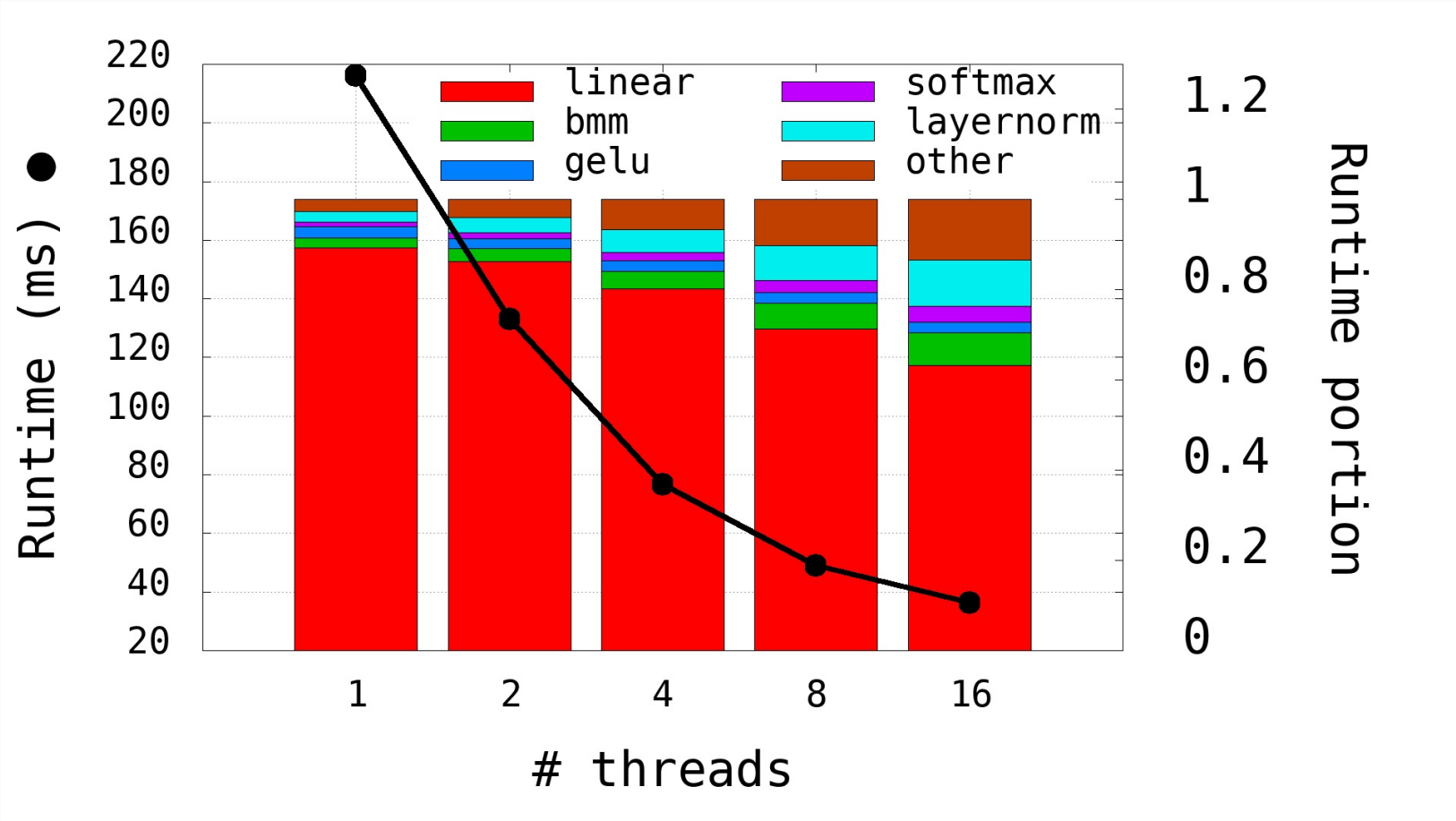
Transformers python package (v3.0.2) from HuggingFace

- state-of-the-art implementation of numerous Transformer-based NLP models

“BERT-base” model (from Transformers)

Runtime breakdown by modules

- Relatively poor scalability
  - ~5x from 1 to 16 threads
- Matmuls scale better than other ops
  - their share decreases with increase in #threads
- Between 66 and 92% of the time is spent in matmul ops



Key observation

The matmul operation is heavily impacted not only by the shape (dimensions) of the source matrices and available resources (#threads), but also by the form (transposed/normal) of those matrices

- the form dictates the memory layout (column-major vs. row-major)

A subset of Basic Linear ALgebra (BLAS) functions to perform matrix-matrix multiplication. More...

Functions

mkldnn_status_t MKLDNN_API mkldnn_sgemm (char transa, char transb, mkldnn_dim_t M, mkldnn_dim_t N, mkldnn_dim_t K, float alpha, const float *A, mkldnn_dim_t lda, const float *B, mkldnn_dim_t ldb, float beta, float *C, mkldnn_dim_t ldc)

SGEMM performs a matrix-matrix multiplication operation defined as. More...

Source: https://oneapi-src.github.io/oneDNN/v1.0/group__c__api__blas.html

Key observation

Why is the difference?

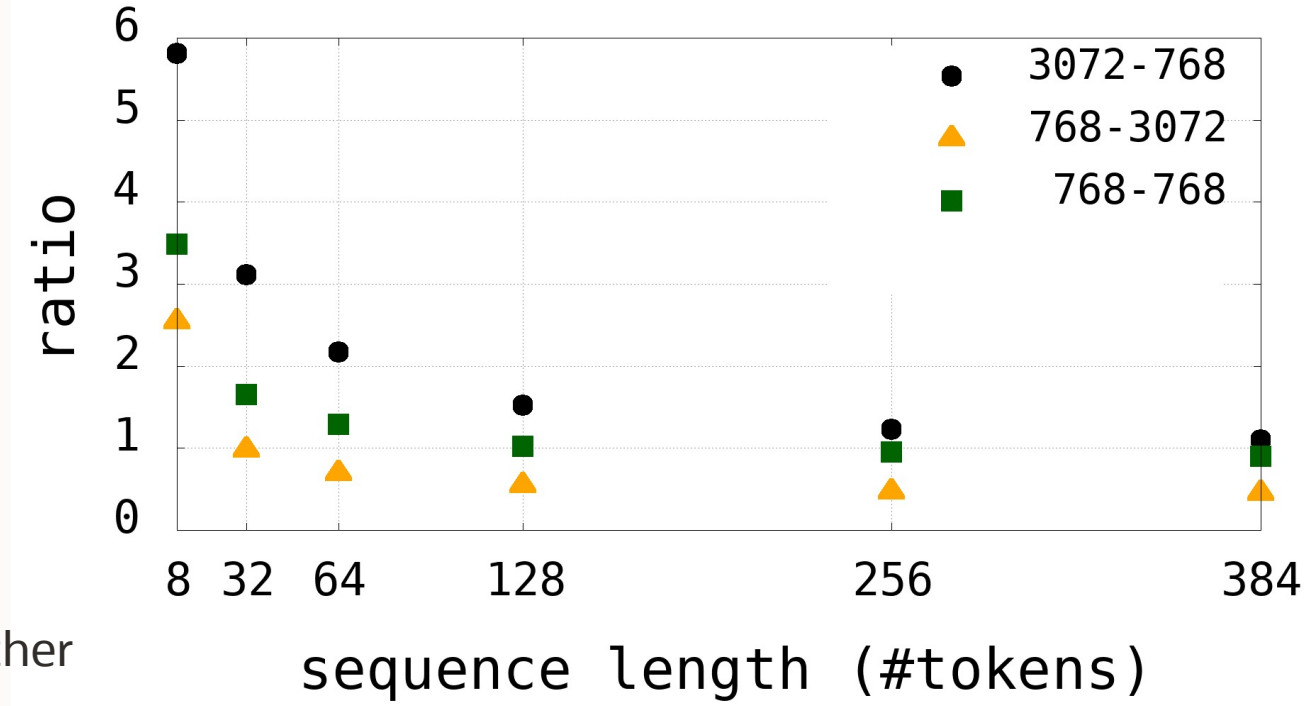
- different matmul variants generate different code paths and memory access patterns
- perf. counters suggest faster variants enjoy better L2 cache locality

Why does it matter?

- Pytorch keeps the weights matrix in the Linear module transposed
  - Tensorlfow keeps the weights matrix normal

➔ Matmuls are <u>always</u> applied in one form or another

☞ But the best approach – adaptivity!



$$\text{ratio} = \frac{\text{A*B, where A and B are normal}}{\text{A*B, where A is normal and B is transposed}}$$

Adaptive Linear Module Optimization (ALMO)

Augment each Linear module* with `transposeFlags` array of Boolean flags

When creating a Linear module with the shape [in ,out]:

- For each i in [0, 9]:
  - Generate random matrix with the shape [2^i, in]
  - Measure time to perform matmul with the weight matrix transposed and normal
  - Record the result (0/1) in `transposeFlags[i]`

During inference, with the input of shape [length, in]:

- Calculate `s = int(log(length))`
- Based on `transposeFlags[s]`, perform matmul with weights either transposed or not

* `transposeFlags` can be shared among Linear modules of the same shape

Experimental results

#threads	sequence length								
	8			64			384		
	onednn base	onednn normal	onednn almo	onednn base	onednn normal	onednn almo	onednn base	onednn normal	onednn almo
1	115	82	79 ± 1 (x1.46)	216	193	162 (x1.33)	884 ± 31	1009	807 (x1.10)
2	83	50	50 ± 1 (x1.66)	133	105	97 ± 1 (x1.37)	471 ± 29	512	413 (x1.14)
4	51	34	34 (x1.50)	76	64	60 (x1.27)	259 ± 27	285	220 (x1.18)
8	35	27	27 (x1.30)	49	45	42 (x1.17)	135	148	128 (x1.05)
16	28	24	24 (x1.17)	36	34	32 (x1.12)	96 ± 14	97	82 (x1.17)

BERT-base inference latency (ms)

Similar results for other models: RoBERTa, DistilBERT

Conclusions

- Faster matmul operations → faster inference

- Adaptive Linear Module Optimization
  - lightweight
  - transparent (no change to the model)
  - does not impact accuracy
  - is not limited to Transformers

- More details in the extended paper (https://arxiv.org/pdf/2102.06621):
  - Two more optimizations for the matmul operations
    - Sequential overhead reduction
    - Modified matrix partitioning
  - More experiments, results, etc.