LLM Optimization

Jordan Boyd-Graber

University of Maryland

Mixture of Experts

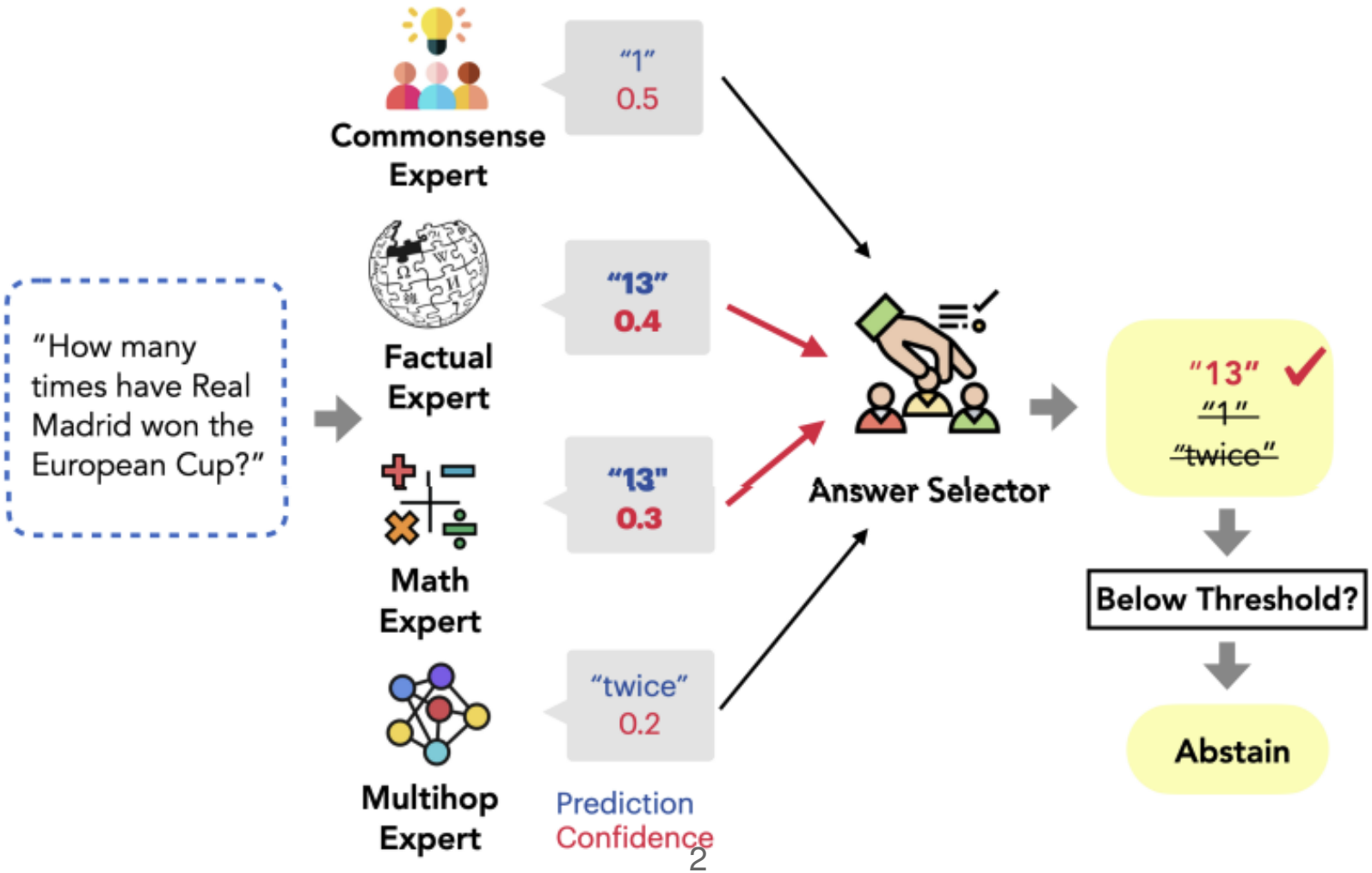
Slides adapted from William Fedus and Barret Zoph

Motivation

- The trend is to bigger models
- But what if you can do as well with N different models?

Motivation

- The trend is to bigger models
- But what if you can do as well with N different models?
- Problem is, how to know which model to use?



Motivation

- The trend is to bigger models
- But what if you can do as well with N different models?
- Problem is, how to know which model to use?

Getting MoRE out of Mixture of Language Model Reasoning Experts

Chenglei Si[1,4] **Weijia Shi**[2] **Chen Zhao**[3]
Luke Zettlemoyer[2] **Jordan Boyd-Graber**[1]
[1] University of Maryland [2] University of Washington
[3] NYU Shanghai [4] Stanford University
clsi@stanford.edu

Why is it a good idea?

- If you need a model to do x, easier to train that model than to train a massive model to do everything and x
- Users find it more interpretable
- Disagreements between models can help with calibration
- Cheaper to serve (smaller models can live on modest, smaller servers)

Why is it a good idea?

- If you need a model to do x, easier to train that model than to train a massive model to do everything and x
- Users find it more interpretable
- Disagreements between models can help with calibration
- Cheaper to serve (smaller models can live on modest, smaller servers)

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

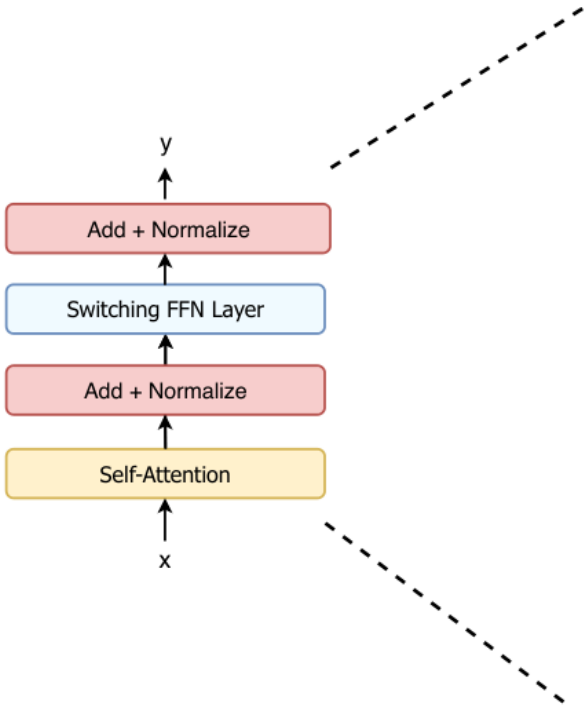
William Fedus*
LIAMFEDUS@GOOGLE.COM

Barret Zoph*
BARRETZOPH@GOOGLE.COM

Noam Shazeer
NOAM@GOOGLE.COM
Google, Mountain View, CA 94043, USA

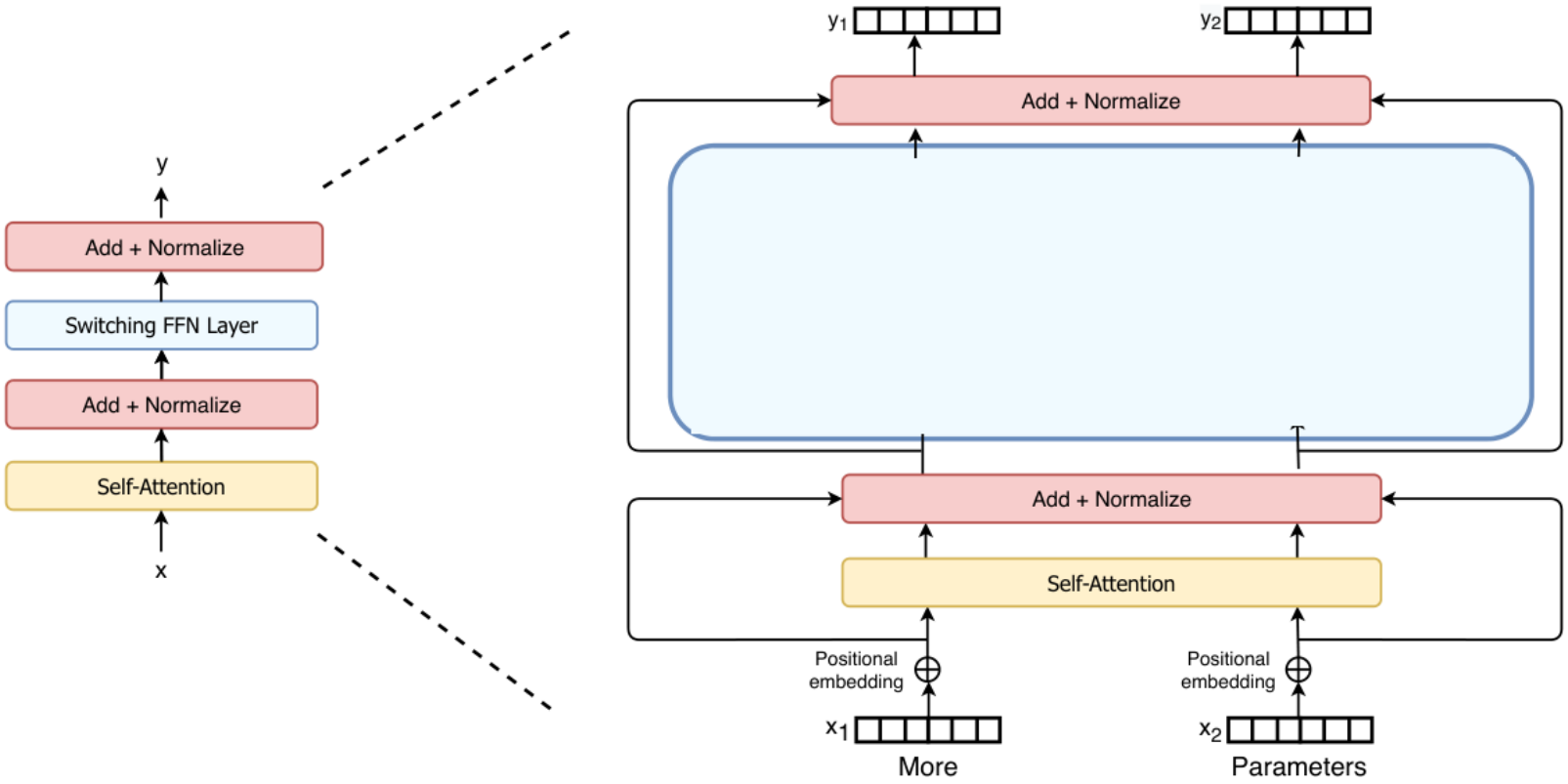
Editor: Alexander Clark

Going to the Token Level



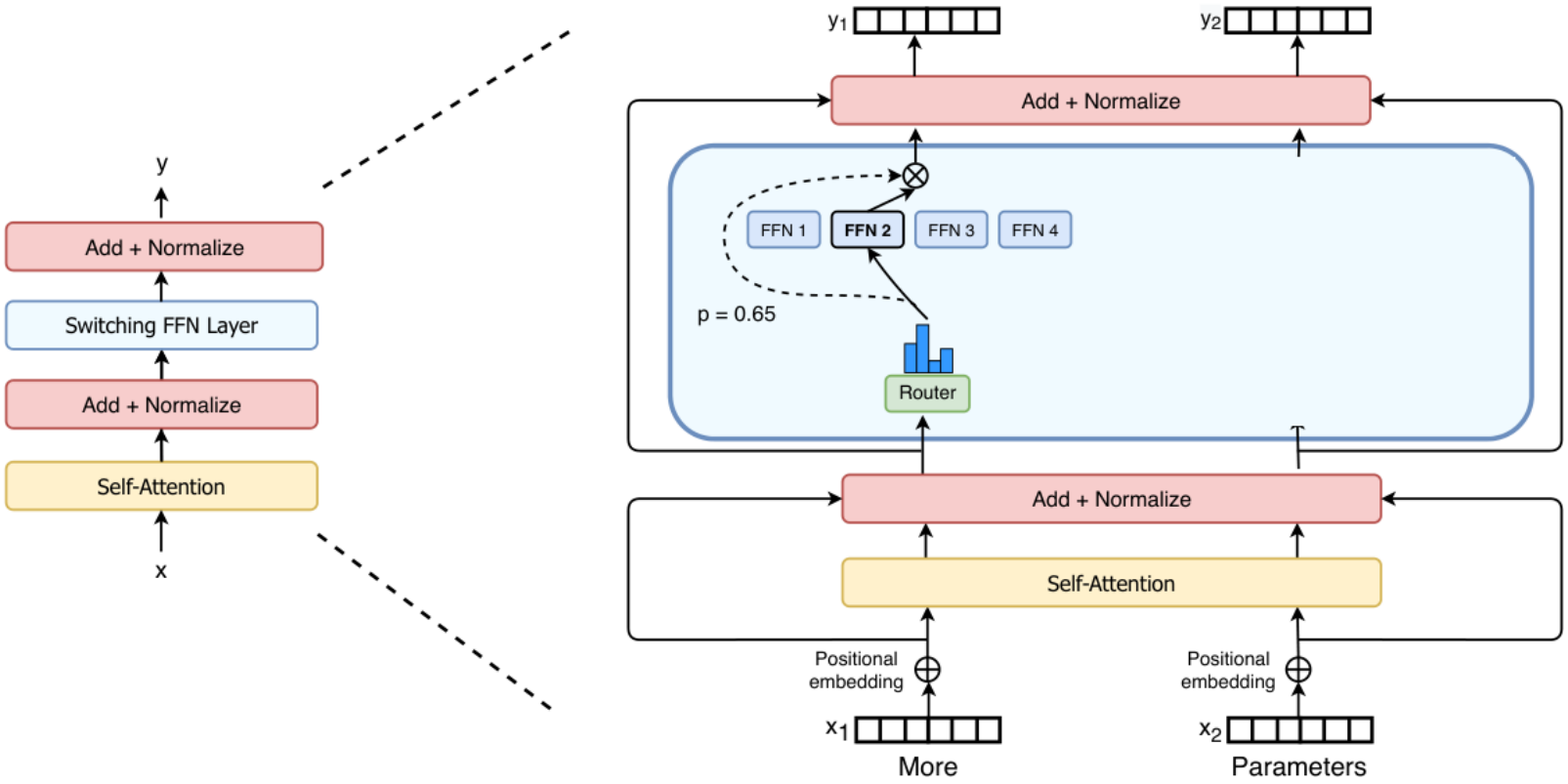
1.0

Going to the Token Level



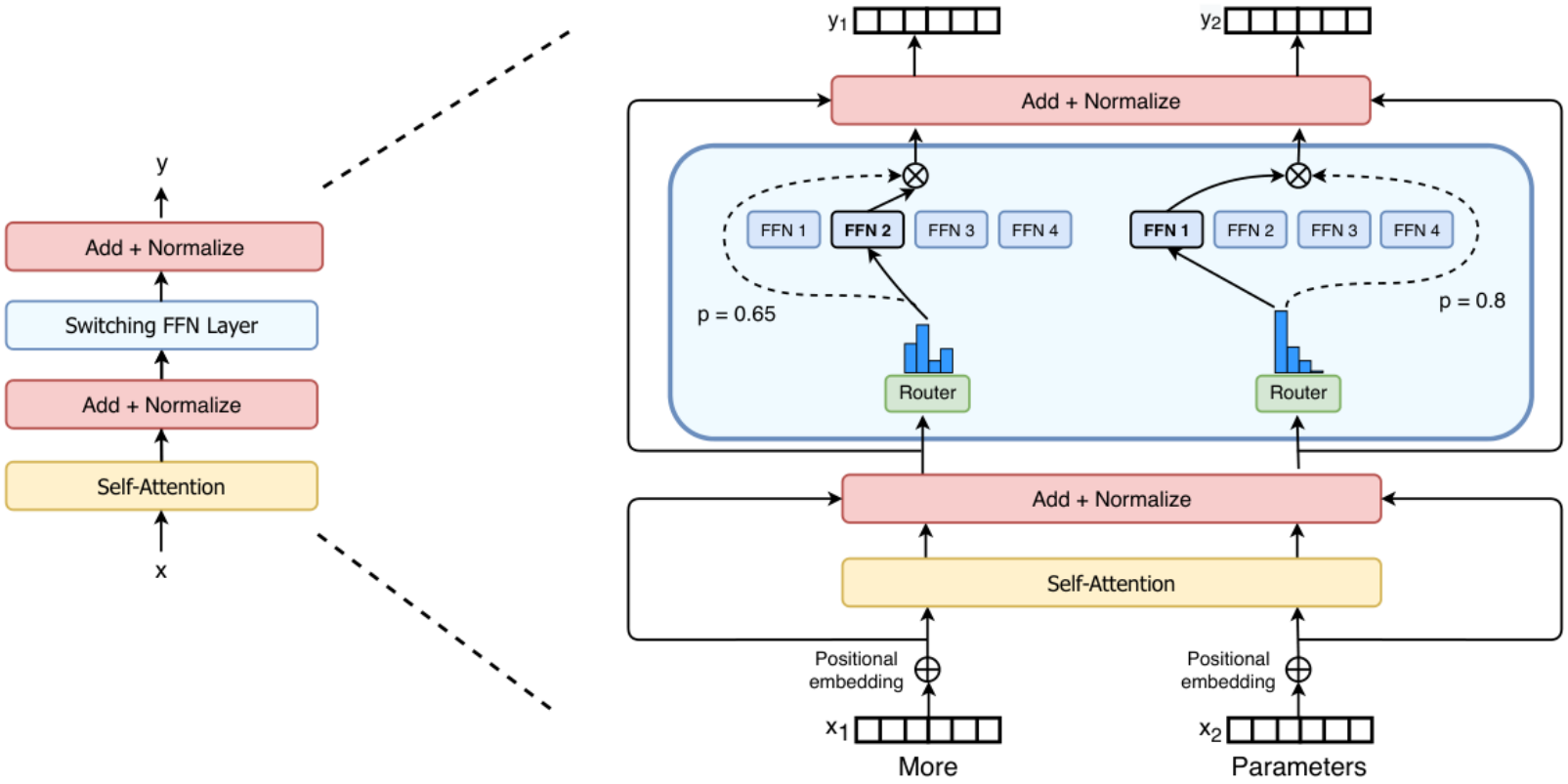
1.0

Going to the Token Level



1.0

Going to the Token Level



1.0

Routing

The router variable W_r produces logits $h(x) = W_r \cdot x$ which are normalized via a softmax distribution:

$$p_i(x) = \frac{e^{h(x)_i}}{\sum_j^N e^{h(x)_j}}. \tag{1}$$

The top-k gate values are selected for routing the token x to linearly weight each expert's contribution

$$y = \sum_{i \in \mathcal{T}} p_i(x) E_i(x). \tag{2}$$

But how do you train this?

- Just use same objective?

But how do you train this?

- Just use same objective?
- Shazeer et al. (2018) and Leipkhin et al. (2020): auxiliary loss
- Given N experts indexed by $i = 1$ to N and a batch $\mathscr{B}$ with T tokens

$$\text{loss} = \alpha \cdot N \cdot \sum_{i=1}^{N} f_i \cdot P_i \tag{3}$$

where f_i is the fraction of tokens dispatched to expert i,

But how do you train this?

- Just use same objective?
- Shazeer et al. (2018) and Leipkhin et al. (2020): auxiliary loss
- Given N experts indexed by $i = 1$ to N and a batch $\mathcal{B}$ with T tokens

$$\text{loss} = \alpha \cdot N \cdot \sum_{i=1}^{N} f_i \cdot P_i \tag{3}$$

where f_i is the fraction of tokens dispatched to expert i,

$$f_i = \frac{1}{T} \sum_{x \in \mathcal{B}} \mathbb{1}\{\text{argmax } p(x) = i\} \tag{4}$$

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

But how do you train this?

- Just use same objective?
- Shazeer et al. (2018) and Leipkhin et al. (2020): auxiliary loss
- Given N experts indexed by $i = 1$ to N and a batch $\mathcal{B}$ with T tokens

$$\text{loss} = \alpha \cdot N \cdot \sum_{i=1}^{N} f_i \cdot P_i \tag{3}$$

where f_i is the fraction of tokens dispatched to expert i,

$$f_i = \frac{1}{T} \sum_{x \in \mathcal{B}} \mathbb{1}\{\text{argmax } p(x) = i\} \tag{4}$$

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

$$P_i = \frac{1}{T} \sum_{x \in \mathcal{B}} p_i(x). \tag{5}$$

Since we seek uniform routing of the batch of tokens across the N experts, we desire both vectors to have values of $1/N$.

Why can this help?

- Easier fine tuning (more on this later)
- Increase in overall scale
- Easier to distribute to more machines (Experts on different devices)

Wrapup

- Most major models are likely using mixture of experts
- It helps keep training scalable
- And allows to claim larger models
- It also leads to better results
- Helps address sparsity issue: most of models aren't useful for any particular example