Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity

Presenters: Zhou Lu, Wenhan Xia

Background

Dense vs Sparse Models

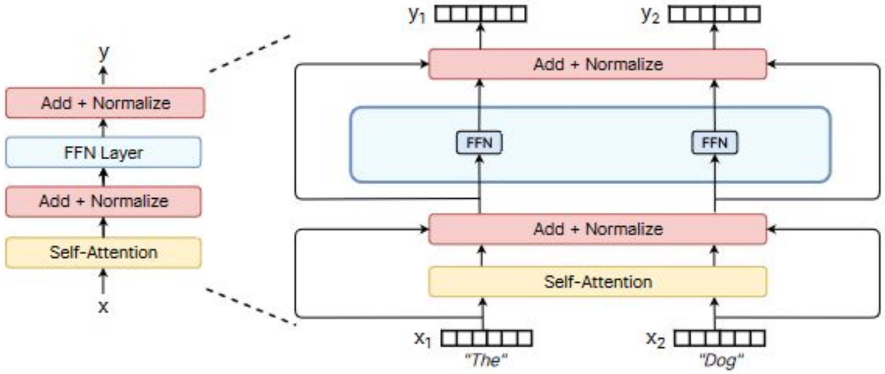
Dense model (e.g. GPT3)

- Most popular
- Excellent performance
- Expensive training and computation

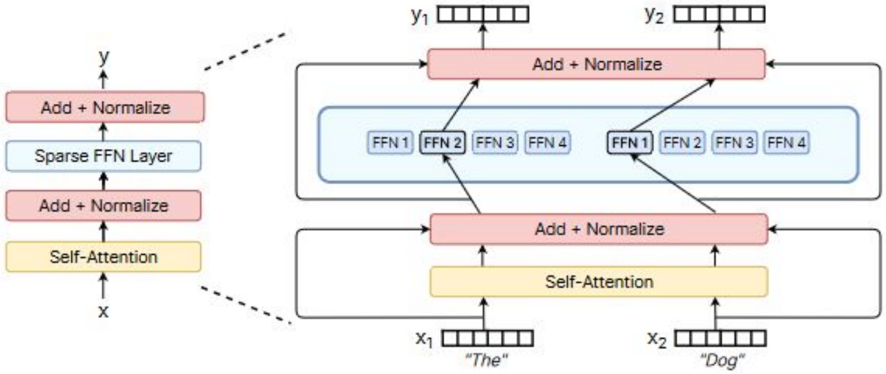
Sparse model

- Less popular
- Good performance
- Potentially cheaper computation

Dense Model



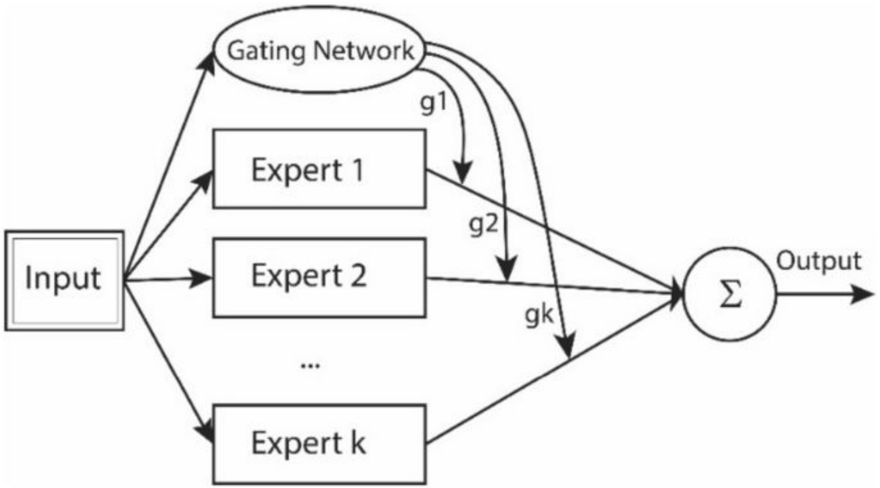
Sparse Model



How to make inference more computationally efficient?

Mixture of Experts (MoE)

- Train many experts (models), expensive training
- Route an input to a few experts, cheap inference



History of MoE

The idea of mixture of experts has been 30 years already

- Adaptive mixtures of local experts. JJNH91
- Twenty years of mixture of experts. YWG12
- Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. **Shazeer et al 17**
- Gshard (Levipkhin et al 20) and Switch Transformers (FZS17)

Shazeer et al 17

- The first work that made MoE works well
- Train the largest model and achieve state-of-the-art results

Method:

- Train many neural networks as the candidate set of experts
- Train a gating network to map the input to a few experts

The MoE (gating) layer

Let $h(x)$ be the initial output, use softmax to get weights

$$p_i(x) = \frac{e^{h(x)_i}}{\sum_j^N e^{h(x)_j}}.$$

Final output is the convex combination of experts

$$y = \sum_{i \in \mathcal{T}} p_i(x) E_i(x).$$

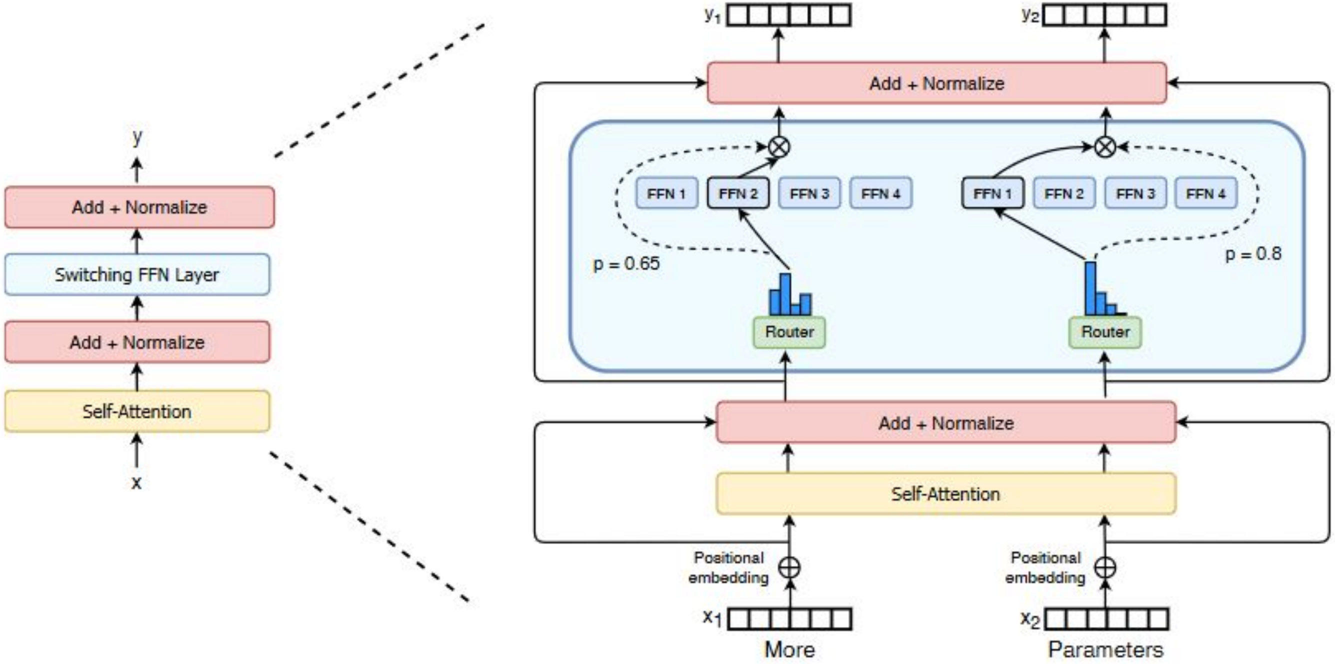
Typically, we consider only the top-k experts where k<N

Some Technical Challenges

- Complexity
- Communication costs
- Training instabilities

Switch transformer

- The guiding design principle: maximizing the parameter count efficiently
- A fourth axis: increasing the parameter count, keeping FLOPs constant
- The sparsely activated layers split unique weights on different devices



Dense feed forward network (FFN) layer is replaced by a sparse Switch FFN layer (light blue box)

New ingredients

- Switch routing
- Distributed switch implementation
- Differentiable load balance loss

Switch routing

- Previous method: using top-k experts out of N experts
- Now routing to only a single expert

Advantages:

1, Reduced routing computation

2, Reduced communication cost

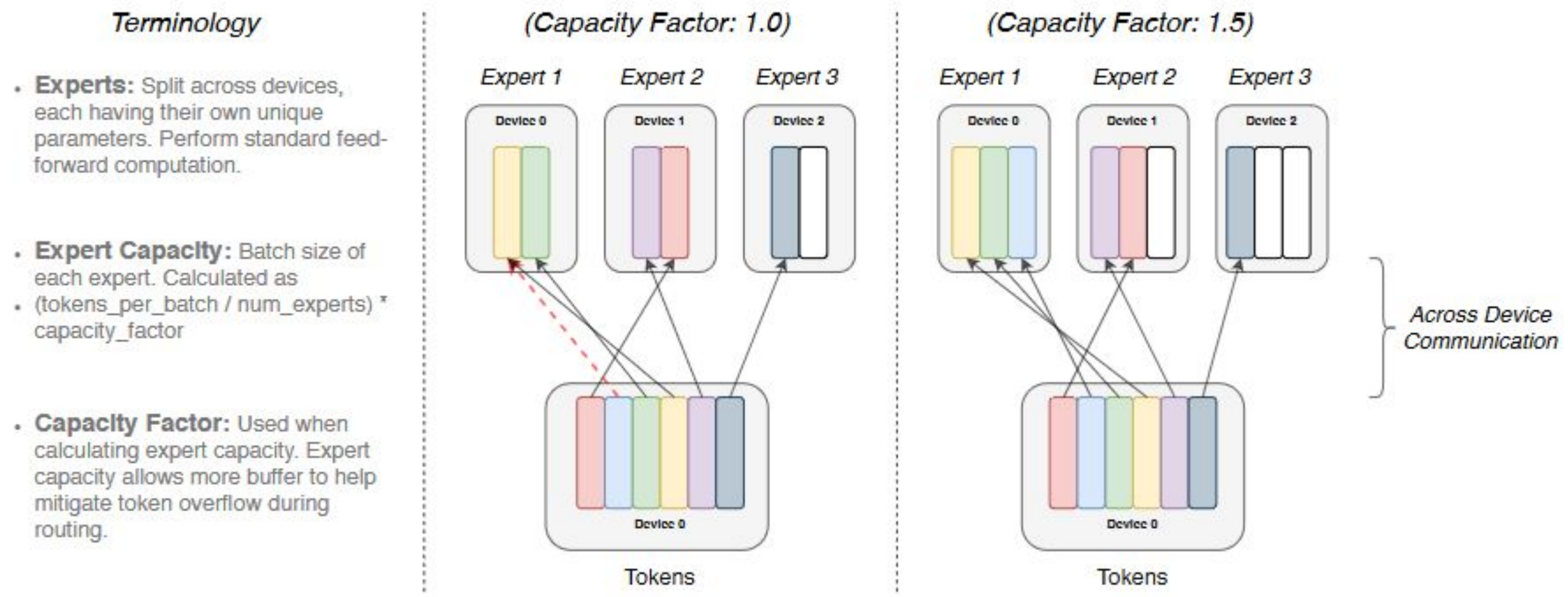
3, Better performance

Distributed switch implementation

Setting the expert capacity: the number of tokens each expert computes

$$\text{expert capacity} = \left(\frac{\text{tokens per batch}}{\text{number of experts}}\right) \times \text{capacity factor}.$$

- Capacity factor = 1: potential overflow issue
- Capacity factor > 1: additional buffer for imperfect distribution



Tradeoff: a larger capacity factor alleviates this overflow issue, but also increases computation and communication costs

A differentiable load balancing loss

Given N experts and a batch with T tokens, we add an auxiliary loss:

$$\text{loss} = \alpha \cdot N \cdot \sum_{i=1}^{N} f_i \cdot P_i$$

f_i is the fraction of tokens dispatched to expert i

$$f_i = \frac{1}{T} \sum_{x \in \mathcal{B}} \mathbb{1}\{\operatorname{argmax} p(x) = i\}$$

P_i is the fraction of the router probability allocated for expert i

$$P_i = \frac{1}{T} \sum_{x \in \mathcal{B}} p_i(x).$$

Why such loss?

- The paper wants both vectors to have values of 1/N
- It’s claimed that the auxiliary loss encourages uniform routing since it is minimized under a uniform distribution

$$\sum_{i=1}^{N}(f_i \cdot P_i) = \sum_{i=1}^{N}(\frac{1}{N} \cdot \frac{1}{N}) = \frac{1}{N}$$

Rethinking the loss choice

The claim is wrong: minimal value can be smaller than 1/N, achieved by non-uniform distributions. Consider this example with N=2, T=3

	Expert 1	Expert 2
Token 1	0.51	0.49
Token 2	0.51	0.49
Token 3	0	1

$$f = \left(\frac{2}{3}, \frac{1}{3}\right), \qquad P = (0.34, 0.66), \qquad \langle f, P \rangle = 0.447 < \frac{1}{2}$$

Open question: can we design a better loss?

Putting It All Together: The Switch Transformer

First test of Switch Transformer is on “Colossal Clean Crawled Corpus” (C4)

- A masked language modeling task is used for the pre-training objective
- 15% of tokens are dropped out and replaced by the masked sequence
- The negative log perplexity is recorded to compare the models

Model	Capacity Factor	Quality after 100k steps (↑) (Neg. Log Perp.)	Time to Quality Threshold (↓) (hours)	Speed (↑) (examples/sec)
T5-Base	—	-1.731	Not achieved†	1600
T5-Large	—	-1.550	131.1	470
MoE-Base	2.0	-1.547	68.7	840
Switch-Base	2.0	-1.554	72.8	860
MoE-Base	1.25	-1.559	80.7	790
Switch-Base	1.25	-1.553	65.0	910
MoE-Base	1.0	-1.572	80.1	860
Switch-Base	1.0	-1.561	**62.8**	1000
Switch-Base+	1.0	**-1.534**	67.6	780

Switch transformers are better, fixing time or quality

Key findings

- Switch Transformers outperform both carefully tuned dense models and MoE Transformers on a speed-quality basis.

- The Switch Transformer has a smaller computational footprint

- Switch Transformers perform better at lower capacity factors (1.0, 1.25)

Improved Training and Fine-Tuning Techniques

- Selective precision with large sparse models
- Smaller parameter initialization for stability
- Regularizing large sparse models

Selective precision with large sparse models

- Instability hinders the ability to train using efficient bfloat16 precision
- Casting expensive float32 precision only on the router function
- Benefit from efficiency of bfloat16 and stability of float 32

Model (precision)	Quality (Neg. Log Perp.) (↑)	Speed (Examples/sec) (↑)
Switch-Base (float32)	-1.718	1160
Switch-Base (bfloat16)	-3.780 [*diverged*]	**1390**
Switch-Base (Selective precision)	**-1.716**	1390

Selective precision achieves benefits on both quality and speed

Smaller parameter initialization

- The weight matrices are initialized by sampling from a truncated normal distribution with mean $\mu = 0$ and standard deviation $\sigma = \sqrt{s/n}$
- We reduce the default initialization scale s = 1.0 by a factor of 10
- This both improves quality and reduces the likelihood of instability

Model (Initialization scale)	Average Quality (Neg. Log Perp.)	Std. Dev. of Quality (Neg. Log Perp.)
Switch-Base (0.1x-init)	**-2.72**	**0.01**
Switch-Base (1.0x-init)	-3.60	0.68

Smaller parameter initialization improves both quality and stability

Regularizing large sparse models

- Overfitting is an issue: many fine-tuning tasks have very few examples
- Switch Transformers have more parameters: more severe overfitting
- A simple remedy: increasing the dropout inside the experts, which we name as expert dropout

Model (dropout)	GLUE	CNNDM	SQuAD	SuperGLUE
T5-Base (d=0.1)	82.9	**19.6**	83.5	72.4
Switch-Base (d=0.1)	84.7	19.1	**83.7**	**73.0**
Switch-Base (d=0.2)	84.4	19.2	**83.9**	**73.2**
Switch-Base (d=0.3)	83.9	19.6	83.4	70.7
Switch-Base (d=0.1, ed=0.4)	**85.2**	**19.6**	**83.7**	**73.0**

A smaller dropout rate (0.1) at non-expert layers and a larger dropout rate (0.4) at expert layers is the best

Scaling properties

- When the model is not bottlenecked by computation or amount of data
- Use the large C4 corpus and train until diminishing returns
- Increasing the experts keeps the computational cost approximately fixed

Scaling versus

- Fixed training steps: more parameters (experts) speeds up training

- Fixed training time: Switch Transformers yield a substantial speed-up

- Large dense models: Switch-Base is still more sample efficient and yields a 2.5x speedup

Improved language learning abilities for **downstream applications**

Downstream Results

- Fine-tuning
- Distillation
- Multilingual Learning

Downstream Results

- **Fine-tuning**
- Distillation
- Multilingual Learning

Downstream: fine-tuning

mixture of tasks (sentiment analysis, sentence similarities etc)

Model	GLUE	SQuAD	SuperGLUE	Winogrande (XL)
T5-Base	84.3	85.5	75.1	66.6
Switch-Base	**86.7**	**87.2**	**79.5**	**73.3**
T5-Large	87.8	88.1	82.7	79.1
Switch-Large	**88.5**	**88.6**	**84.7**	**83.0**

Model	XSum	ANLI (R3)	ARC Easy	ARC Chal.
T5-Base	18.7	51.8	56.7	**35.5**
Switch-Base	**20.3**	**54.0**	**61.3**	32.8
T5-Large	20.9	56.6	**68.8**	**35.5**
Switch-Large	**22.3**	**58.6**	66.0	**35.5**

Model	CB Web QA	CB Natural QA	CB Trivia QA
T5-Base	26.6	25.8	24.5
Switch-Base	**27.4**	**26.8**	**30.7**
T5-Large	27.7	27.6	29.5
Switch-Large	**31.3**	**29.5**	**36.9**

significant improvements across many tasks

Downstream: fine-tuning

Model	GLUE	SQuAD	SuperGLUE	Winogrande (XL)
T5-Base	84.3	85.5	75.1	66.6
Switch-Base	**86.7**	**87.2**	**79.5**	**73.3**
T5-Large	87.8	88.1	82.7	79.1
Switch-Large	**88.5**	**88.6**	**84.7**	**83.0**

summarize articles

Model	XSum	ANLI (R3)	ARC Easy	ARC Chal.
T5-Base	18.7	51.8	56.7	**35.5**
Switch-Base	**20.3**	**54.0**	**61.3**	32.8
T5-Large	20.9	56.6	**68.8**	**35.5**
Switch-Large	**22.3**	**58.6**	66.0	**35.5**

Model	CB Web QA	CB Natural QA	CB Trivia QA
T5-Base	26.6	25.8	24.5
Switch-Base	**27.4**	**26.8**	**30.7**
T5-Large	27.7	27.6	29.5
Switch-Large	**31.3**	**29.5**	**36.9**

significant improvements across many tasks

Downstream: fine-tuning

question answering

Model	GLUE	SQuAD	SuperGLUE	Winogrande (XL)
T5-Base	84.3	85.5	75.1	66.6
Switch-Base	**86.7**	**87.2**	**79.5**	**73.3**
T5-Large	87.8	88.1	82.7	79.1
Switch-Large	**88.5**	**88.6**	**84.7**	**83.0**

Model	XSum	ANLI (R3)	ARC Easy	ARC Chal.
T5-Base	18.7	51.8	56.7	**35.5**
Switch-Base	**20.3**	**54.0**	**61.3**	32.8
T5-Large	20.9	56.6	**68.8**	**35.5**
Switch-Large	**22.3**	**58.6**	66.0	**35.5**

Model	CB Web QA	CB Natural QA	CB Trivia QA
T5-Base	26.6	25.8	24.5
Switch-Base	**27.4**	**26.8**	**30.7**
T5-Large	27.7	27.6	29.5
Switch-Large	**31.3**	**29.5**	**36.9**

significant improvements across many tasks

Downstream: fine-tuning

common sense reasoning

Model	GLUE	SQuAD	SuperGLUE	Winogrande (XL)
T5-Base	84.3	85.5	75.1	66.6
Switch-Base	**86.7**	**87.2**	**79.5**	**73.3**
T5-Large	87.8	88.1	82.7	79.1
Switch-Large	**88.5**	**88.6**	**84.7**	**83.0**

Natural language inference

Model	XSum	ANLI (R3)	ARC Easy	ARC Chal.
T5-Base	18.7	51.8	56.7	**35.5**
Switch-Base	**20.3**	**54.0**	**61.3**	32.8
T5-Large	20.9	56.6	**68.8**	**35.5**
Switch-Large	**22.3**	**58.6**	66.0	**35.5**

Model	CB Web QA	CB Natural QA	CB Trivia QA
T5-Base	26.6	25.8	24.5
Switch-Base	**27.4**	**26.8**	**30.7**
T5-Large	27.7	27.6	29.5
Switch-Large	**31.3**	**29.5**	**36.9**

significant improvements across many tasks

Downstream: fine-tuning

Model	GLUE	SQuAD	SuperGLUE	Winogrande (XL)
T5-Base	84.3	85.5	75.1	66.6
Switch-Base	**86.7**	**87.2**	**79.5**	**73.3**
T5-Large	87.8	88.1	82.7	79.1
Switch-Large	**88.5**	**88.6**	**84.7**	**83.0**

Model	XSum	ANLI (R3)	ARC Easy	ARC Chal.
T5-Base	18.7	51.8	56.7	**35.5**
Switch-Base	**20.3**	**54.0**	**61.3**	32.8
T5-Large	20.9	56.6	**68.8**	**35.5**
Switch-Large	**22.3**	**58.6**	66.0	**35.5**

Model	CB Web QA	CB Natural QA	CB Trivia QA
T5-Base	26.6	25.8	24.5
Switch-Base	**27.4**	**26.8**	**30.7**
T5-Large	27.7	27.6	29.5
Switch-Large	**31.3**	**29.5**	**36.9**

closed book QA

significant improvements across many tasks

Downstream Results

- Fine-tuning
- **Distillation**
- Multilingual Learning

Knowledge Distillation in a nutshell

Distill knowledge from teacher model to student model.
A popular technique for model compression.

$$\mathcal{L}_{dist}(\boldsymbol{w}) = -\sum_{i=1}^{n}\sum_{j=1}^{k}[\mathrm{softmax}\big(f_{\boldsymbol{w}_t}(\boldsymbol{x})/T\big)]_j \log p(y = j|\boldsymbol{x}_i;\boldsymbol{w})$$

$$\mathcal{L}(\boldsymbol{w}) = \alpha\mathcal{L}_{cce}(\boldsymbol{w}) + (1-\alpha)\mathcal{L}_{dist}(\boldsymbol{w})$$

Distillation techniques

Quality: Neg Log Perplexity

Technique	Parameters	Quality (↑)
T5-Base **student**	223M	-1.636
Switch-Base **teacher**	3,800M	-1.444
Distillation	223M	(3%) -1.631
+ Init. non-expert weights from teacher	223M	(20%) -1.598
+ 0.75 mix of hard and soft loss	223M	(29%) -1.580
Initialization Baseline (no distillation)		
Init. non-expert weights from teacher	223M	-1.639

29% quality gain with only 1/20th of the parameters.

Distillation compression rates

	Dense	Sparse				
Parameters	223M	1.1B	2.0B	3.8B	7.4B	14.7B
Pre-trained Neg. Log Perp. (↑)	-1.636	-1.505	-1.474	-1.444	-1.432	-1.427
Distilled Neg. Log Perp. (↑)	—	-1.587	-1.585	-1.579	-1.582	-1.578
Percent of Teacher Performance	—	37%	32%	30 %	27 %	28 %
Compression Percent	—	82 %	90 %	95 %	97 %	99 %

compress the model by 99% and maintain 28% of the teacher quality improvements

Distilling fine-tuned SuperGLUE model

Model	Parameters	FLOPS	SuperGLUE (↑)
T5-Base	223M	124B	74.6
Switch-Base	7410M	124B	81.3
Distilled T5-Base	223M	124B	(30%) 76.6

Sparse teacher can be an effective teacher on small dataset

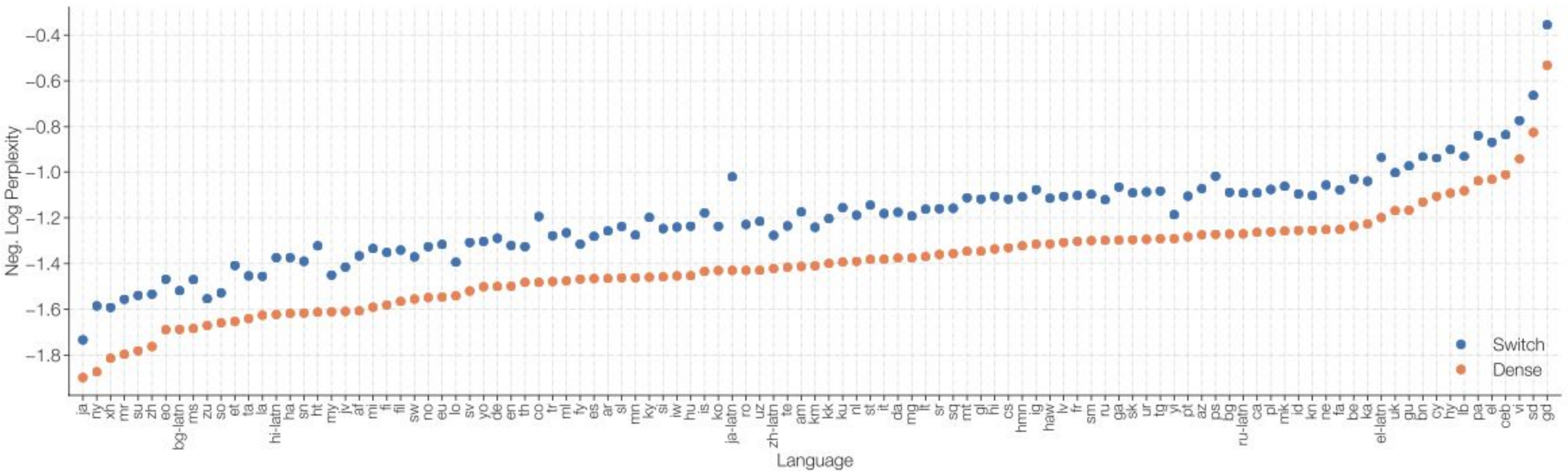
Downstream Results

- Fine-tuning
- Distillation
- **Multilingual Learning**

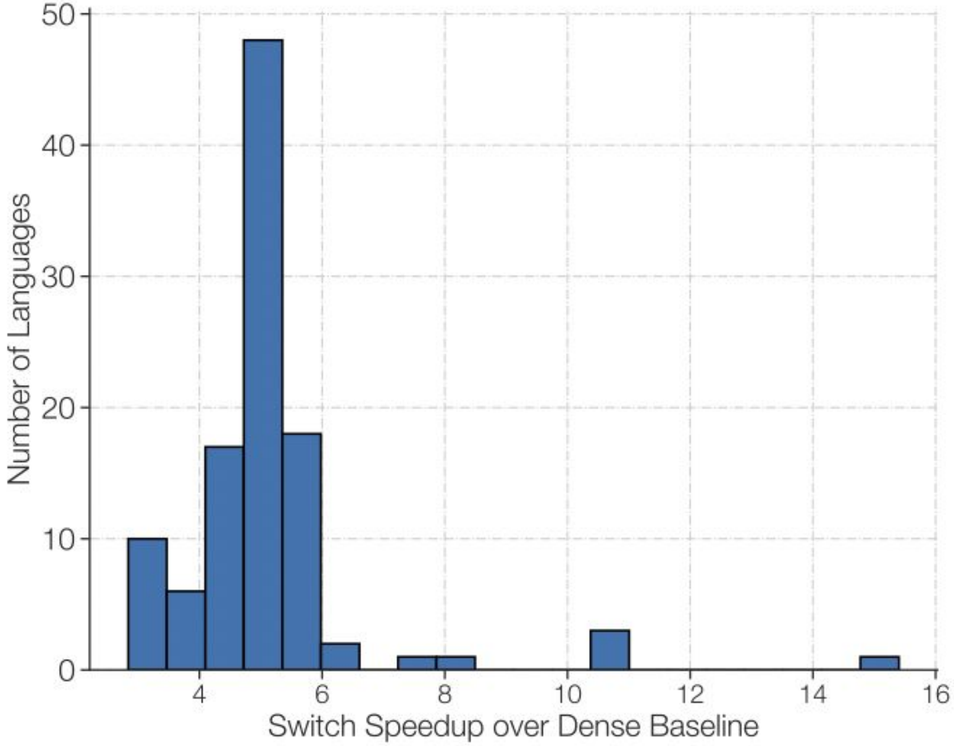
Multilingual Learning

Pre-training on 101 languages

Comparison of FLOP-matched Switch model (mSwitch-Base) to T5 base (mT5-Base)



Histogram of speedup on 101 languages



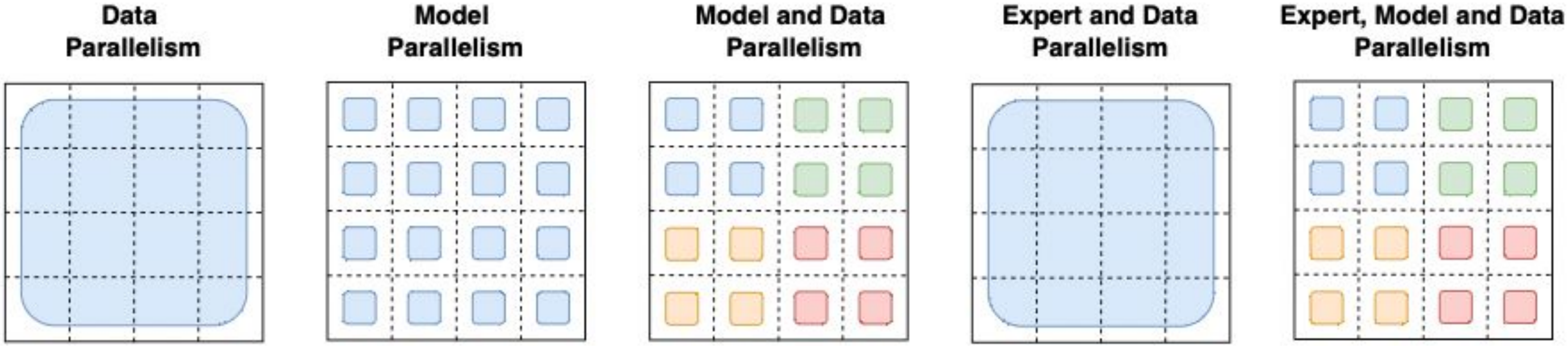
- **5x avg. per step speedup over baseline**
- **> 4x speedup for 91% languages**

Baseline: mT5-Base

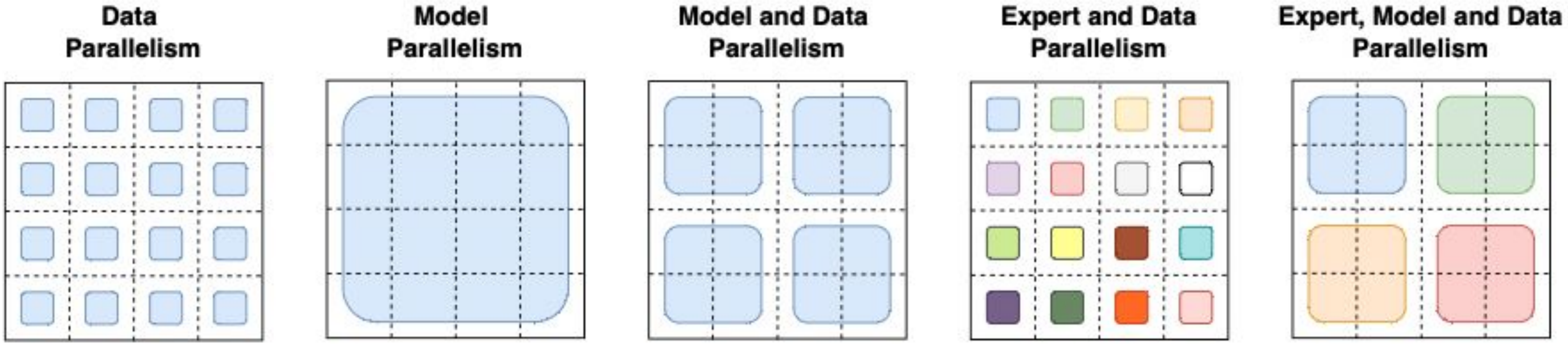
Implementation Discussion

Data Parallelism, Model Parallelism and Expert Parallelism

How the *data* is split over cores



How the *model weights* are split over cores



Switch model design and pre-training performance

Model	Parameters	FLOPs/seq	d_{model}	FFN_{GEGLU}	d_{ff}	d_{kv}	Num. Heads
T5-Base	0.2B	124B	768	✓	2048	64	12
T5-Large	0.7B	425B	1024	✓	2816	64	16
T5-XXL	11B	6.3T	4096	✓	10240	64	64
Switch-Base	7B	124B	768	✓	2048	64	12
Switch-Large	26B	425B	1024	✓	2816	64	16
Switch-XXL	395B	6.3T	4096	✓	10240	64	64
Switch-C	1571B	890B	2080		6144	64	32

Model	Expert Freq.	Num. Experts	Num Layers	Neg. Log Perp. @250k	Neg. Log Perp. @ 500k
T5-Base	–	12	–	-1.599	-1.556
T5-Large	–	24	–	-1.402	-1.350
T5-XXL	–	24	–	-1.147	-1.095
Switch-Base	1/2	12	128	-1.370	-1.306
Switch-Large	1/2	24	128	-1.248	-1.177
Switch-XXL	1/2	24	64	**-1.086**	**-1.008**
Switch-C	1	15	2048	-1.096	-1.043

Pre-lecture Questions

Pre-Lecture Question 1

How are sparse models different from the dense models by design? What is the biggest insight of Switch Transformers compared to previous Mixture-of-Experts models (Shazeer et al 2017)?

Sparse models generally refer to those with only a subset of the parameters of dense model.

In the context of sparse expert model, a set of parameters are partitioned into "experts" with unique weights. Unlike dense model where the entire network is used for each input, in sparse expert models, only a fraction of the experts/parameters are used for each example.

Switch Transformer routes a token to only a single expert rather than multiple experts, which was proposed in Shazeer 2017.

Pre-Lecture Question 2

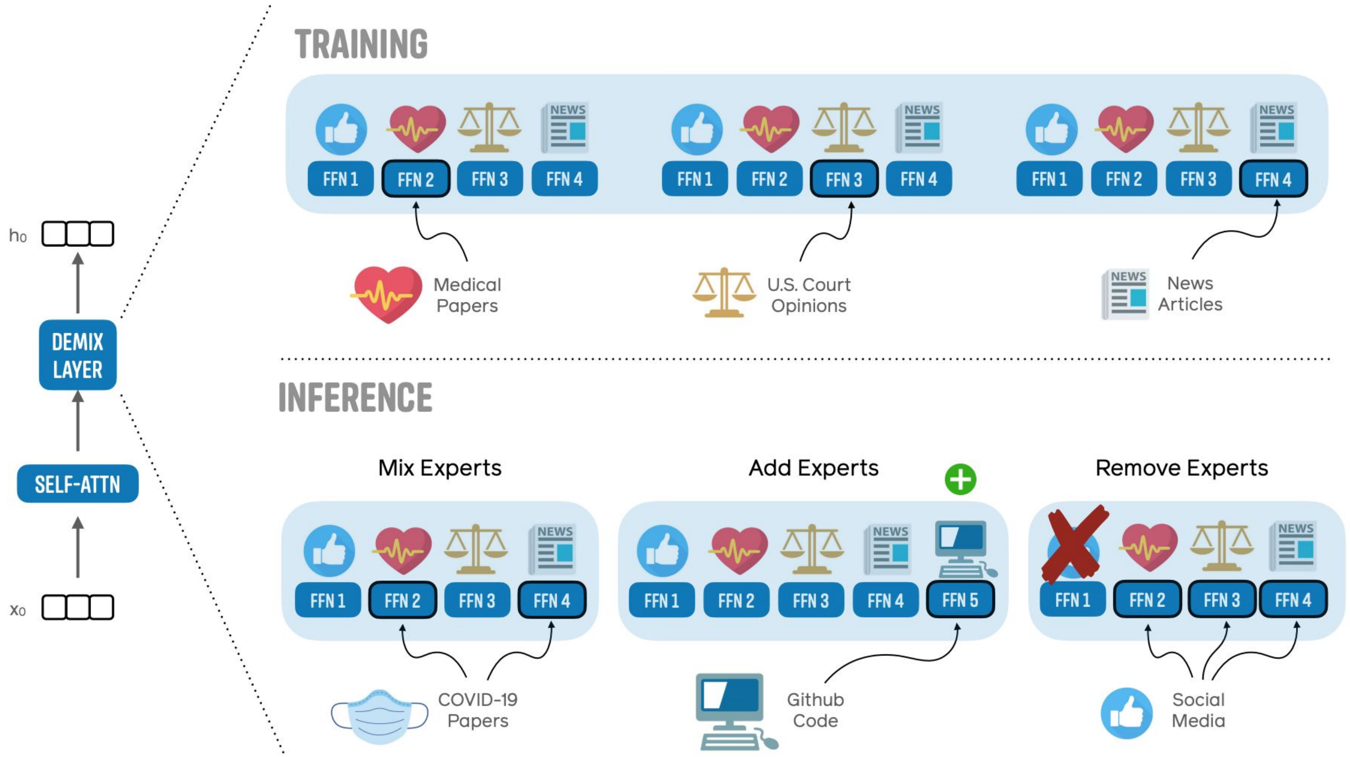
How to fine-tune sparse models for downstream tasks? What issue may arise in fine-tuning sparse models and what is the fix in Switch Transformers?

Switch Transformers have significantly more parameters than the FLOP-matched dense baseline, and therefore can be more prone to overfitting on downstream tasks with very few examples.

To fine-tune sparse models, the authors increase the dropout rate at the feedforward stage for each expert.

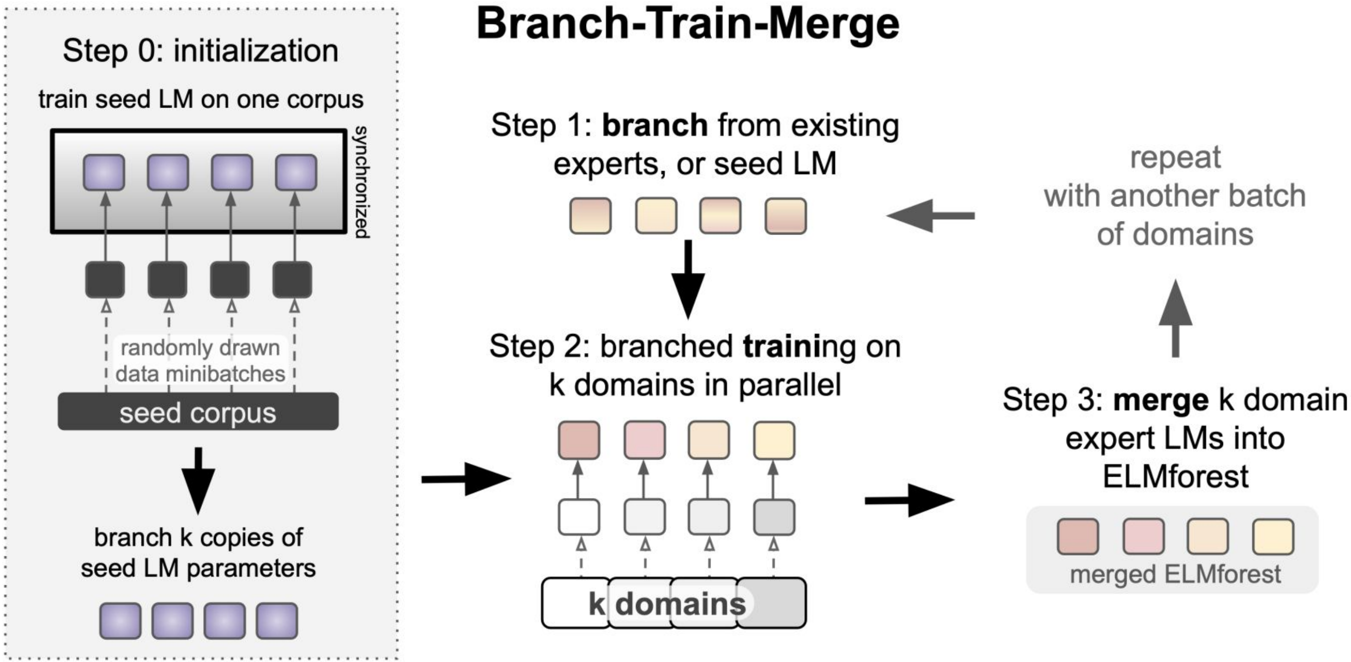
Follow up work

Follow Up Work - Domain Expert Mixture



Gururangan, Suchin, et al. "Disentangling domains for modular language modeling", 2021.

Follow Up Work - Parallel Training of Experts



Li, Margaret, et al. “Branch-train-merge: Embarrassingly parallel training of expert language models”, 2022.

Pre-Lecture Question 3

If we continue scaling up LLMs, sparse vs dense models - which one do you think is more promising? Can you discuss their pros and cons (computation, storage and different use cases e.g., fine-tuning, prompting, in-context learning)?

Thank You!