Week 3: Large Language Models and In-context Learning

Generative AI
Saarland University – Winter Semester 2024/25

Adish Singla
genai-w24-tutors@mpi-sws.org

MAX PLANCK INSTITUTE
FOR SOFTWARE SYSTEMS

MAX-PLANCK-GESELLSCHAFT

Outline of the Lecture

- Organizational updates
- Week 2 recap and assignment
- From Simple Ff Neural LM to Transformer-based LM
  - Single attention head
  - Multiple attention heads
  - Transformer block
  - Transformer architecture
- GPT-1/2/3 models and in-context learning capabilities
- Week 3 assignment

Outline of the Lecture

- Organizational updates
- Week 2 recap and assignment
- From Simple Ff Neural LM to Transformer-based LM
  - Single attention head
  - Multiple attention heads
  - Transformer block
  - Transformer architecture
- GPT-1/2/3 models and in-context learning capabilities
- Week 3 assignment

Organizational Updates

Assignments

- Deadlines are strict and enforced based on submission timesteps
- You should verify your submitted files from the personal link
- In case you are on sick leave:
  - Request an extension before the deadline
  - Send us a sick note from the doctor

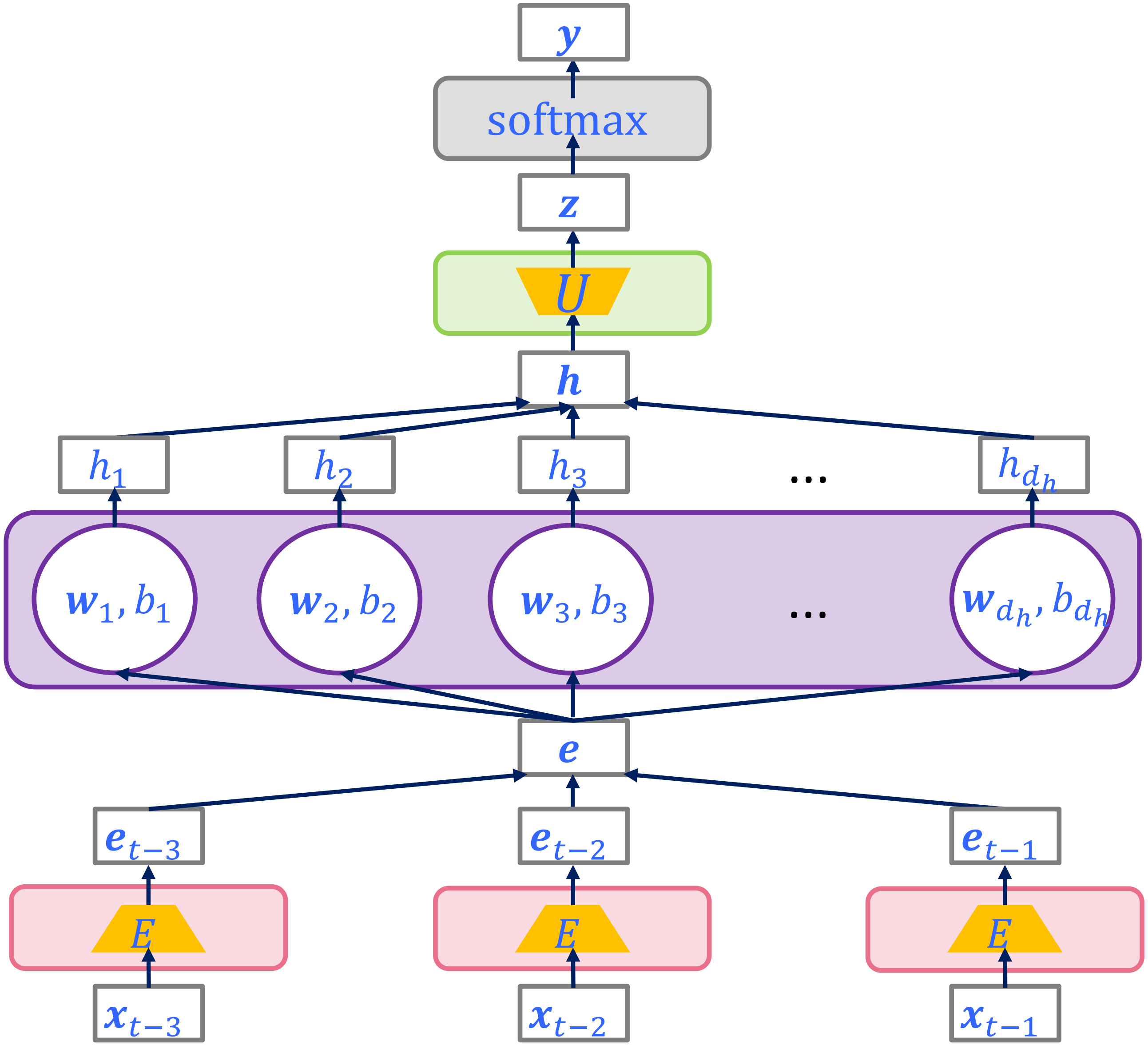
Mailing lists and communication

- genai-w24-tutors@mpi-sws.org
  - Use this to contact tutors; CC this with any tutor for course-related discussions
  - Send email from your "official" email id
- genai-w24-announcements@sympa.mpi-sws.org
  - Tutors will use this to send course-related announcements
- There is no mailing list for group discussions
  - Students are welcome to clarify their questions with us via email or during office hours
  - For questions and clarifications of general interest, we will let everyone know

Outline of the Lecture

- Organizational updates
- Week 2 recap and assignment
- From Simple Ff Neural LM to Transformer-based LM
  - Single attention head
  - Multiple attention heads
  - Transformer block
  - Transformer architecture
- GPT-1/2/3 models and in-context learning capabilities
- Week 3 assignment

Simple Feedforward Neural LM: Detailed Architecture



probability vector, $|V|$

softmax operation

score vector (logits), $|V|$

unembedding matrix, $|V| \times d_h$

joint hidden layer outputs, d_h

d_h hidden layer outputs, 1

d_h neural units:
- each weight vector, $3d$
- each bias term, 1

joint embedding vector, $3d$

3 embedding vectors, d

embedding matrix, $d \times |V|$

3 one-hot vectors, $|V|$

Reference: [Bengio et al., NeurIPS'00; JMLR'03]

Week 2 Assignment: E.1

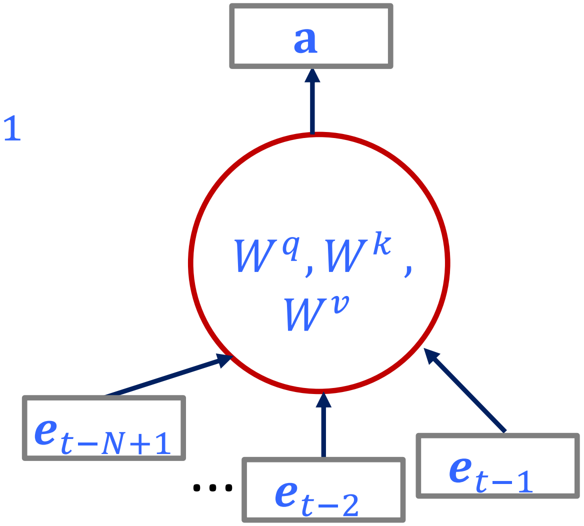
E.1 Steps to find number of parameters for general N

- Parameters for embedding matrix E: $|V|d$
- Parameters for d_h neural units:
  - each weight vector: $(N-1)d$
  - each bias term: 1
- Parameters for unembedding matrix U: $|V|d_h$

- The number of parameters for this architecture is

$$|V|(d+d_h)+\big((N-1)d+1\big)d_h$$

Attention Mechanism with Single Head: Actual Version



Input vectors

- $N-1$ vectors of size d denoted by $\boldsymbol{e}_{t-N+1}, \ldots, \boldsymbol{e}_{t-2}, \boldsymbol{e}_{t-1}$

Output vector

- 1 vector of size d_v denoted by $\mathbf{a}$

Parameters

- query matrix W^q of size $d_k \times d$
- key matrix W^k of size $d_k \times d$
- value matrix W^v of size $d_v \times d$

Computation of vector $\mathbf{a}$

- Given scalar values α_i, attention vector is $\mathbf{a} = \sum_{i=t-N+1}^{t-1} \alpha_i W^v \boldsymbol{e}_i$
  - Compute similarity scores for $\boldsymbol{e}_{t-1}$ with $\boldsymbol{e}_i$ for $i = t-N+1, \ldots, t-2, t-1$
    using vector dot prodct similarity $\mathrm{score}(\boldsymbol{e}_{t-1}, \boldsymbol{e}_i) = \frac{W^q \boldsymbol{e}_{t-1} \cdot W^k \boldsymbol{e}_i}{\mathrm{sqrt}(d_k)}$
  - Compute α_i using softmax over scores: $\alpha_i = \mathrm{softmax}(\mathrm{score}(\boldsymbol{e}_{t-1}, \boldsymbol{e}_i))$

Week 2 Assignment: E.2 and E.3

E.2 Steps to compute the attention vector

- Steps are mentioned in the previous slide

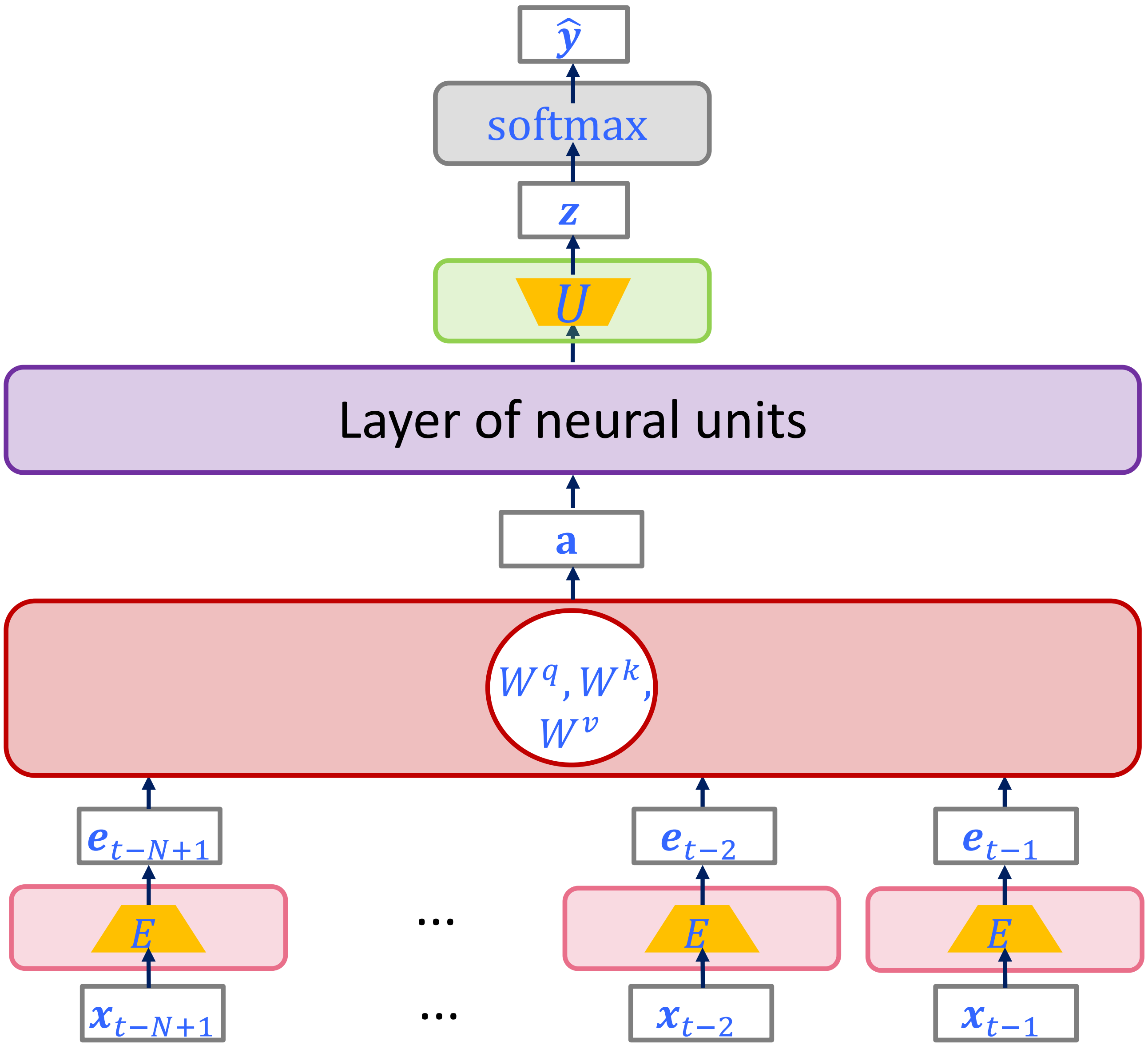
E.3 Steps to find number of parameters in one attention head

- Parameters for query matrix W^q: $d_k d$
- Parameters for key matrix W^k: $d_k d$
- Parameters for value matrix W^v: $d_v d$

- The number of parameters in one attention head is

$$2d_k d + d_v d$$

Incorporating Attention Mechanism in Simple Ff Neural LM



probability vector, $|V|$

softmax operation

score vector (logits), $|V|$

unembedding matrix, $|V| \times d_h$

d_h neural units:
- each weight vector, d_v
- each bias value, 1

attention vector, d_v

1 attention head:
- query matrix, $d_k \times d$
- key matrix, $d_k \times d$
- value matrix, $d_v \times d$

$N-1$ embedding vectors, d

embedding matrix, $d \times |V|$

$N-1$ one-hot vectors, $|V|$

Week 2 Assignment: E.4

E.4 Steps to find number of parameters in the architecture

- Parameters for embedding matrix E: $|V|d$
- Parameters for one attention head: $2d_k d + d_v d$
- Parameters for d_h neural units:
  - each weight vector: d_v
  - each bias term: 1
- Parameters for unembedding matrix U: $|V|d_h$

- The number of parameters for this architecture is

$$|V|(d + d_h) + (2d_k d + d_v d) + (d_v + 1)d_h$$

Week 2 Assignment: I.1 and I.2

I.2 Number of parameters

	$Layers = 2$	$Layers = 4$
$N = 3$	9,721,596	14,980,860
$N = 5$	9,721,596	14,980,860

- Number of parameters do not depend on N
- Number of parameters increased with number of layers

I.1 Perplexity

	$Layers = 2$	$Layers = 4$
$N = 3$	8.32	8.15
$N = 5$	10.63	6.65

- Fix $Layers$: Simply increasing N without enough parameters increased perplexity
- Fix N: Fix Increasing the number of layers helped in reducing perplexity

Outline of the Lecture

- Organizational updates
- Week 2 recap and assignment
- From Simple Ff Neural LM to Transformer-based LM
  - Single attention head
  - Multiple attention heads
  - Transformer block
  - Transformer architecture
- GPT-1/2/3 models and in-context learning capabilities
- Week 3 assignment

How to Deal with Large Contexts?

The water of Walden Pond is beautifully **…**

The chicken did not cross the road because it **…**

A woman is throwing a **…**

Acknowledgements: Chapter 9 of SLP book; [Xu et al., ICML'15]

By Attending to Relevant Parts and Integrating Information

The water of Walden Pond is beautifully **blue**

The chicken did not cross the road because it **was**

A woman is throwing a **frisbee**

By Attending and Integrating Different Types of Information

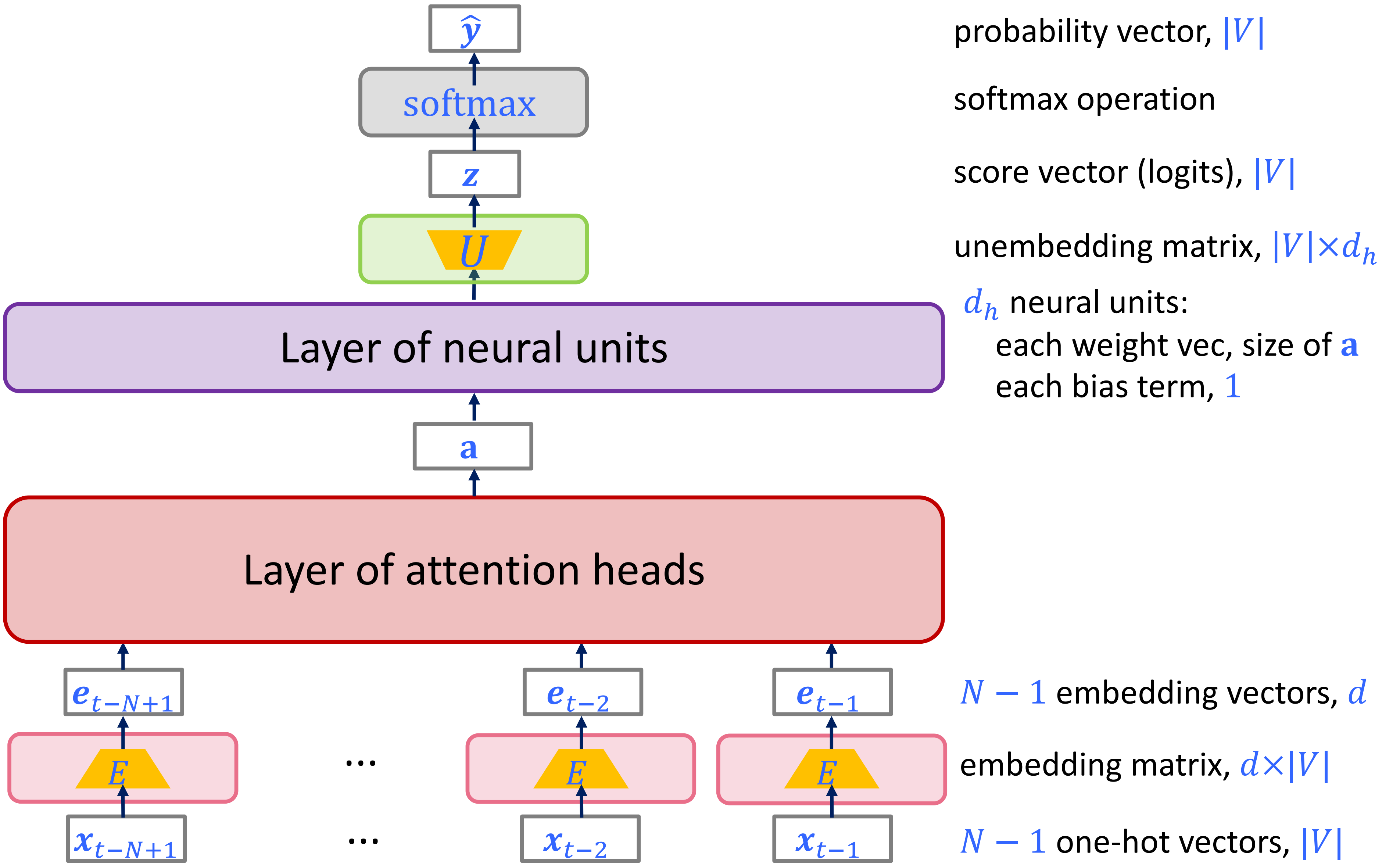
The water of Walden Pond is beautifully **blue**

The chicken did not cross the road because it **was**

A woman is throwing a **frisbee**

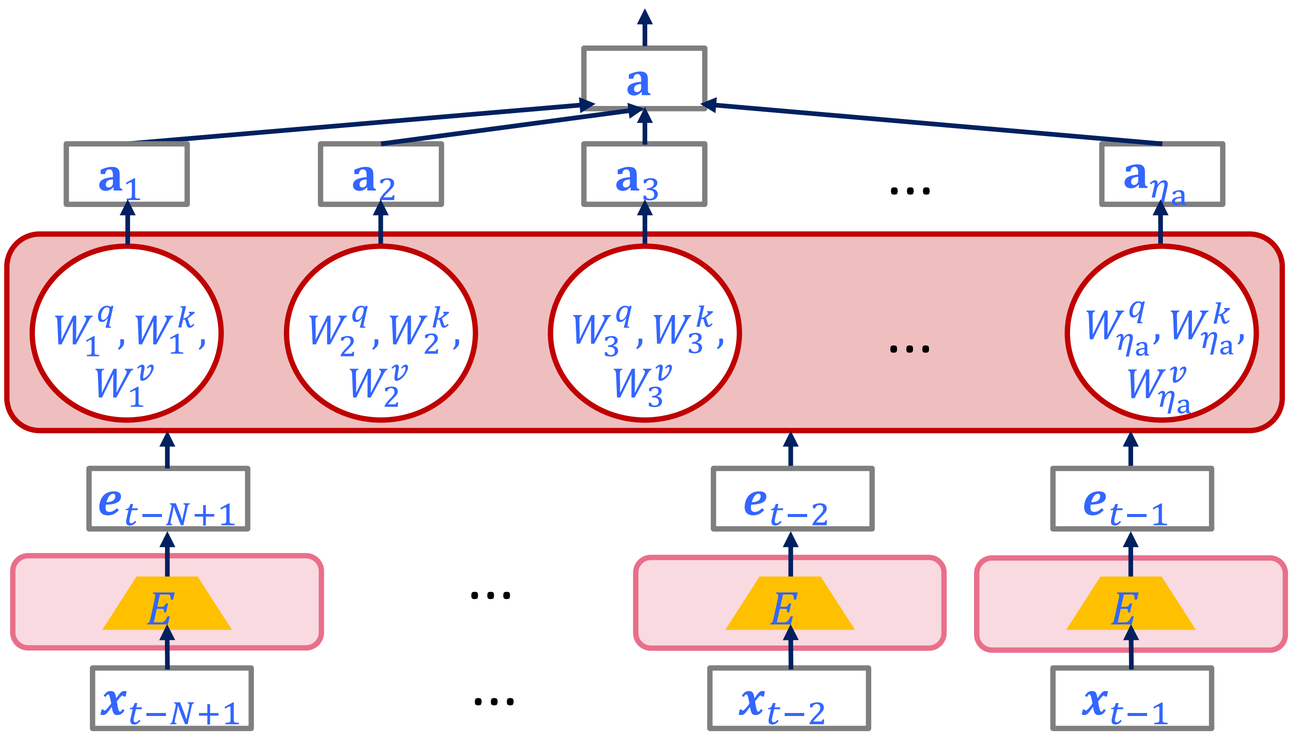
Acknowledgements: Chapter 9 of SLP book; [Xu et al., ICML'15]

Incorporating Multi-head Attention in Simple Ff Neural LM



Layer of Attention Heads: Computing Attention Vector

- η_a number of attention heads
- For each attention head indexed i:
  - compute single attention vector $\mathbf{a}_i$ of size d_v using W_i^q, W_i^k, W_i^v
- Concatenate single vectors to obtain attention vector $\mathbf{a}$ of size $\eta_a d_v$



attention vector, $\eta_a d_v$

η_a single attention vectors, d_v

η_a attention heads:

each query matrix, $d_k \times d$

each key matrix, $d_k \times d$

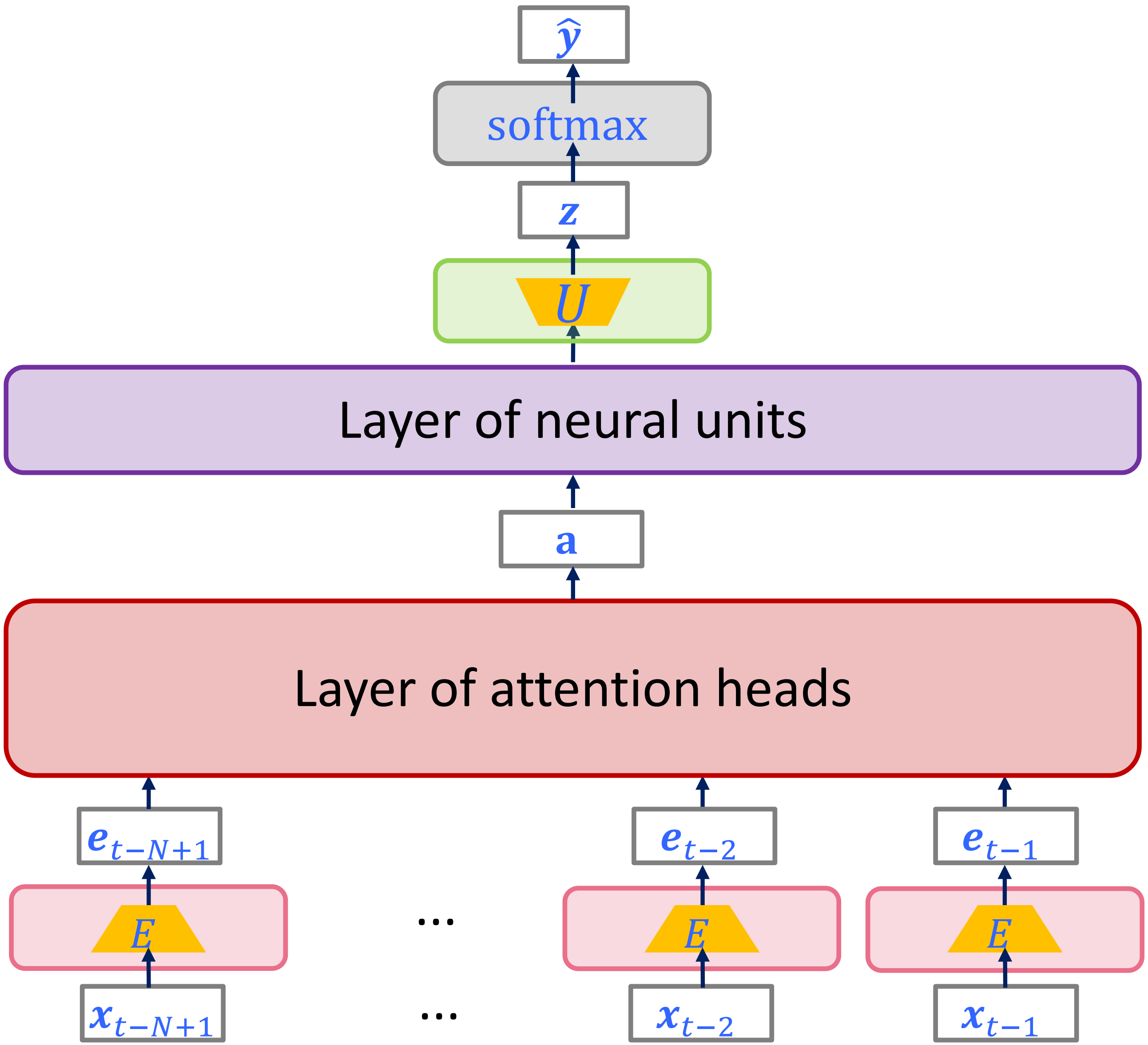
each value matrix, $d_v \times d$

$N-1$ embedding vectors, d

embedding matrix, $d \times |V|$

$N-1$ one-hot vectors, $|V|$

Multi-head Attention in Simple Ff Neural LM



probability vector, $|V|$

softmax operation

score vector (logits), $|V|$

unembedding matrix, $|V| \times d_h$

d_h neural units:
- each weight vector, $\eta_\mathrm{a} d_v$
- each bias value, 1

attention vector, $\eta_\mathrm{a} d_v$

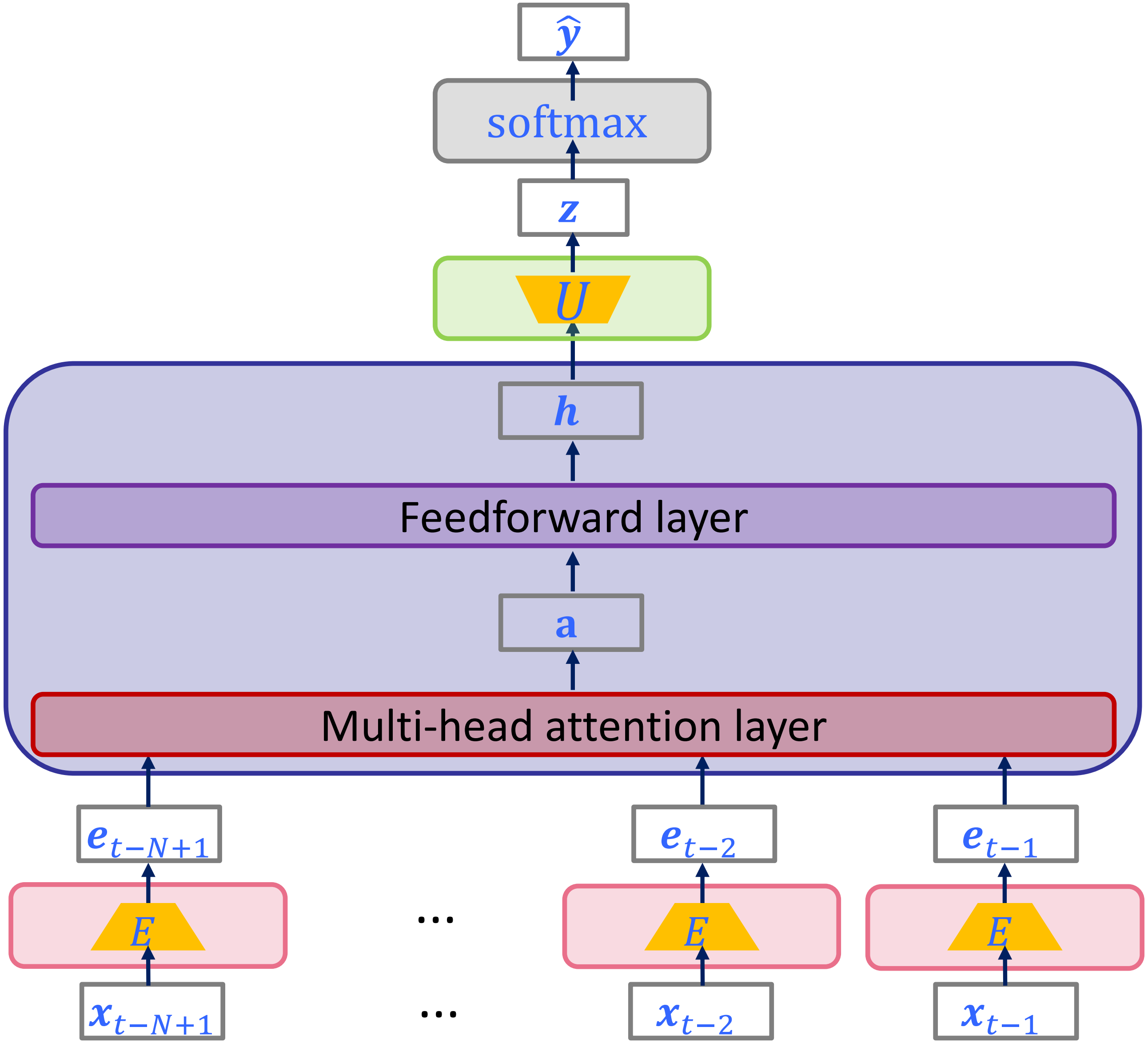
η_a attention heads:
- each query matrix, $d_k \times d$
- each key matrix, $d_k \times d$
- each value matrix, $d_v \times d$

$N-1$ embedding vectors, d

embedding matrix, $d \times |V|$

$N-1$ one-hot vectors, $|V|$

Multi-head Attention in Simple Ff Neural LM | Same as #19



probability vector, $|V|$

softmax operation

score vector (logits), $|V|$

unembedding matrix, $|V| \times d_h$

feedforward layer vector, d_h

d_h neural units

attention vector, $\eta_a d_v$

η_a attention heads

$N-1$ embedding vectors, d

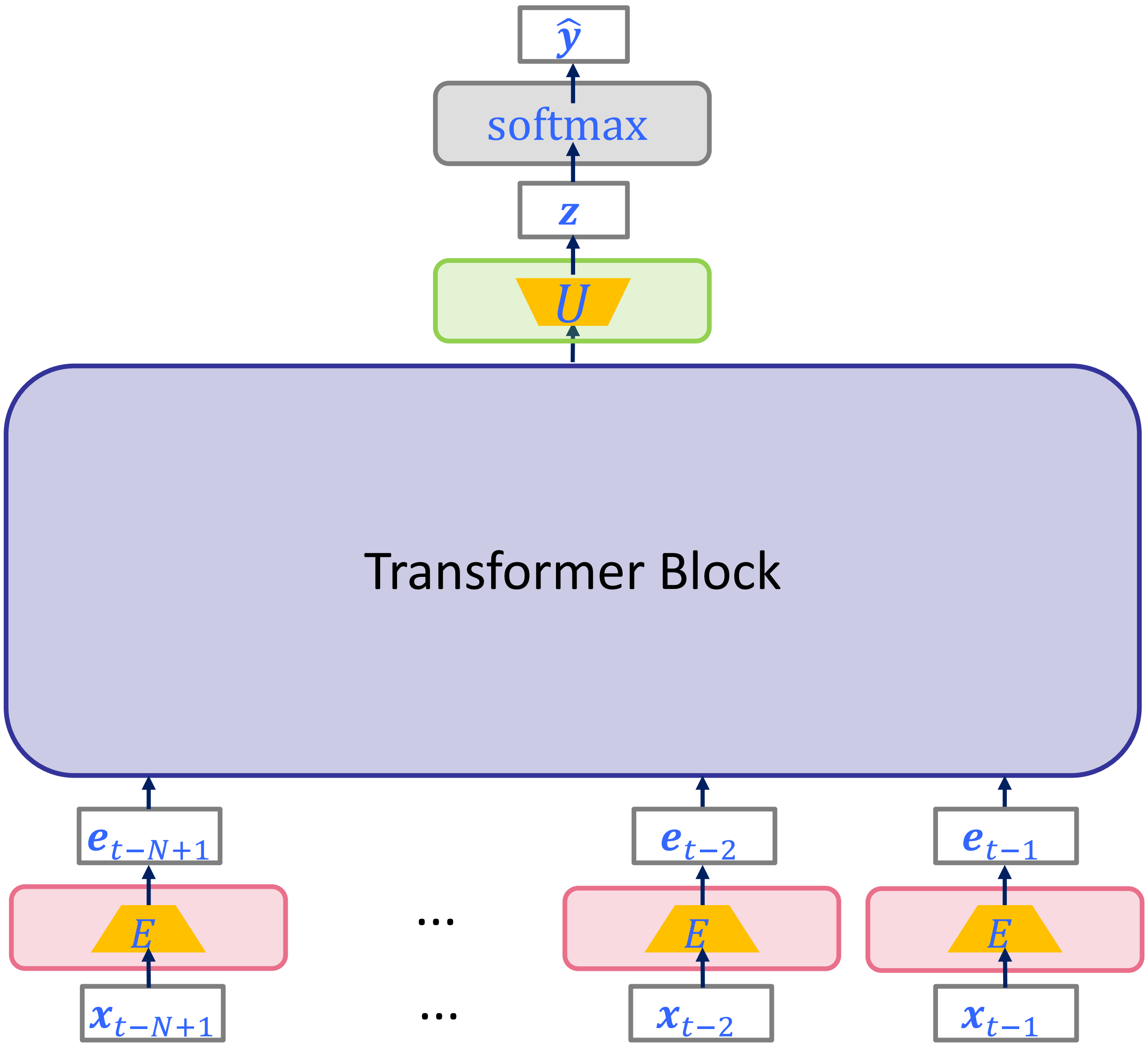
embedding matrix, $d \times |V|$

$N-1$ one-hot vectors, $|V|$

Outline of the Lecture

- Organizational updates
- Week 2 recap and assignment
- From Simple Ff Neural LM to Transformer-based LM
  - Single attention head
  - Multiple attention heads
  - Transformer block
  - Transformer architecture
- GPT-1/2/3 models and in-context learning capabilities
- Week 3 assignment

Transformer Block: Overview of Input-Output Interface



probability vector, $|V|$

softmax operation

score vector (logits), $|V|$

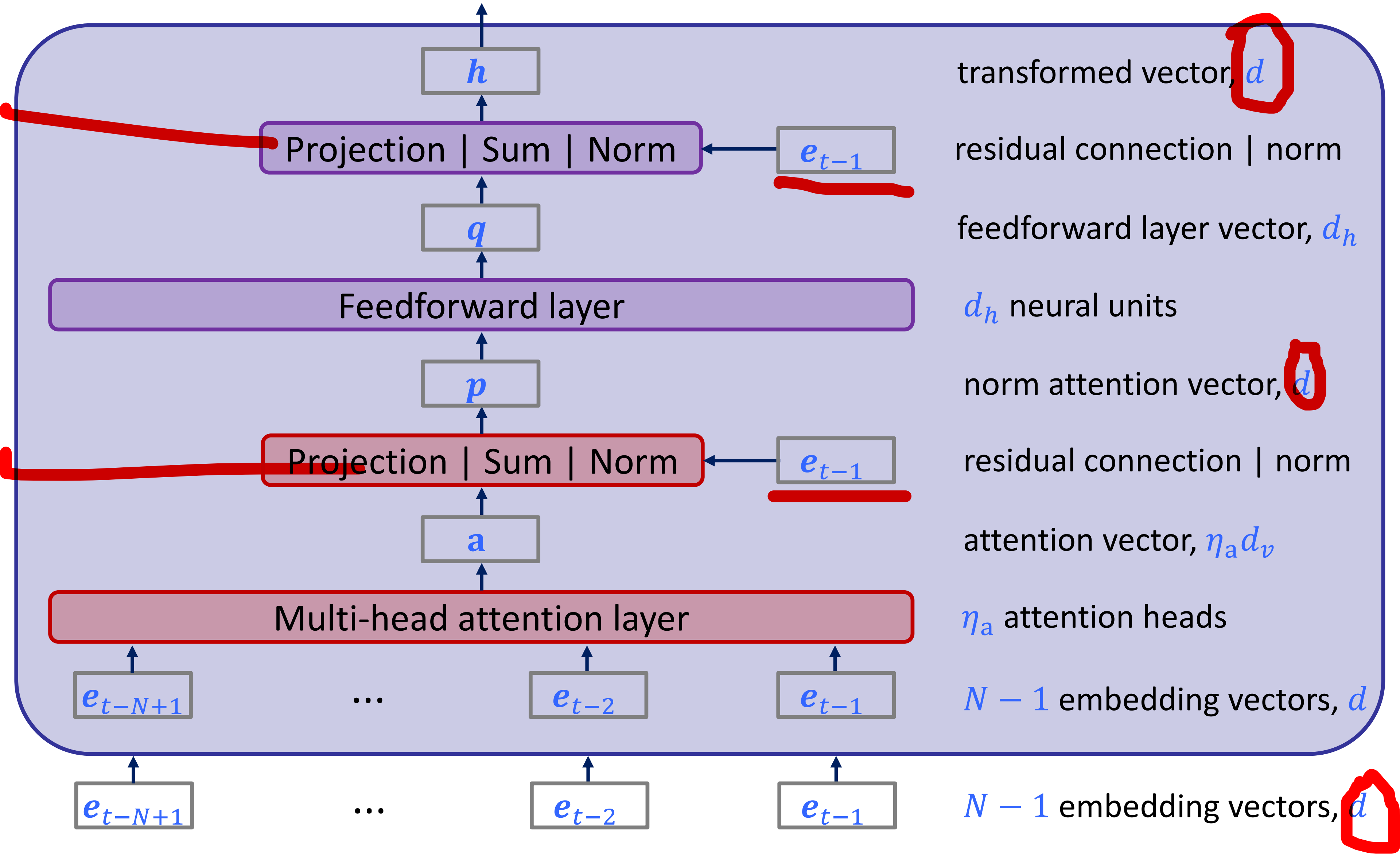
unembedding matrix, $|V|\times d$

$N-1$ embedding vectors, d

embedding matrix, $d\times|V|$

$N-1$ one-hot vectors, $|V|$

Transformer Block: Detailed Processing Steps



Transformer Block: [Attention Layer]

Input

- $N-1$ vectors of size d denoted by $\boldsymbol{e}_{t-N+1}, \dots, \boldsymbol{e}_{t-2}, \boldsymbol{e}_{t-1}$

Output

- 1 vector of size $\eta_{\mathrm{a}} d_v$ denoted by $\mathbf{a}$

Parameters for η_{a} attention heads

- For each attention head indexed i:
  - each query matrix W_i^q of size $d_k \times d$
  - each key matrix W_i^k of size $d_k \times d$
  - each value matrix W_i^v of size $d_v \times d$

Computation of output vector $\mathbf{a}$

- See earlier slide #18: *Layer of Attention Heads: Computing Attention Vector*

Transformer Block: [Attention Projection, Sum, Norm]

Input

- 1 vector of size $\eta_\mathrm{a} d_v$ denoted by attention vector $\mathbf{a}$
- 1 vector of size d denoted by residual connection $\boldsymbol{e}_{t-1}$

Output vector

- 1 vector of size d denoted by $\boldsymbol{p}$

Parameters

- attention layer projection matrix W^{AProj} of size $d \times \eta_\mathrm{a} d_v$
- two parameters for attention layer normalization denoted by $\gamma^{\mathrm{ANorm}}, \beta^{\mathrm{ANorm}}$

Computation of output vector $\boldsymbol{p}$

- Project $\mathbf{a}$ to a vector of size d: $W^{\mathrm{AProj}}\mathbf{a}$
- Add residual connection: $(W^{\mathrm{AProj}}\mathbf{a}) + \boldsymbol{e}_{t-1}$
- Normalize vector: $\mathrm{LayerNorm}\left((W^{\mathrm{AProj}}\mathbf{a}) + \boldsymbol{e}_{t-1}; \gamma^{\mathrm{ANorm}}, \beta^{\mathrm{ANorm}}\right)$
  - Optional reading: Chapters 9.2 provide details about this normalization step

Transformer Block: [Feedforward Layer]

Input

- 1 vector of size d denoted by $\boldsymbol{p}$

Output

- 1 vector of size d_h denoted by $\boldsymbol{q}$

Parameters for d_h neural units

- For neural unit i:
  - each weight vector $\boldsymbol{w}_i$ of size d
  - each bias term b_i of size 1

Computation of output vector $\boldsymbol{q}$

- See earlier slide #6: *Simple Feedforward Neural LM: Detailed Architecture*

Transformer Block: [Feedforward Projection, Sum, Norm]

Input

- 1 vector of size d_h denoted by feedforward layer vector $\boldsymbol{q}$
- 1 vector of size d denoted by residual connection $\boldsymbol{e}_{t-1}$

Output vector

- 1 vector of size d denoted by $\boldsymbol{h}$

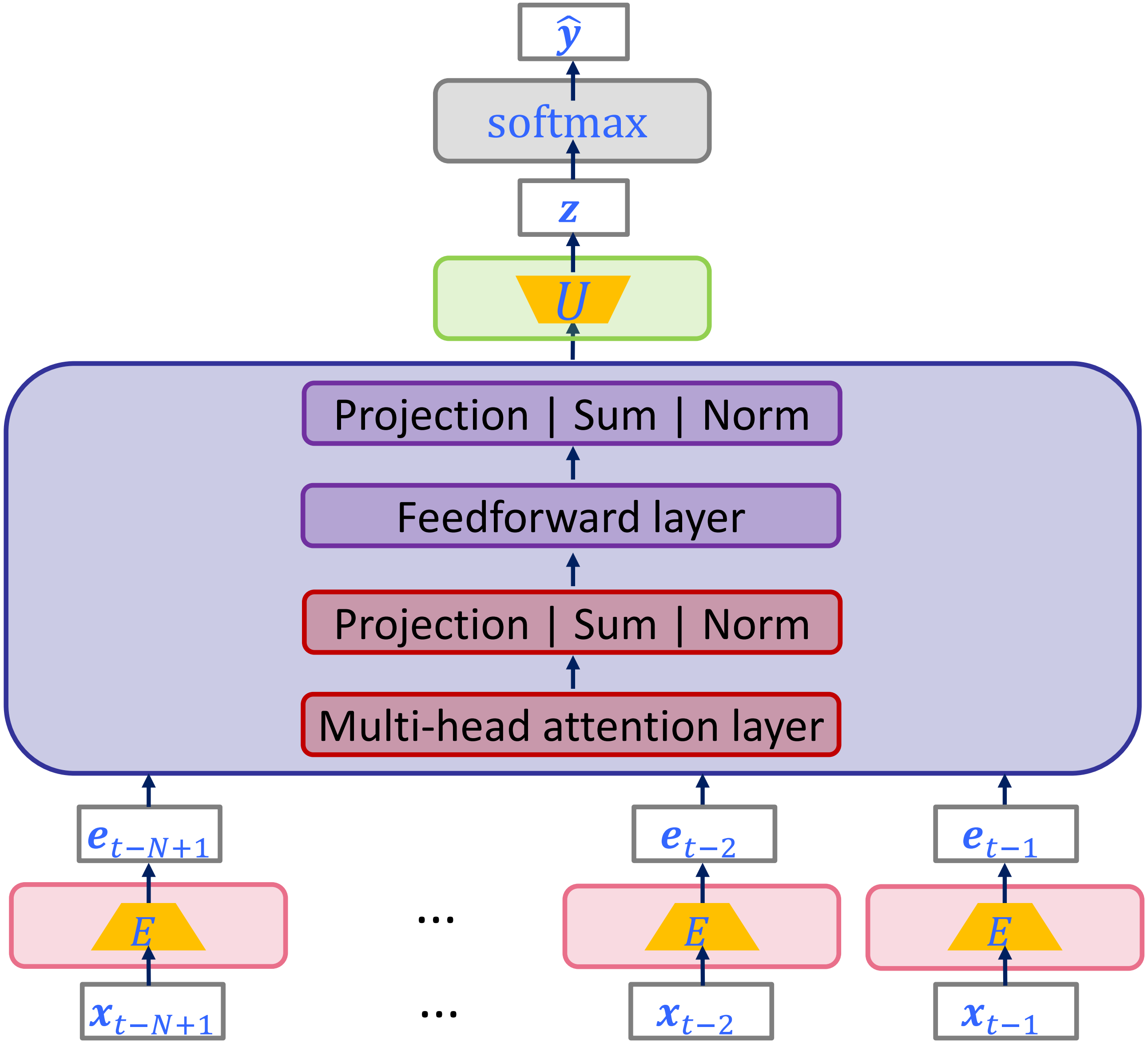
Parameters

- feedforward layer projection matrix W^{FfProj} of size $d \times d_h$
- feedforward layer bias term vector $\boldsymbol{b}^{\text{FfProj}}$ of size d
- two parameters for feedforward layer normalization denoted by $\gamma^{\text{FfNorm}}, \beta^{\text{FfNorm}}$

Computation of output vector $\boldsymbol{h}$

- Project $\boldsymbol{q}$ to a vector of size d and add bias term vector: $W^{\text{FfProj}}\boldsymbol{q} + \boldsymbol{b}^{\text{FfProj}}$
- Add residual connection: $\left(W^{\text{FfProj}}\boldsymbol{q} + \boldsymbol{b}^{\text{FfProj}}\right) + \boldsymbol{e}_{t-1}$
- Normalize vector: $\text{LayerNorm}\left(\left(W^{\text{FfProj}}\boldsymbol{q} + \boldsymbol{b}^{\text{FfProj}}\right) + \boldsymbol{e}_{t-1}; \gamma^{\text{FfNorm}}, \beta^{\text{FfNorm}}\right)$

Transformer Block: Interface and Processing



probability vector, $|V|$

softmax operation

score vector (logits), $|V|$

unembedding matrix, $|V|\times d$

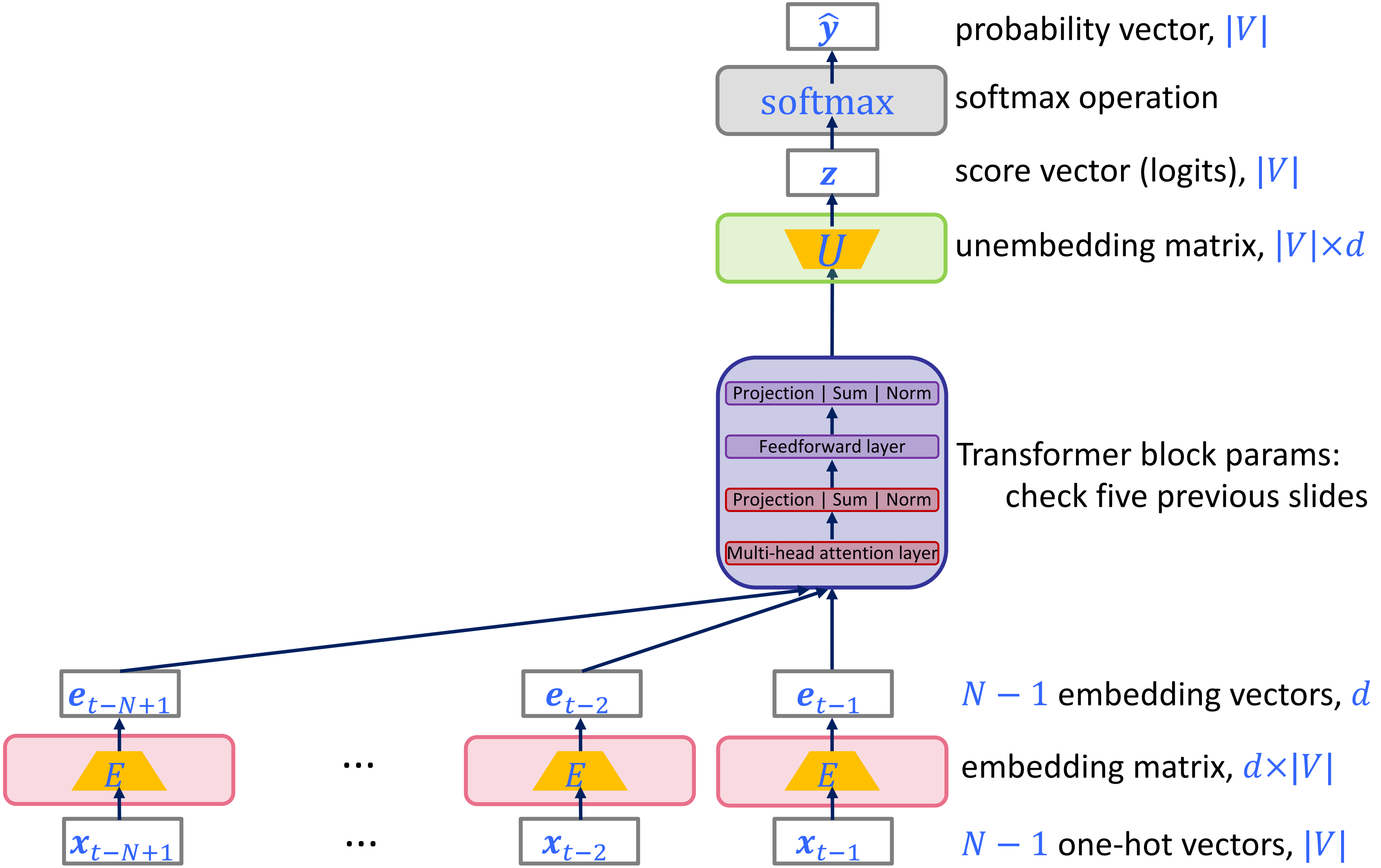
Transformer block params:
check five previous slides

$N-1$ embedding vectors, d

embedding matrix, $d\times|V|$

$N-1$ one-hot vectors, $|V|$

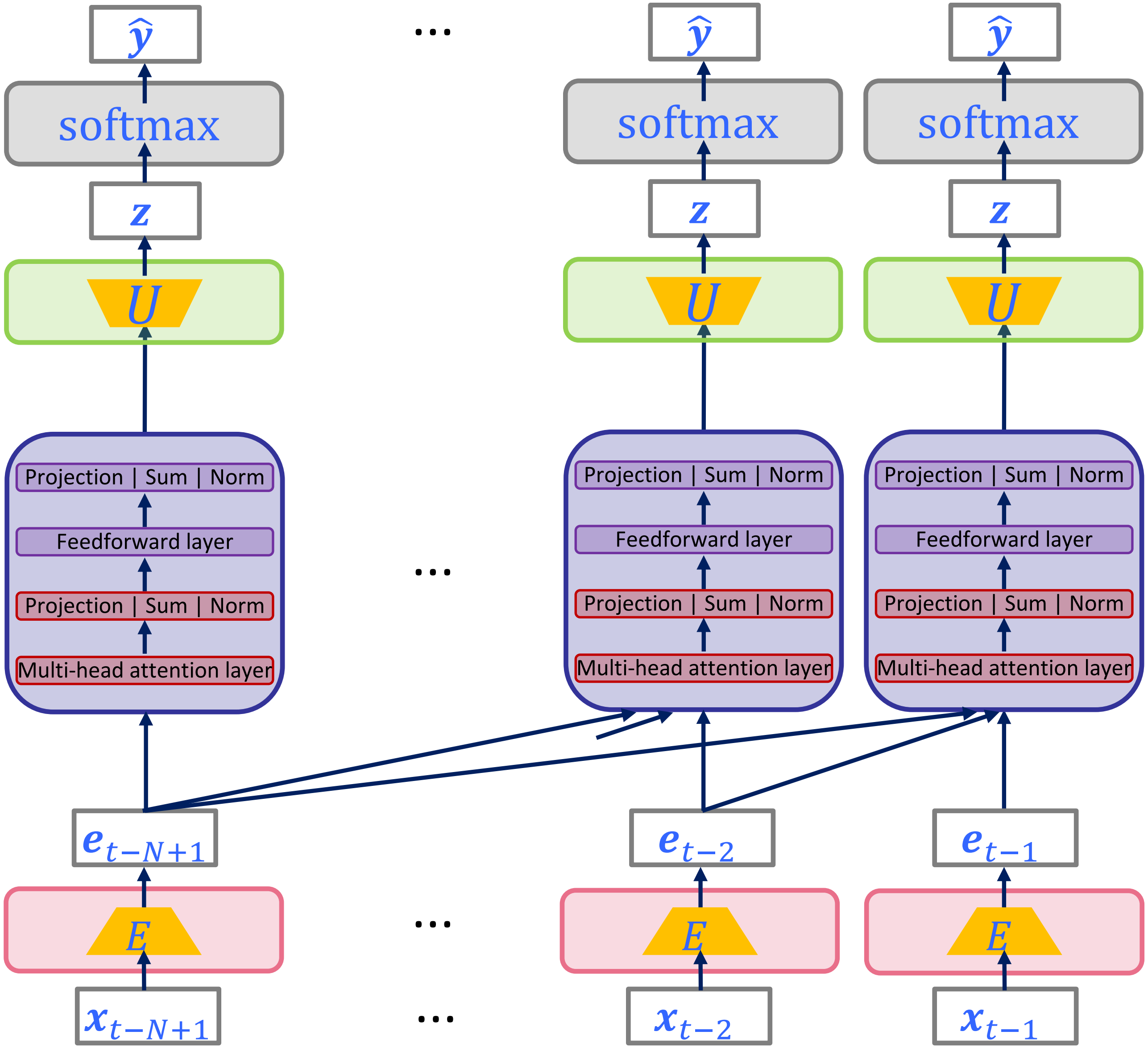
Transformer Block: Interface and Processing | Same as #28



Outline of the Lecture

- Organizational updates
- Week 2 recap and assignment
- From Simple Ff Neural LM to Transformer-based LM
  - Single attention head
  - Multiple attention heads
  - Transformer block
  - Transformer architecture
- GPT-1/2/3 models and in-context learning capabilities
- Week 3 assignment

Transformer Architecture: One Layer (Row) of Blocks



probability vector, $|V|$

softmax operation

score vector (logits), $|V|$

unembedding matrix, $|V| \times d$

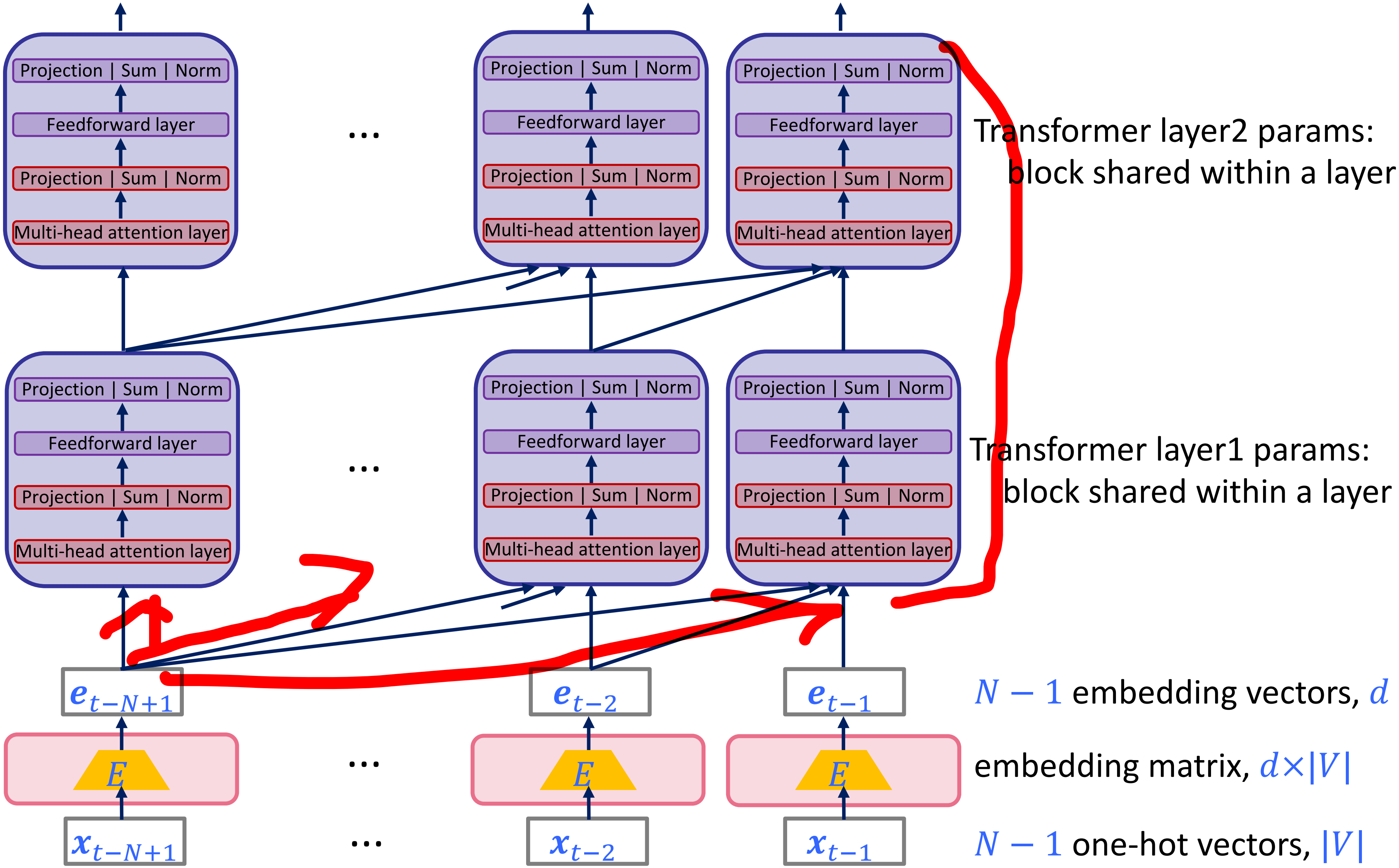
Transformer layer params:
block shared within a layer

$N-1$ embedding vectors, d

embedding matrix, $d \times |V|$

$N-1$ one-hot vectors, $|V|$

Transformer Architecture: Layers of Transformer Blocks



Transformer Architecture: Number of parameters

Parameters in L transformer layers

- The number of parameters in a transformer block, henceforth referred as B, is

$$B = (2d_k d + d_v d)\eta_{\mathrm{a}} + (d_v d \eta_a + 2) + (d + 1)d_h + (d d_h + d + 2)$$

- The number of parameters in L layers of transformer blocks is LB

Parameters in transformer architecture

- Consider weights of unembedding matrix U shared with that of E
  - U could be set as transpose of E
- The number of parameters in this architecture is $|V|d + LB$

Part of Week 3 assignment

- In the assignment, you will compute the number of parameters step by step

Transformer Architecture: Additional Remarks

Positional Information of Tokens: Chapter 9.4 of SLP book

- For a token $\boldsymbol{e}_{t-1}$, two types of embeddings vectors are combined by addition:
  - Embedding vector based on embedding matrix: $E\boldsymbol{e}_{t-1}$
  - Embedding vector based on position index: $t-1$

Parallelization of computation: Chapter 9.3 of SLP book

- During training, we can easily parallelize the entire computation
- During inference, tokens get generated one by one left to right
  - not parallelizable; attention embeddings get reused from previous time steps

Optional reading material

- Additional reading resources are provided as optional in assignments:
  - Week 3: Chapter 9 of SLP book; GPT-1 paper [Radford et al., ’18]
  - Week 2: Transformer [Vaswani et al., NeurIPS’17]
- Slightly different design, notation, and terminology across resources

Outline of the Lecture

- Organizational updates
- Week 2 recap and assignment
- From Simple Ff Neural LM to Transformer-based LM
  - Single attention head
  - Multiple attention heads
  - Transformer block
  - Transformer architecture
- GPT-1/2/3 models and in-context learning capabilities
- Week 3 assignment

How to build the GPT-1 Model?

- Let's instantiate the transformer architecture in slides #31—34 with following:
  - Context $N = 512$
  - Vocabulary $|V| = 40{,}478$
  - Transformer layers $L = 12$
  - Embedding dimension $d = 768$
  - Number of attention heads $\eta_a = 12$
  - Dimension for key and value vectors in attention head as $d_k = d_v = 64$
  - Number of neural units $d_h = 3072$
- That's $116{,}067{,}888$ parameters ➔ That's what is inside the GPT-1 model!

Optional reading material

- Week 3: GPT-1 paper [Radford et al., '18]
- Vocabulary: https://huggingface.co/docs/transformers/en/tokenizer_summary
- Andrej Karpathy's implementation: https://github.com/karpathy/minGPT/

  "…GPT is not a complicated model and this implementation is appropriately about 300 lines of code…"

Quick Evolution from GPT-1 to GPT-3

- 2018: OpenAI’s GPT-1 [Radford et al., ’18]
  - Size: 117 million parameters
  - Context length: 512
- 2019: OpenAI’s GPT-2 [Radford et al.,’19]
  - Size: 1.5 billion parameters
  - Context length: 1024
- 2020: OpenAI’s GPT-3 [Brown et al., NeurIPS’20]
  - Size: 175 billion parameters
  - Context length: 2048

Quick Evolution from GPT-1 to GPT-3

	Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{head}	Batch Size	Learning Rate
≈size GPT-1	GPT-3 Small	125M	12	768	12	64	0.5M	6.0×10^{-4}
	GPT-3 Medium	350M	24	1024	16	64	0.5M	3.0×10^{-4}
	GPT-3 Large	760M	24	1536	16	96	0.5M	2.5×10^{-4}
≈size GPT-2	GPT-3 XL	1.3B	24	2048	24	128	1M	2.0×10^{-4}
	GPT-3 2.7B	2.7B	32	2560	32	80	1M	1.6×10^{-4}
	GPT-3 6.7B	6.7B	32	4096	32	128	2M	1.2×10^{-4}
	GPT-3 13B	13.0B	40	5140	40	128	2M	1.0×10^{-4}
GPT-3	GPT-3 175B or "GPT-3"	175.0B	96	12288	96	128	3.2M	0.6×10^{-4}

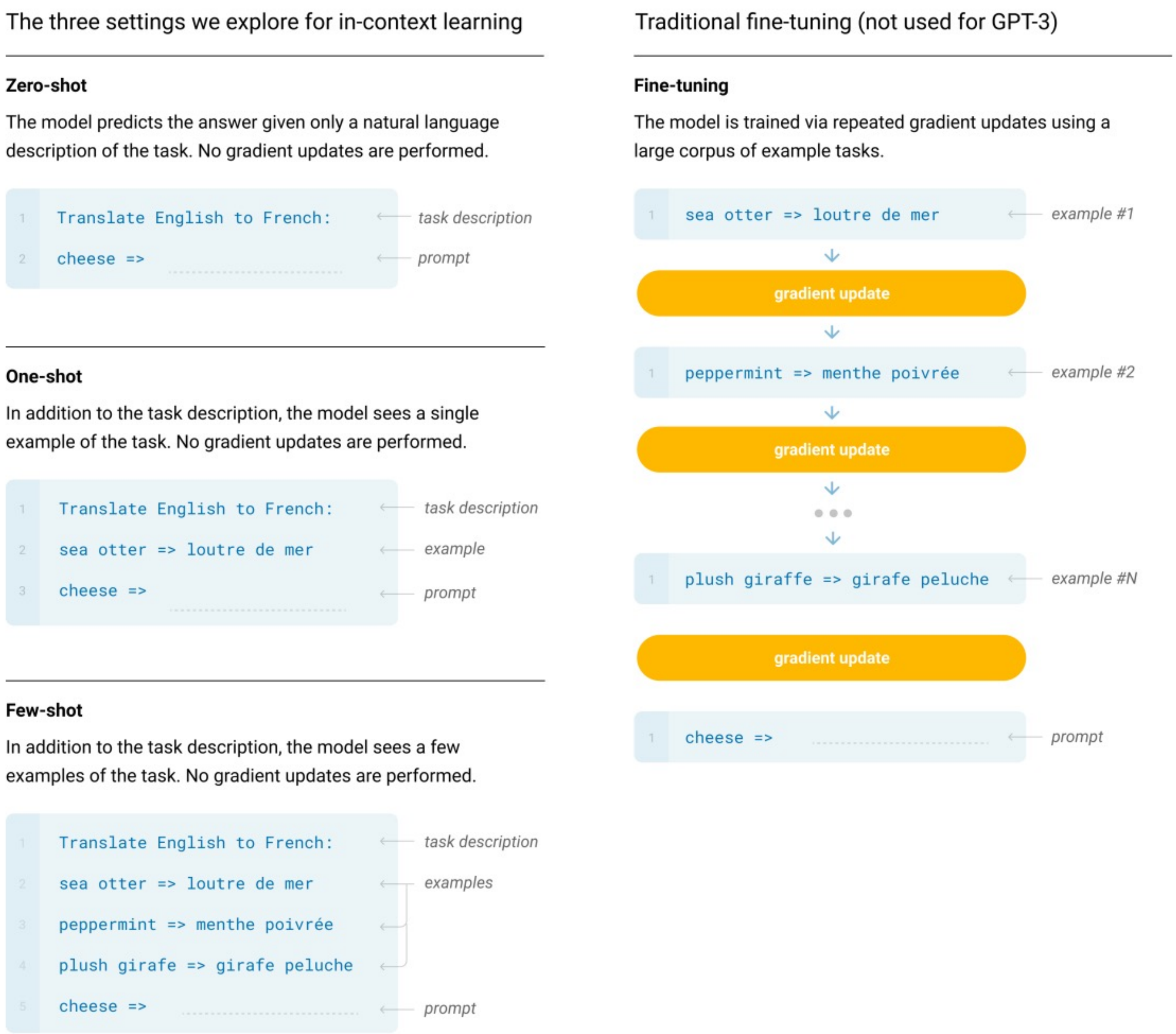
Table 2.1: Sizes, architectures, and learning hyper-parameters (batch size in tokens and learning rate) of the models which we trained. All models were trained for a total of 300 billion tokens.

Dataset	Quantity (tokens)	Weight in training mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4

Table 2.2: Datasets used to train GPT-3. "Weight in training mix" refers to the fraction of examples during training that are drawn from a given dataset, which we intentionally do not make proportional to the size of the dataset. As a result, when we train for 300 billion tokens, some datasets are seen up to 3.4 times during training while other datasets are seen less than once.

Source: GPT-3 [Brown et al., NeurIPS'20]; longer version of paper: https://arxiv.org/pdf/2005.14165

GPT-3 as a Few Shot Learner: In-context Learning Settings



Source: GPT-3 [Brown et al., NeurIPS'20]; longer version of paper: https://arxiv.org/pdf/2005.14165

GPT-3 as a Few Shot Learner: In-context Learning Gains

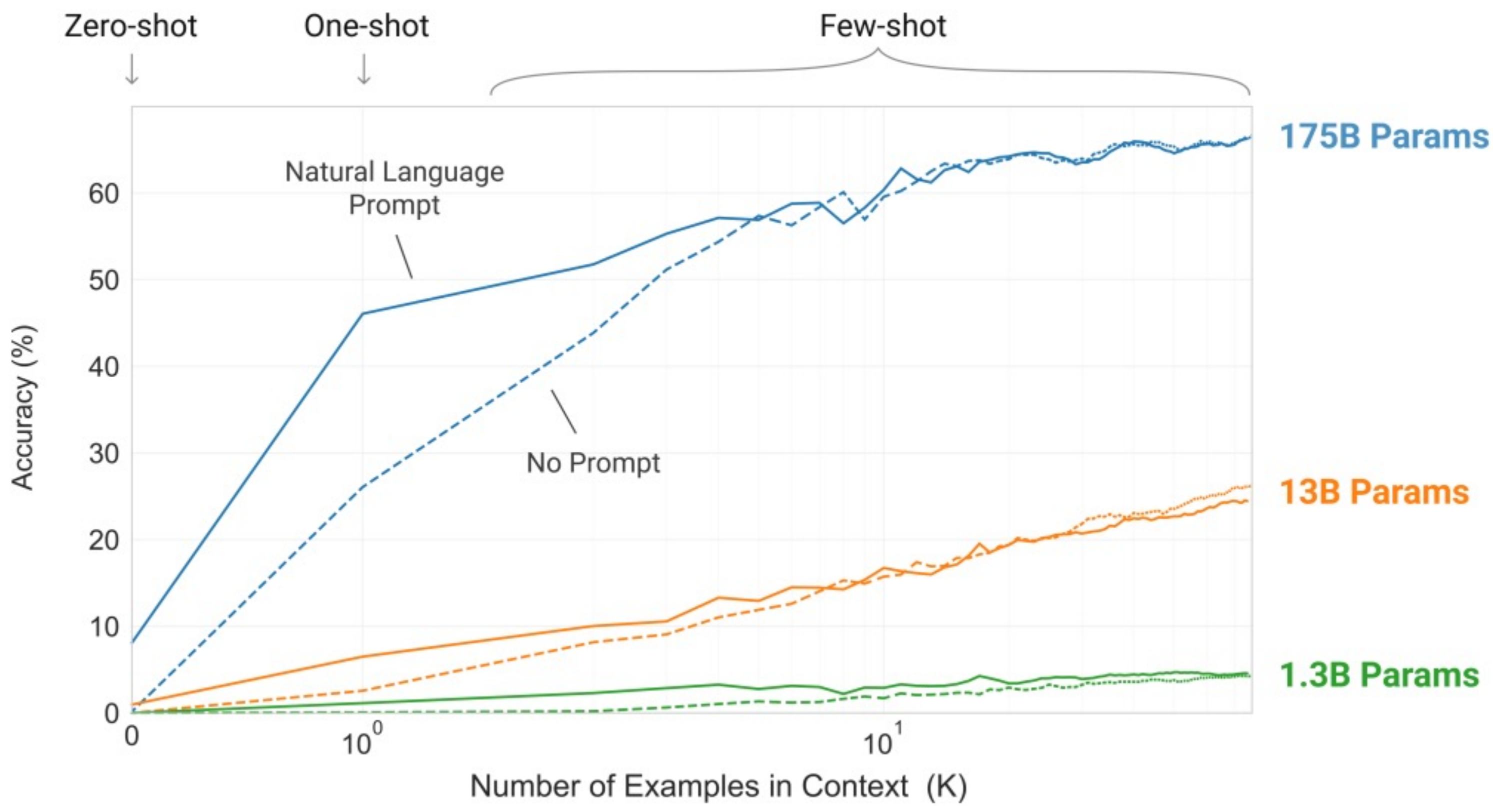


Figure 1.2: Larger models make increasingly efficient use of in-context information. We show in-context learning performance on a simple task requiring the model to remove random symbols from a word, both with and without a natural language task description (see Sec. 3.9.2). The steeper "in-context learning curves" for large models demonstrate improved ability to learn a task from contextual information. We see qualitatively similar behavior across a wide range of tasks.

Source: GPT-3 [Brown et al., NeurIPS'20]; longer version of paper: https://arxiv.org/pdf/2005.14165

GPT-3 as a Few Shot Learner: Results on Translation

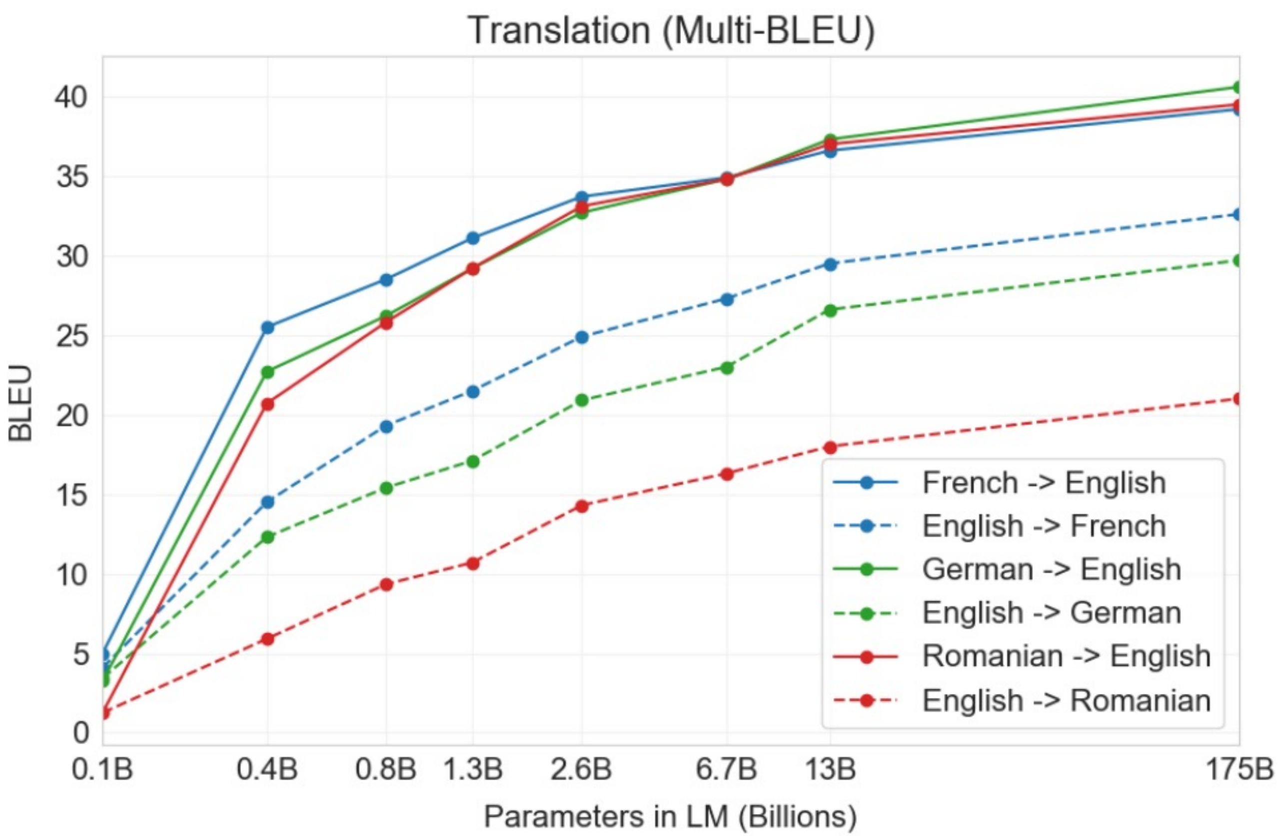


Figure 3.4: Few-shot translation performance on 6 language pairs as model capacity increases. There is a consistent trend of improvement across all datasets as the model scales, and as well as tendency for translation into English to be stronger than translation from English.

GPT-3 as a Few Shot Learner: Results on TriviaQA

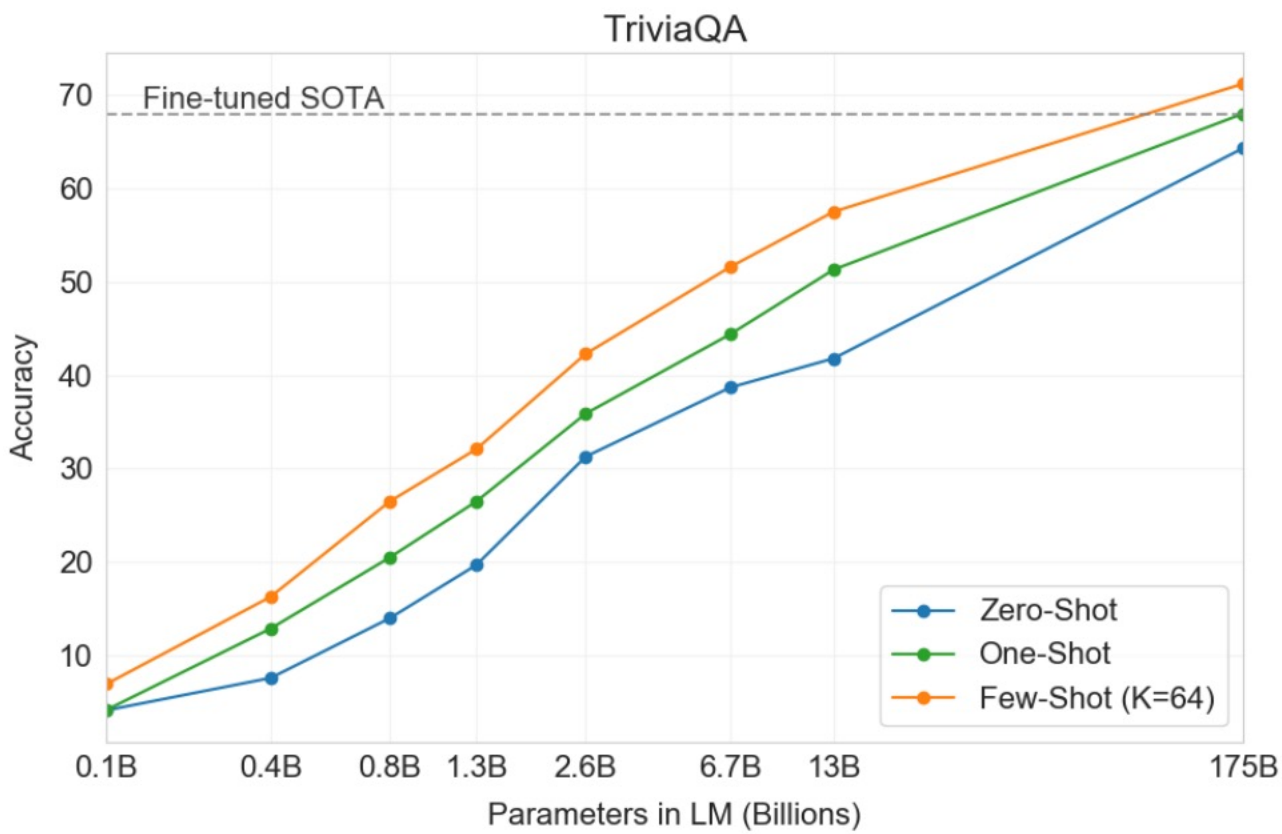


Figure 3.3: On TriviaQA GPT3's performance grows smoothly with model size, suggesting that language models continue to absorb knowledge as their capacity increases. One-shot and few-shot performance make significant gains over zero-shot behavior, matching and exceeding the performance of the SOTA fine-tuned open-domain model, RAG [LPP+20]

In-Context Learning and Prompting Strategies

Few-shot prompting

- Interesting questions: How many examples? How to select examples?
- Resources
  - Week 3 reading: Chapter 12.1 of SLP book; GPT-3 paper [Brown et al., NeurIPS'20]
  - Week 3 exercise: E.5

Retrieval augmented generation (RAG)

- Basic idea: Augment the prompt with documents relevant to a user's question
- Resources
  - Week 3 reading: Chapter 14.3 of SLP book

Chain-of-thought (CoT) prompting

- Basic idea: Ask the model to write detailed reasoning steps before final answer
- Resources
  - Week 3 reading: Chapter 12.4 of SLP book; CoT paper [Wei et al., NeurIPS'22]
  - Week 3 exercise: E.6

Outline of the Lecture

- Organizational updates
- Week 2 recap and assignment
- From Simple Ff Neural LM to Transformer-based LM
  - Single attention head
  - Multiple attention heads
  - Transformer block
  - Transformer architecture
- GPT-1/2/3 models and in-context learning capabilities
- Week 3 assignment

Week 3 Assignment

https://owncloud.mpi-sws.org/index.php/s/9YYZkDAeb58qiT2

From N-gram LMs to GPT-3

- 1970s: Resurgence of N-gram LMs in speech recognition, e.g., [Jelinek et al. ’75]

  …
- 2000: Simple Ff Neural LM [Bengio et al., NeurIPS’00]

  …
- 2014: Recurrent networks (RNN) for translation, e.g., [Sutskever et al., NeurIPS’14]
  - Encoder-decoder architectures with Long Short-Term Memory (LSTM) hidden units
- 2015: Attention mechanisms, e.g., [Bahdanau et al., ICLR’15]

  …
- 2017: Transformer [Vaswani et al., NeurIPS’17]
- 2018: OpenAI’s GPT-1 (~100 million parameters) [Radford et al., ’18]
- 2020: OpenAI’s GPT-3 (175 billion parameters) [Brown et al., NeurIPS’20]

- From 2021 onwards: Life after GPT-3

Course Timeline

[15 Oct] Week 1: Introduction

[22 Oct] Week 2: Background on Language Models and Transformers

[29 Oct] Week 3: Large Language Models and In-context Learning

From 1970's to 2020

[05 Nov] Week 4: Pre-training and Supervised Fine-tuning

[12 Nov] Week 5: Preference-based Fine-tuning for Alignment

[26 Nov] Week 6: Multi-modal Foundation Models

[03 Dec] Week 7: Trustworthiness Aspects I

[10 Dec] Week 8: Trustworthiness Aspects II

[07 Jan] Week 9: GenAI-powered Programming Education I

[14 Jan] Week 10: GenAI-powered Programming Education II

From 2021 onwards

[28 Jan] Week 11: Project Discussion

[04 Feb] Week 12: Examination Preparation