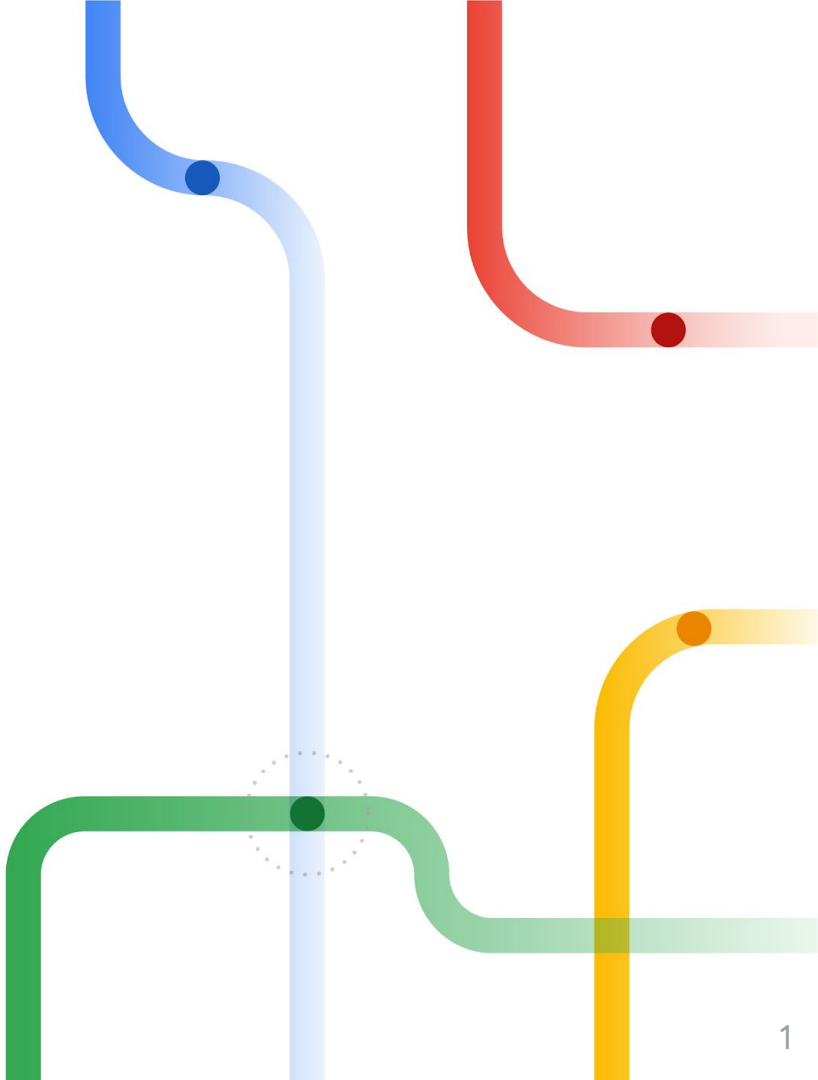
Sparse is Enough in Scaling Transformers

NeurIPS 2021

Authors: Sebastian Jaszczur, Aakanksha Chowdhery, Afroz Mohiuddin, Łukasz Kaiser, Wojciech Gajewski, Henryk Michalewski, Jonni Kanerva

Google Research



Agenda

01 Introduction

02 Sparse Feed Forward

03 Sparse QKV Layer

04 Enabling gains for Long sequences

05 Future Possibilities

01

Introduction

Motivation

Over time bigger models are used, as they use data more efficiently.

What about running fast inference on non-specialized hardware?

We look at big Transformer models for text synthesis.

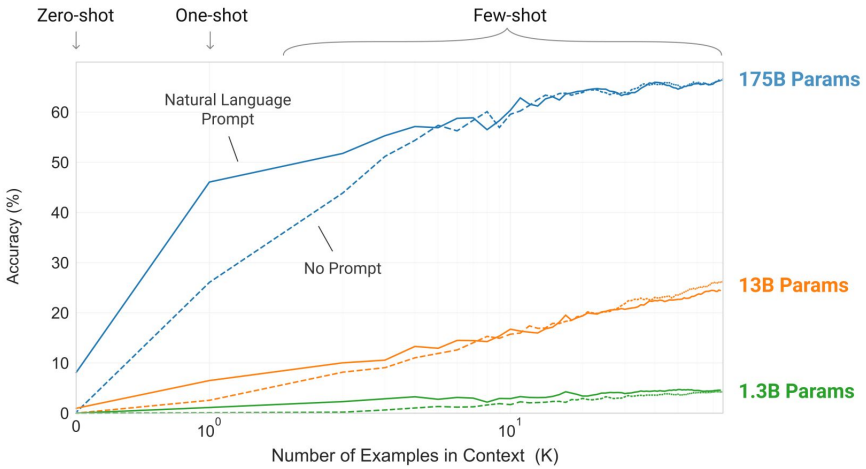
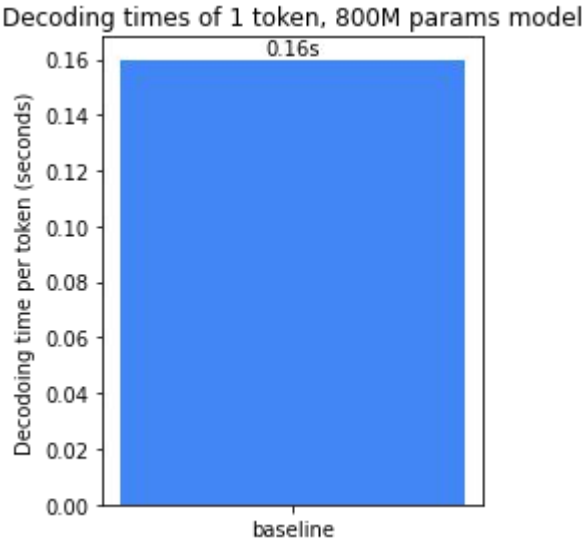


Figure 1.2 from *Language Models are Few-Shot Learners*, Brown et al. 2020

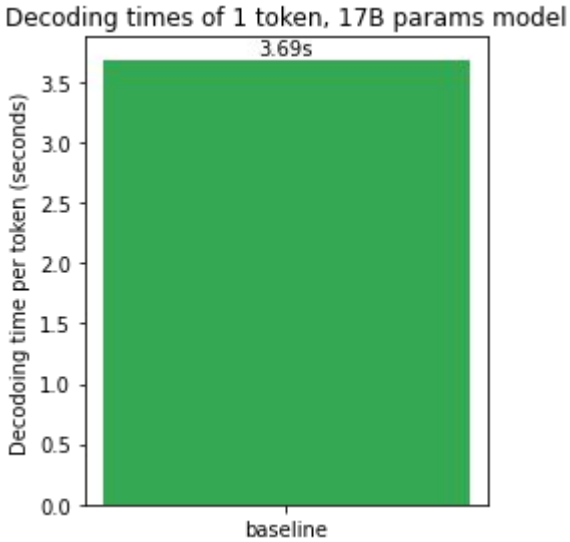
Larger models make increasingly efficient use of in-context information.

Motivation - Inference with Large Models

Running unbatched text generation on non-specialized hardware (just CPU):



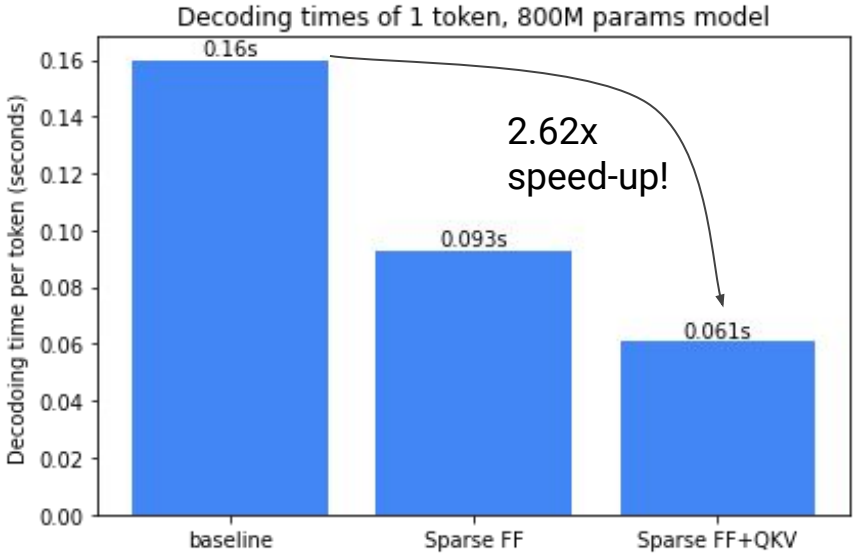
~4 seconds per sentence



~90 seconds per sentence

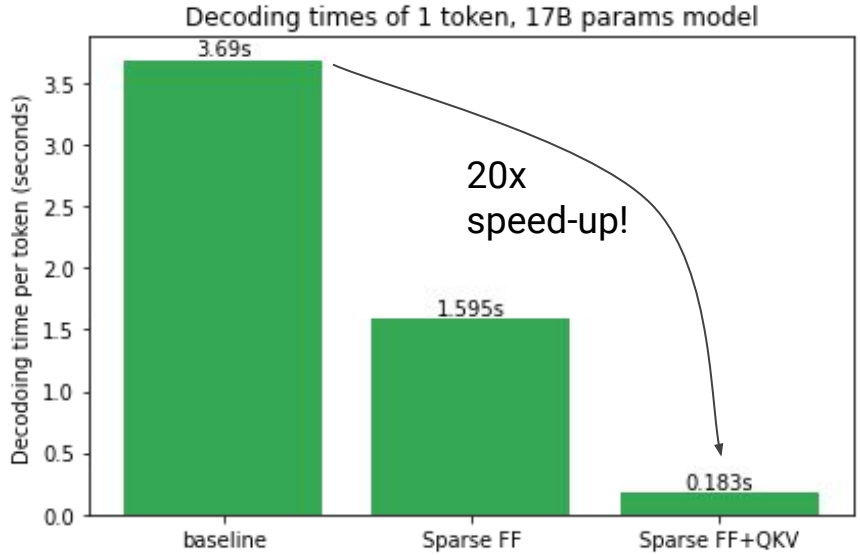
Motivation - Inference with Large Models

Running unbatched text generation on non-specialized hardware (just CPU):



~4 seconds per sentence

to ~1.5 seconds



~90 seconds per sentence

to ~4.5 seconds!

Reducing Inference Cost: Prior Approaches

Prior approaches to reduce inference time include:

- Model distillation[1]
- Model compression/pruning[2]
- Quantization [3]

Our work focuses instead on conditional skipping of parameters, like in Mixture of Experts[4].

[1] Kim et al., 2020, Fastformers: Highly efficient transformer models for natural language understanding
[2] Li et al., 2020, Train big, then compress: Rethinking model size for efficient training and inference of transformers
[3] Shen et al., 2020, Q-bert: Hessian based ultra low precision quantization of bert
[4] Shazeer et al., 2017, Outrageously large neural networks: The sparsely-gated mixture-of-experts layer.

Are sparse models enough?

We show that sparse model can perform just as well as a dense model with the same number of parameters.

To achieve this we designed:

- Sparse Feed Forward
- Sparse QKV

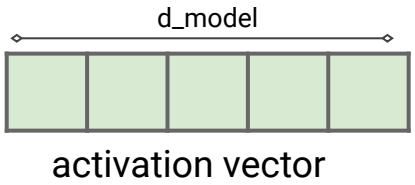
02

Sparse Feed Forward

Standard
Feed Forward Layer

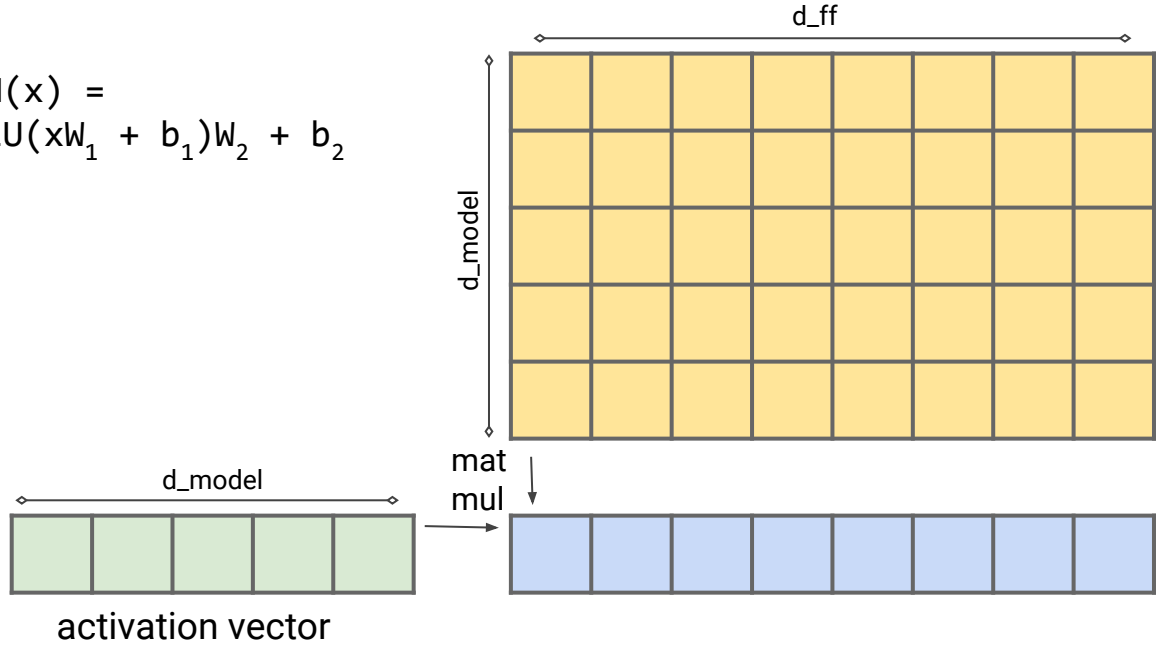
Standard Feed Forward Layer

$$FNN(x) = ReLU(xW_1 + b_1)W_2 + b_2$$



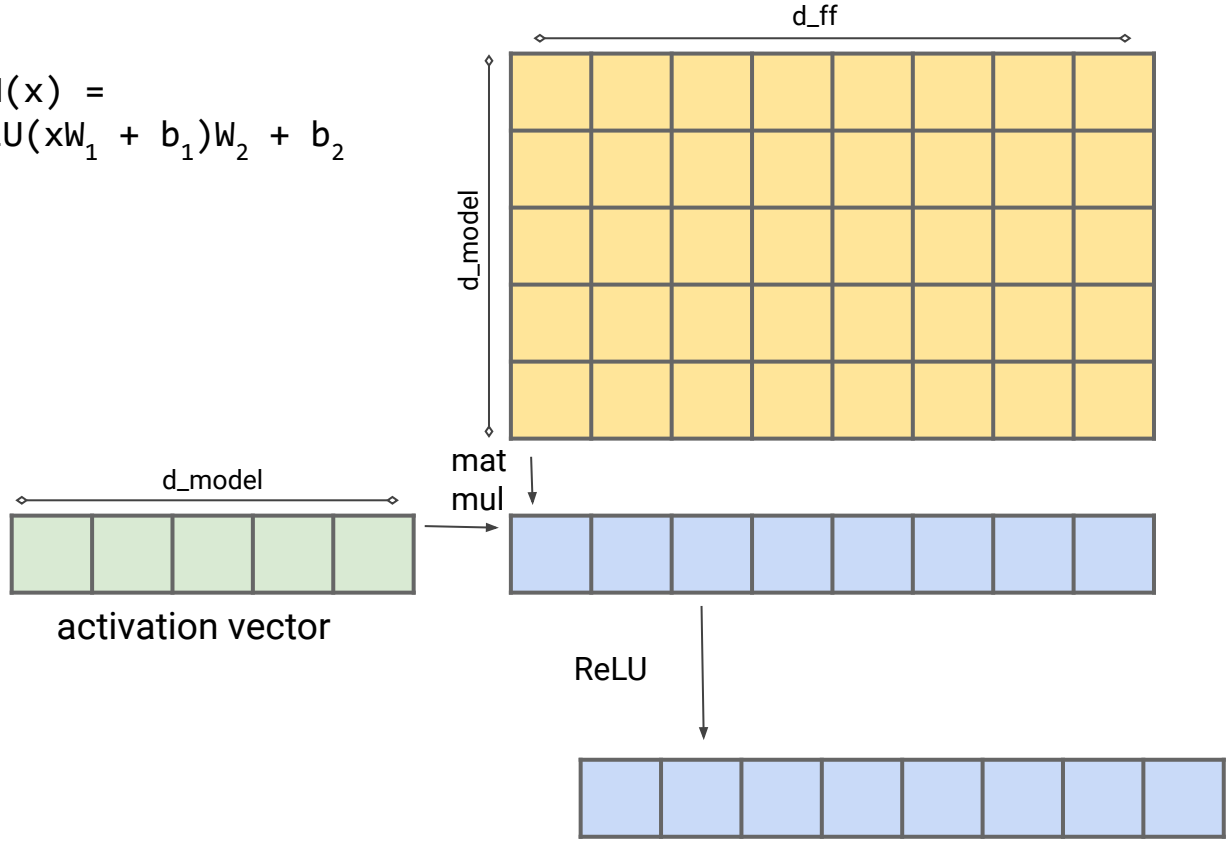
Standard Feed Forward Layer

$$FNN(x) = ReLU(xW_1 + b_1)W_2 + b_2$$



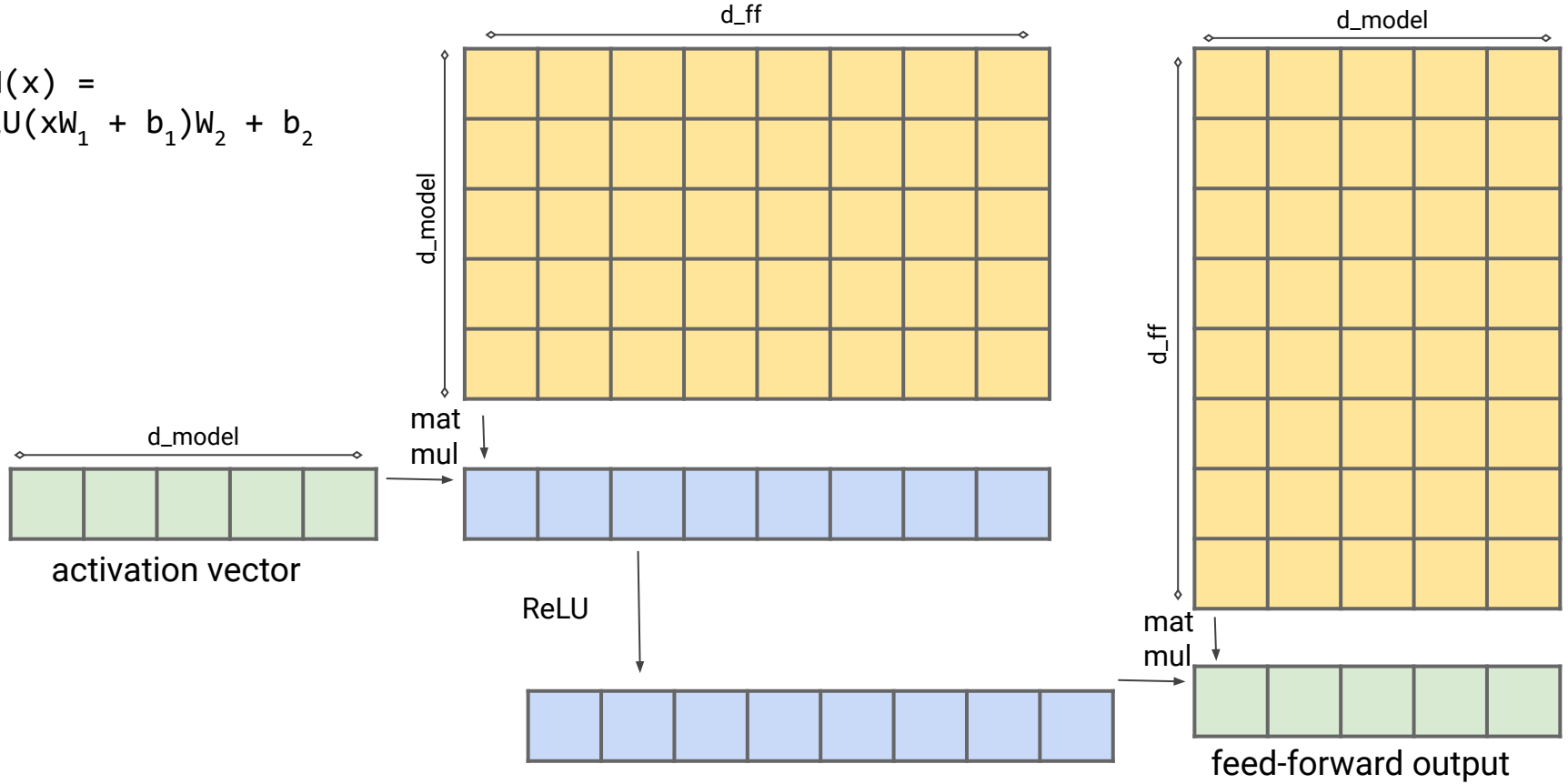
Standard Feed Forward Layer

$$FNN(x) = ReLU(xW_1 + b_1)W_2 + b_2$$



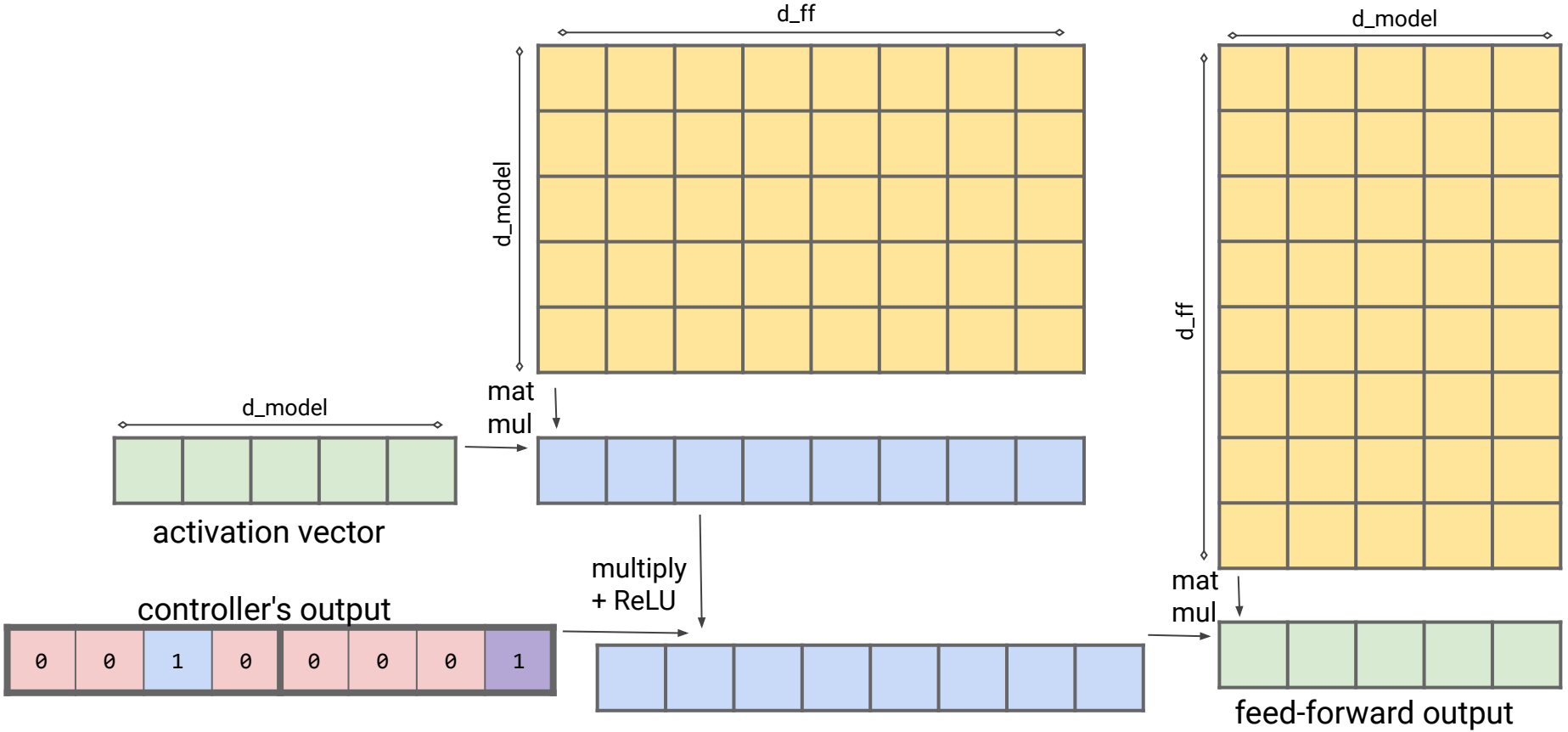
Standard Feed Forward Layer

$$FNN(x) = ReLU(xW_1 + b_1)W_2 + b_2$$

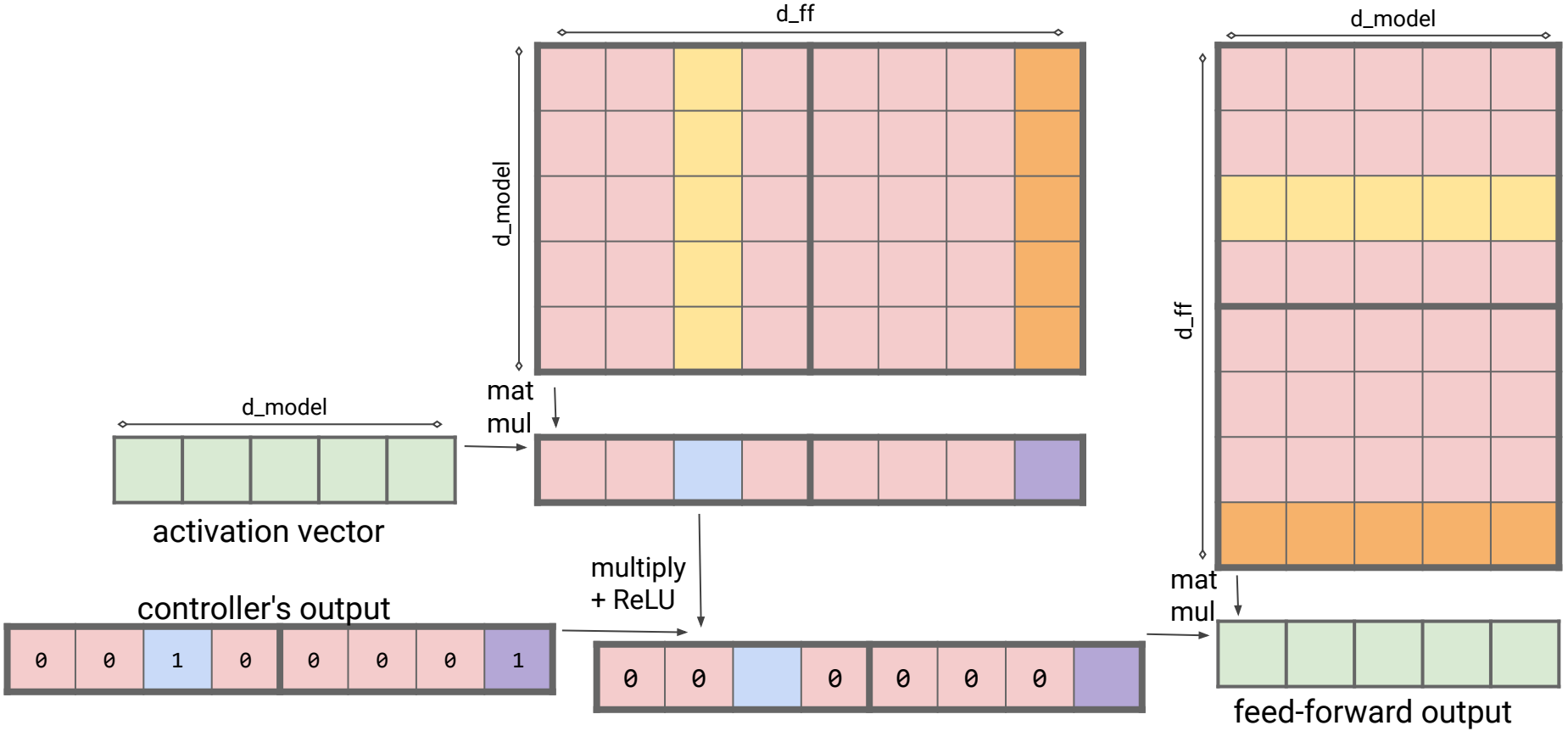


Sparse Feed Forward

Sparse Feed Forward

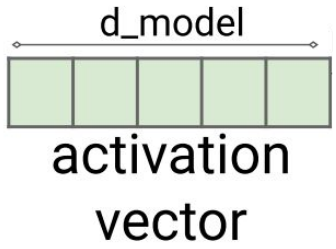


Sparse Feed Forward

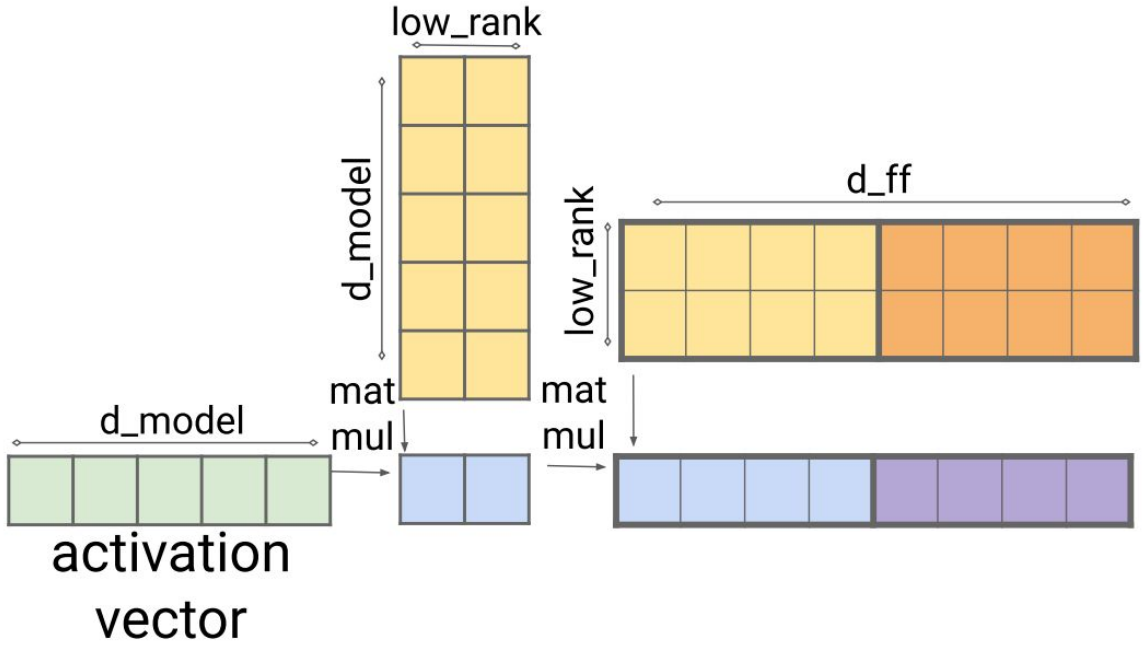


Sparse Feed Forward: Controller

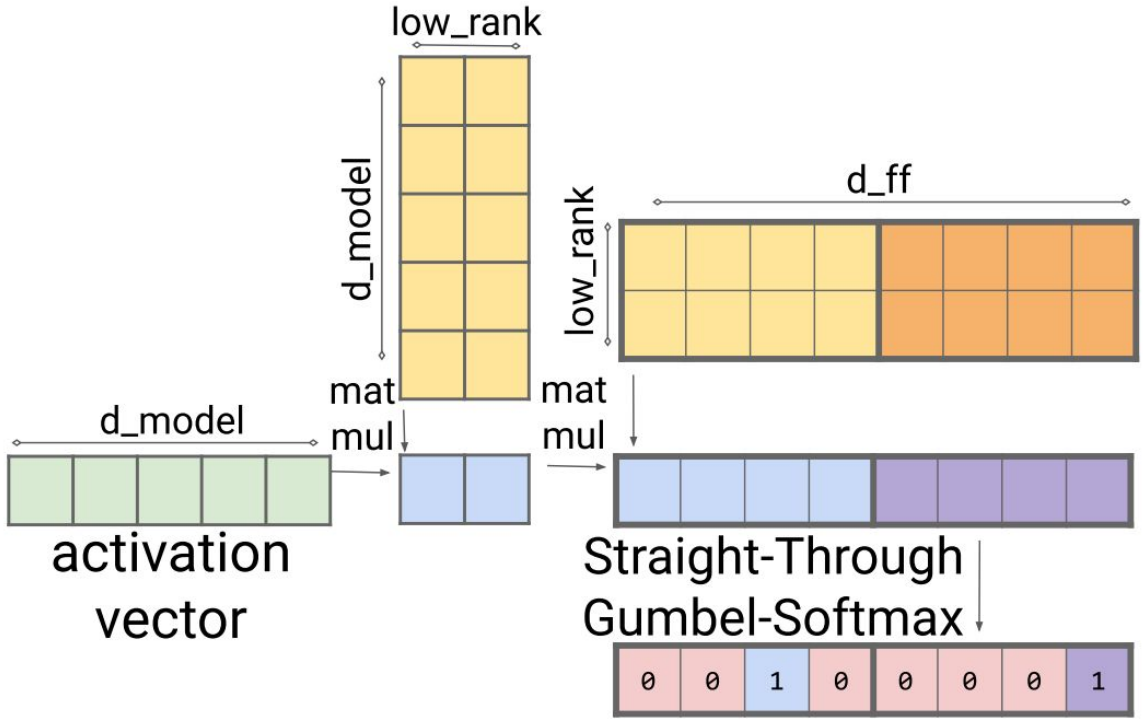
How to design Controller?



How to design Controller?

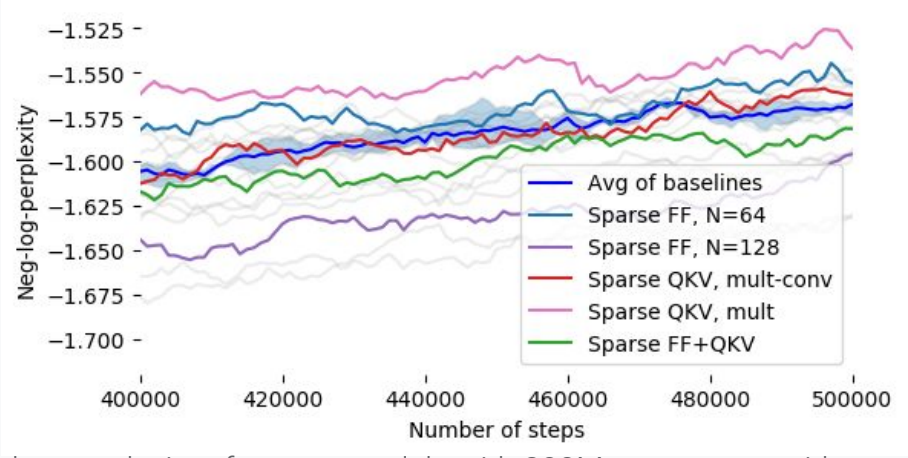


How to design Controller?



Results

Sparse FF doesn't impact model quality!

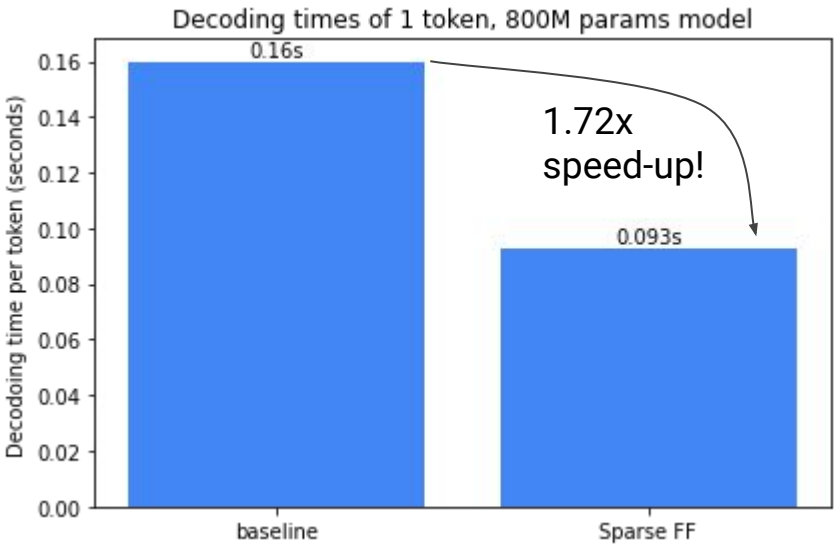


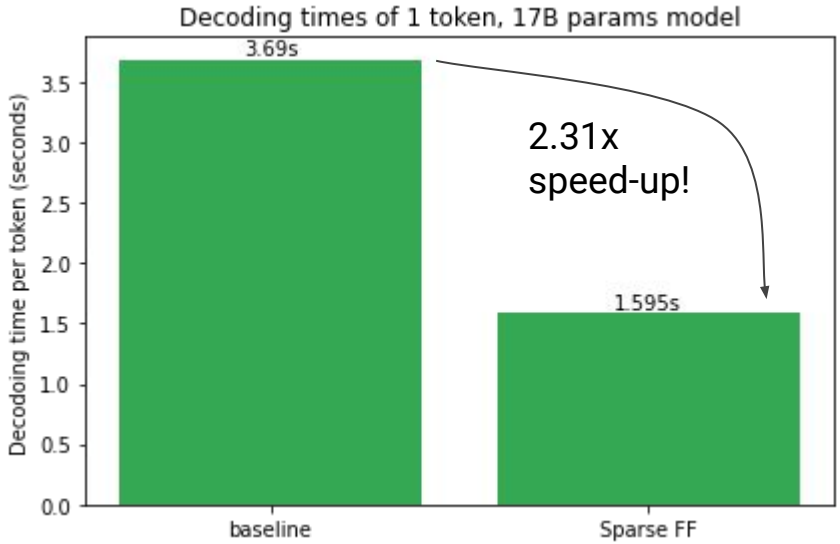
Neg-log-perplexity of sparse models with 800M parameters with proposed sparsity mechanisms matches baselines.

Model	R-1	R-2	R-LSum	R-LSent
Terraformer	45.40	17.86	41.21	26.33
DANCER RUM	42.70	16.54	38.44	—
BIGBIRD-RoBERTa	41.22	16.43	36.96	—
Pegasus Large (C4)	44.21	16.95	38.83	25.67
DANCER PEGASUS	45.01	17.6	40.56	—
BIGBIRD-Pegasus	46.63	19.02	41.77	—

Our model is also competitive with strong baselines on ArXiv summarization task.

How to design Controller?





~4 seconds per sentence

to ~2.3 seconds

~90 seconds per sentence

to ~40 seconds!

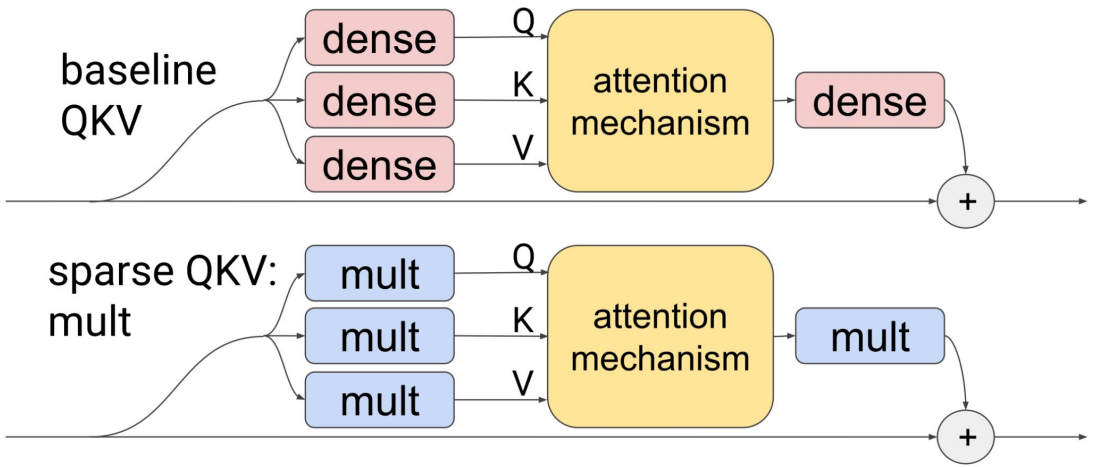
03

Sparse QKV Layer

Sparse QKV Layer - Simple Variant

Our sparse variant:

- has order of magnitude less parameters:
  down to even $O(d_{model}^{1.5})$
  instead of $O(d_{model}^{2})$
- can express any permutation!



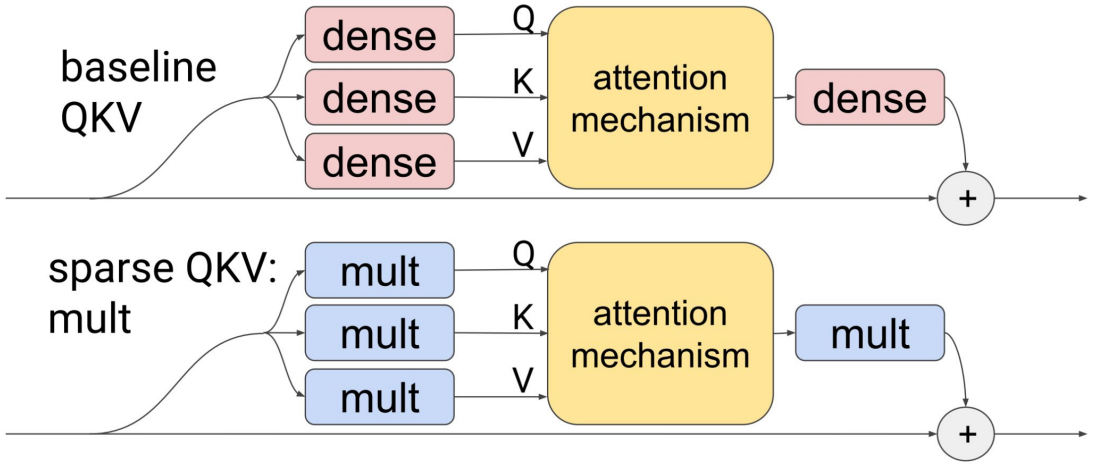
Sparse QKV Layer - Simple Variant

Our sparse variant:

- has order of magnitude less parameters:
  down to even $O(d_{model}^{1.5})$
  instead of $O(d_{model}^{2})$
- can express any permutation!

We increase d_{ff} to keep #params in the model

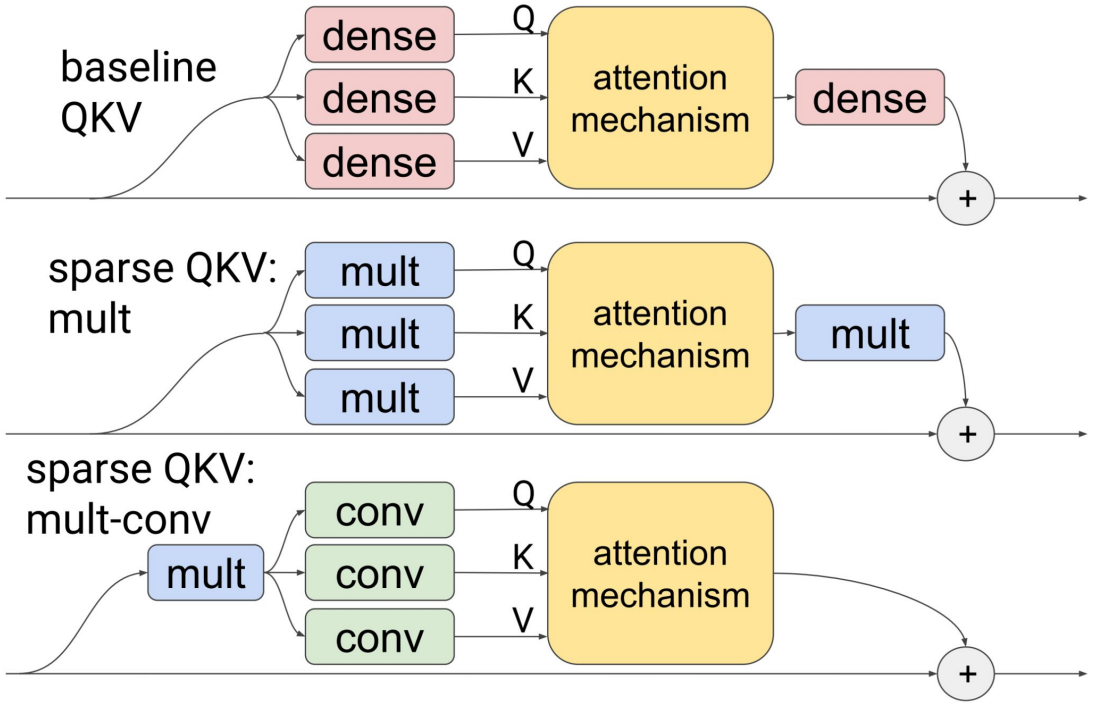
- basically moving params from QKV layer to FF!



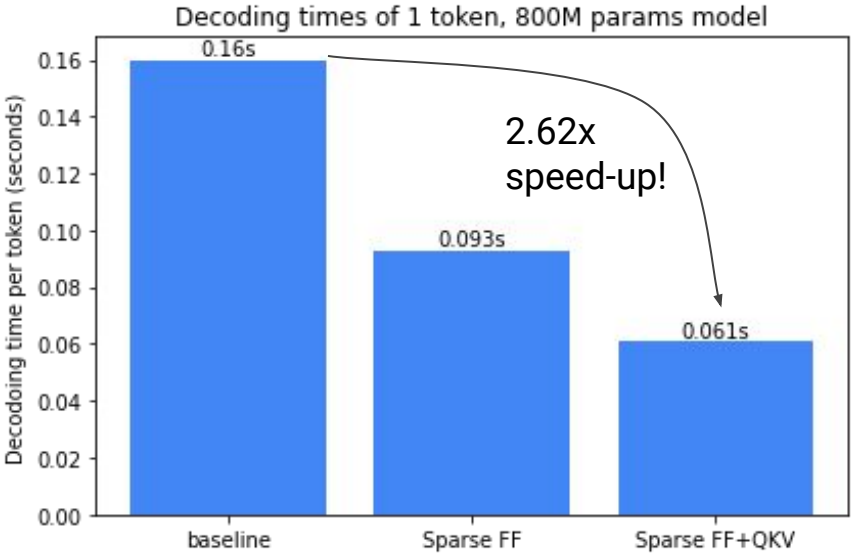
Sparse QKV Layer - Convolution Variant

We can join this permutation layer with convolution in order to get better speed at no cost in model quality.

Please refer to our paper for details.

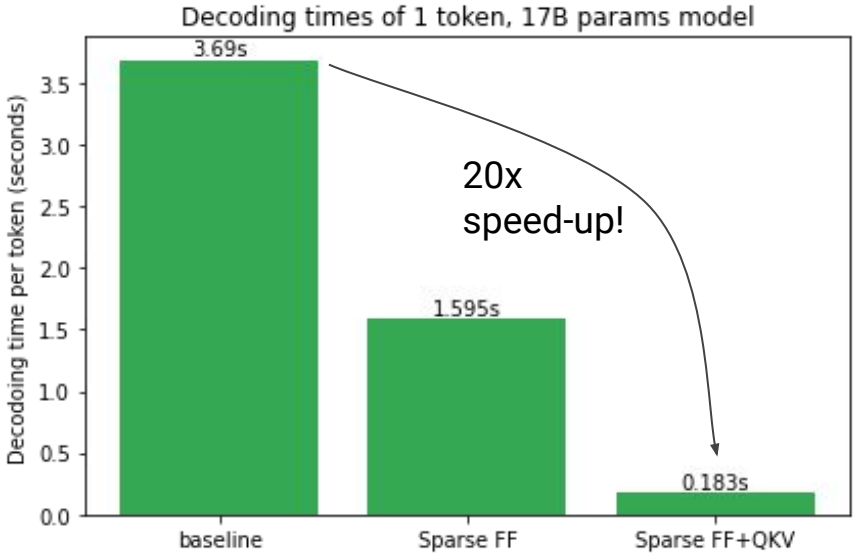


Sparse QKV - Inference Time Improvement



~4 seconds per sentence

to ~1.5 seconds



~90 seconds per sentence

to ~4.5 seconds!

04

Enabling gains for Long sequences

Integration with prior approaches

To enable decoding on long sequences we've added to our models:

- reversibility[1]
- LSH attention[1]
- recurrence in the form of SRU[2]

The project and code is open-source.

[1] Kitaev et al., 2020, Reformer: the efficient Transformer
[2] Lei et al., 2017, Training rnns as fast as cnns

05

Future Work

Future Work

We have shown sparse models perform as well as dense models with same # of params with an order of magnitude speedup in decoding.

- Enable gains for Batched inference
- Use Sparse FF layer to improve training time
- Enable gains for other domains, for e.g. vision transformers

Thank you!

NeurIPS 2021

Authors: Sebastian Jaszczur, Aakanksha Chowdhery,
Afroz Mohiuddin, Łukasz Kaiser, Wojciech Gajewski,
Henryk Michalewski, Jonni Kanerva

Google Research