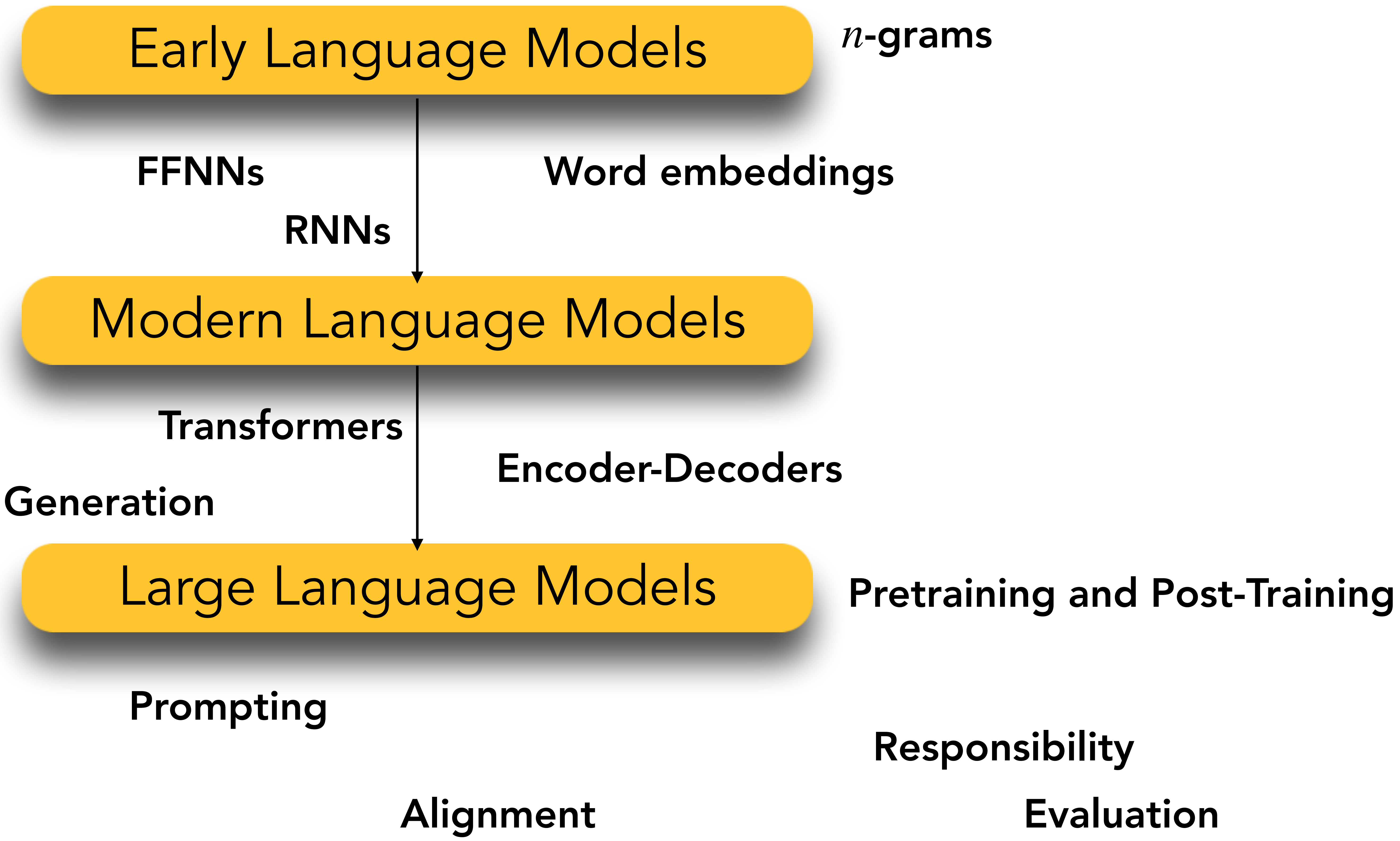
Lecture 19:
Outro

Instructor: Swabha Swayamdipta
USC CSCI 444 NLP
Nov 24, 2025

Some slides adapted from Dan Jurafsky and Chris Manning

Announcements and Logistics

- Today: Quiz 5 + Bonus Quiz
- Next week: Presentations: 15-20 mins + 5-10 mins QA
    - Too long / too short presentations will be penalized
    - There should be some more progress since the progress report
    - Please show up for other presentations + ask questions - this will be part of your participation grade
- Order of Presentations:
    - Dec 1
        - Alp + Valeria, Mo + Henry, Alexios + Ray, Shane + Vish
    - Dec 3
        - Naina + Kayal, Pinru + Harry, Aditya + Avi



Early Language Models

n-grams

FFNNs

Word embeddings

RNNs

Modern Language Models

Transformers

Encoder-Decoders

Generation

Large Language Models

Pretraining and Post-Training

Prompting

Responsibility

Alignment

Evaluation

Early Language Models

n-gram Language Models

- Goal: compute the probability of a sentence or sequence of words:
  - $P(\mathbf{w}) = P(w_1, w_2, w_3, w_4, w_5, \ldots w_n)$
- Chain Rule of Probability
  - $P(w_1, w_2, \ldots w_n) = P(w_1)P(w_2 | w_1)P(w_3 | w_2, w_1)\ldots = \prod_{i=1}^{n} P(w_i | w_{i-1} \ldots w_1)$
- k-gram Assumption
  - $P(w_1, w_2, \ldots w_n) \approx \prod_{i} P(w_i | w_{i-k+1} \ldots w_{i-1})$
- Insufficient model of language
  - Language has long-distance dependencies
  - Smoothing alleviates some of these issues

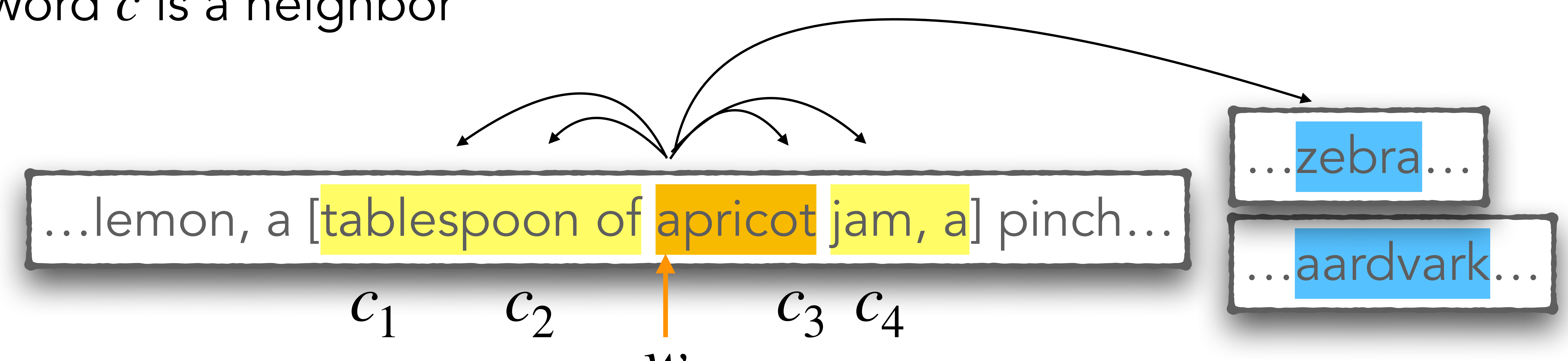
$$P_{MLE}(w_i | w_{i-1}) = \frac{c(w_{i-1}w_i)}{c(w_{i-1})}$$

Bigram Maximum Likelihood Estimate

Word Embeddings

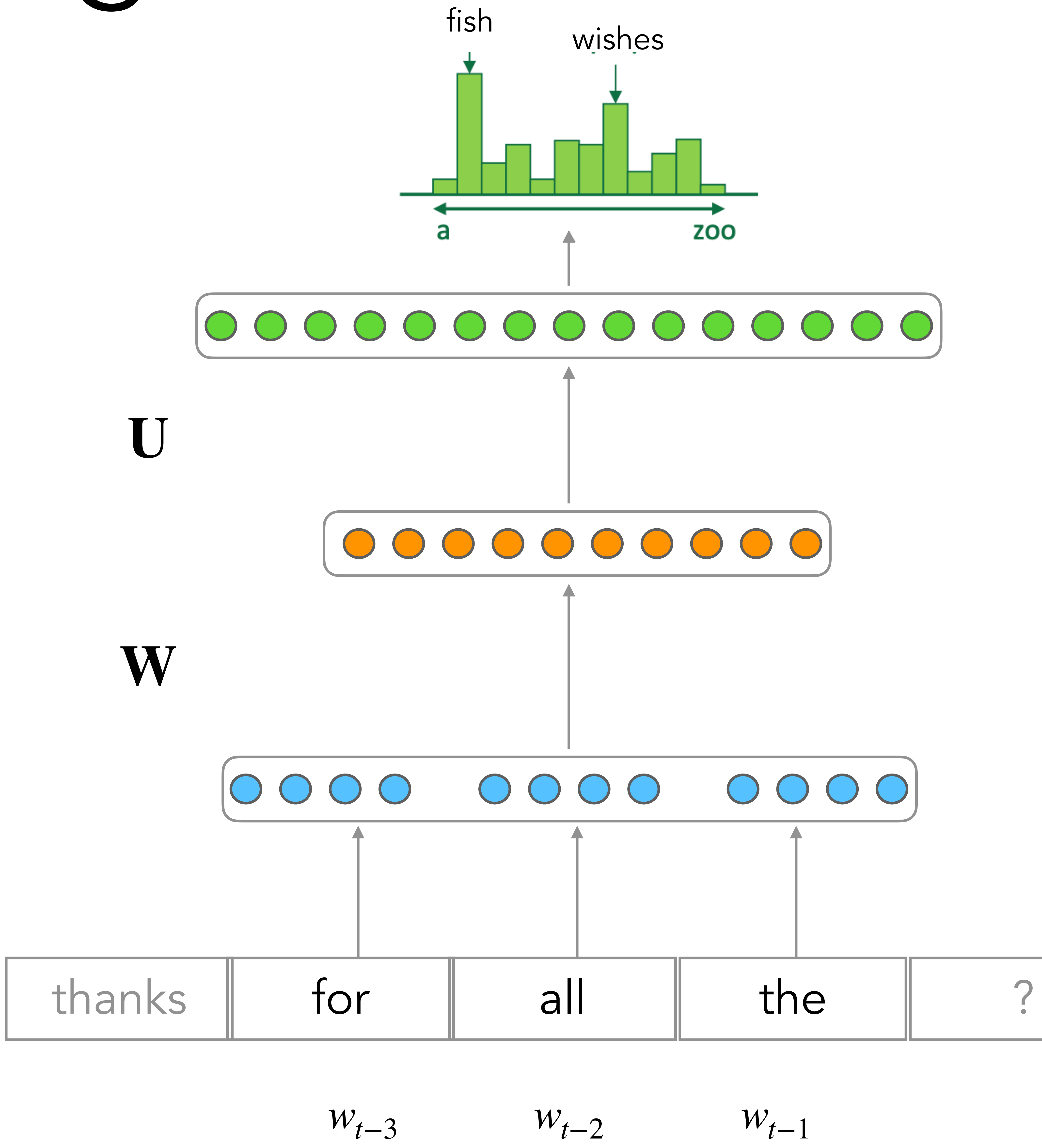
- Represent a word as a point in a multidimensional semantic space
- Fine-grained model of meaning for similarity
  - Cosine similarity of two vectors
- Learning word embeddings: both features and parameters!
- word2vec
  - Skip-Gram with Negative Sampling or SGNS
  - CBOW or continuous bag of words
  - Predict if candidate word c is a neighbor

$$\cos(\vec{v}, \vec{w}) = \frac{\vec{v} \,.\, \vec{w}}{|\vec{v}||\vec{w}|}$$



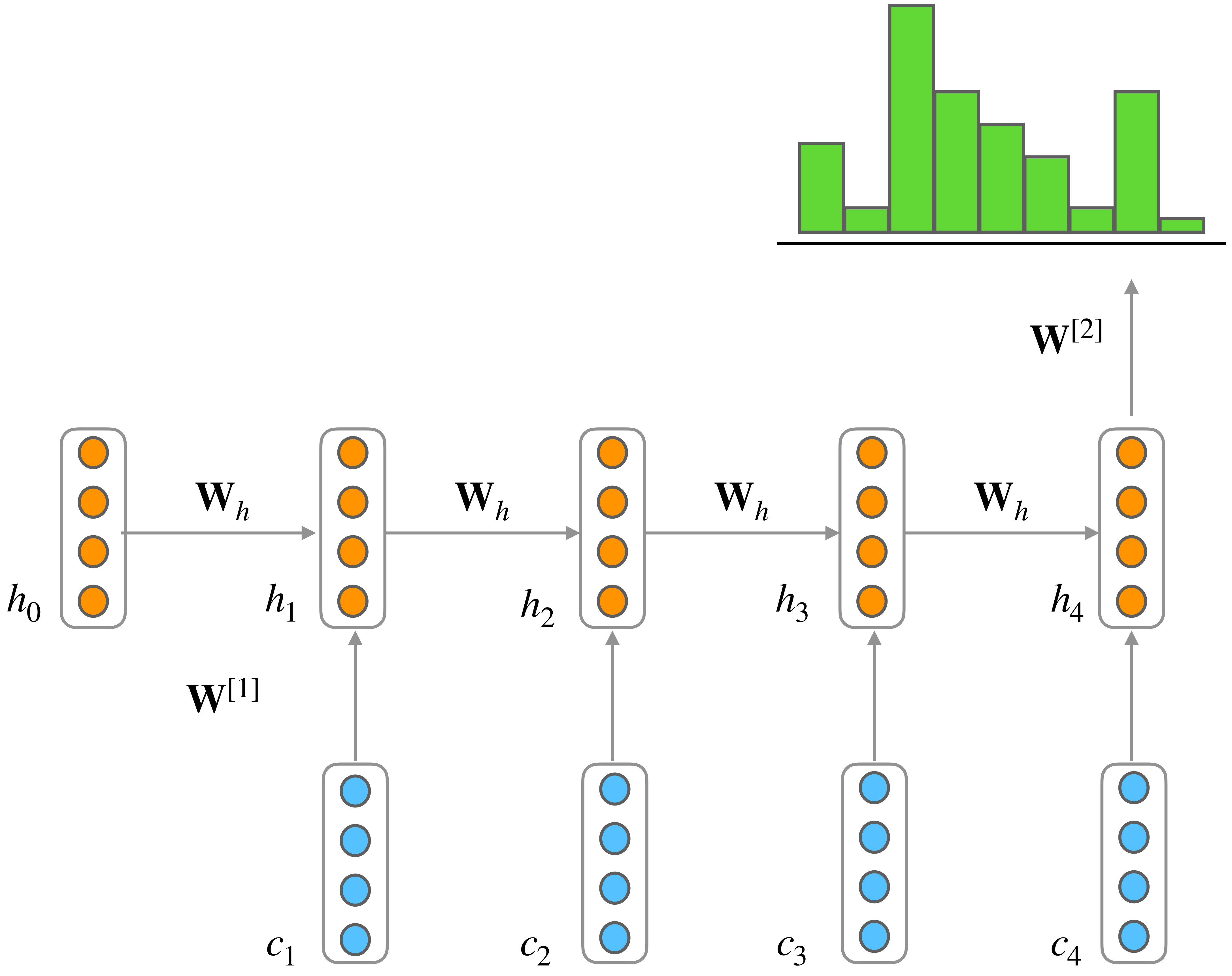
Feedforward Language Models

- Simplest Neural Net
- Processes words within a fixed context window
  - Enlarging window enlarges $\mathbf{W}$
  - Order information?
- Each word uses different rows of $\mathbf{W}$. We don't share weights within the window.
- Parallelized computation within each window
- Training through back propagation
  - Forward pass: compute loss
  - Backward pass: update weights



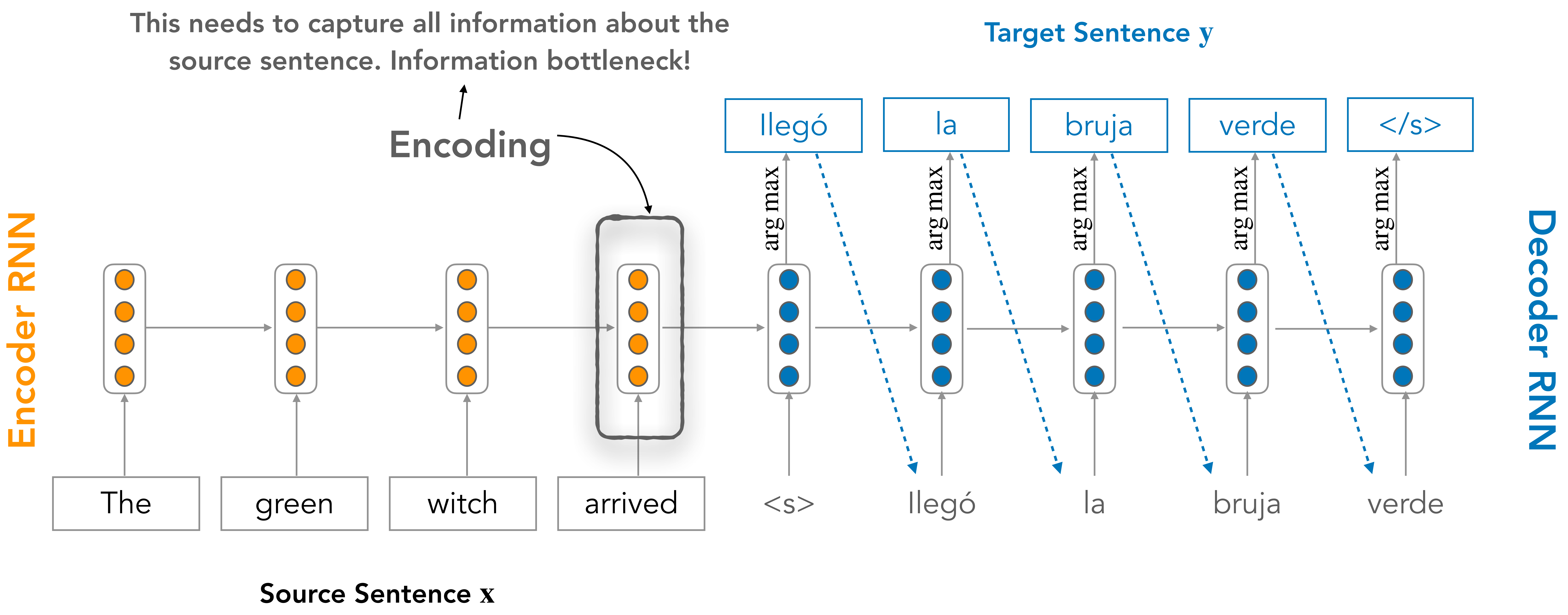
Recurrent Language Models

- Recurrent Neural Networks processes sequences one element at a time
- RNNs do not have
  - the limited context problem of n-gram models
  - the fixed context limitation of feedforward LMs
  - since the hidden state can *in principle* represent information about all of the preceding words all the way back to the beginning of the sequence
- But training RNNs is hard
  - Vanishing gradient problem
  - LSTMs address it by incorporating a memory

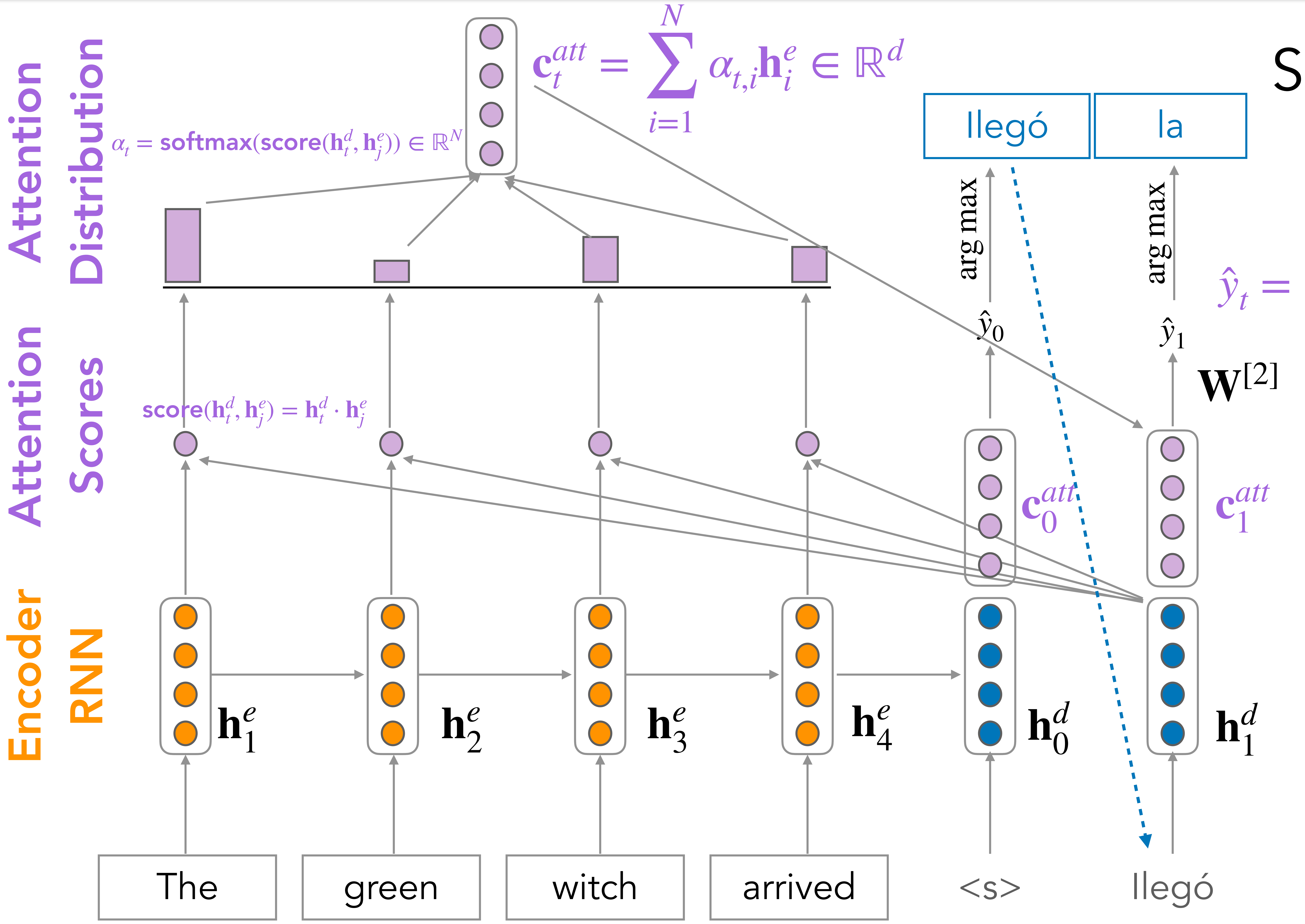


Modern Language Models

Encoders-Decoders


This needs to capture all information about the source sentence. Information bottleneck!
Encoding
Target Sentence $\mathbf{y}$
llegó
la
bruja
verde
</s>
arg max
arg max
arg max
arg max
arg max
Encoder RNN
Decoder RNN
The
green
witch
arrived
<s>
llegó
la
bruja
verde
Source Sentence $\mathbf{x}$

Seq2Seq with Attention



$$\hat{y}_t = \text{softmax}(\mathbf{W}^{[2]}[\mathbf{h}_t^d; \mathbf{c}_t^{att}])$$

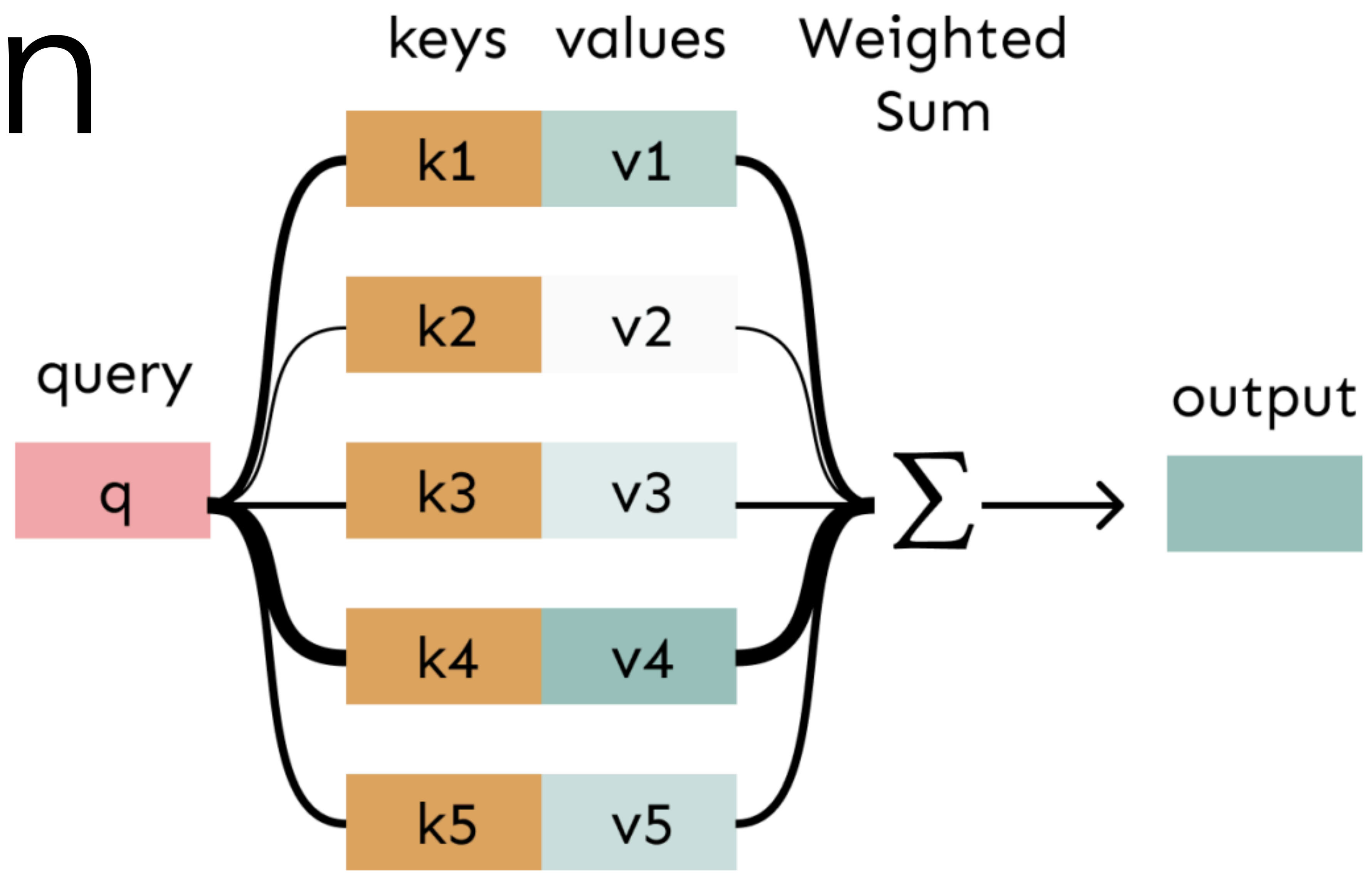
$$\mathbf{W}^{[2]} \in \mathbb{R}^{V \times 2d}$$

Query 2: Decoder, second time step

Source Sentence $\mathbf{x}$

Note: Notation different from J&M

Self-Attention



Keys, Queries, Values from the same sequence

Let $\mathbf{w}_{1:N}$ be a sequence of words in vocabulary V

For each $\mathbf{w}_i$, let $\mathbf{x}_i = \mathbf{E}_{w_i}$, where $\mathbf{E} \in \mathbb{R}^{d \times V}$ is an embedding matrix.

1. Transform each word embedding with weight matrices $\mathbf{Q}, \mathbf{K}, \mathbf{V}$, each in $\mathbb{R}^{d \times d}$

$q_i = Q\boldsymbol{x_i}$ (queries) $k_i = K\boldsymbol{x_i}$ (keys) $v_i = V\boldsymbol{x_i}$ (values)

2. Compute pairwise similarities between keys and queries; normalize with softmax

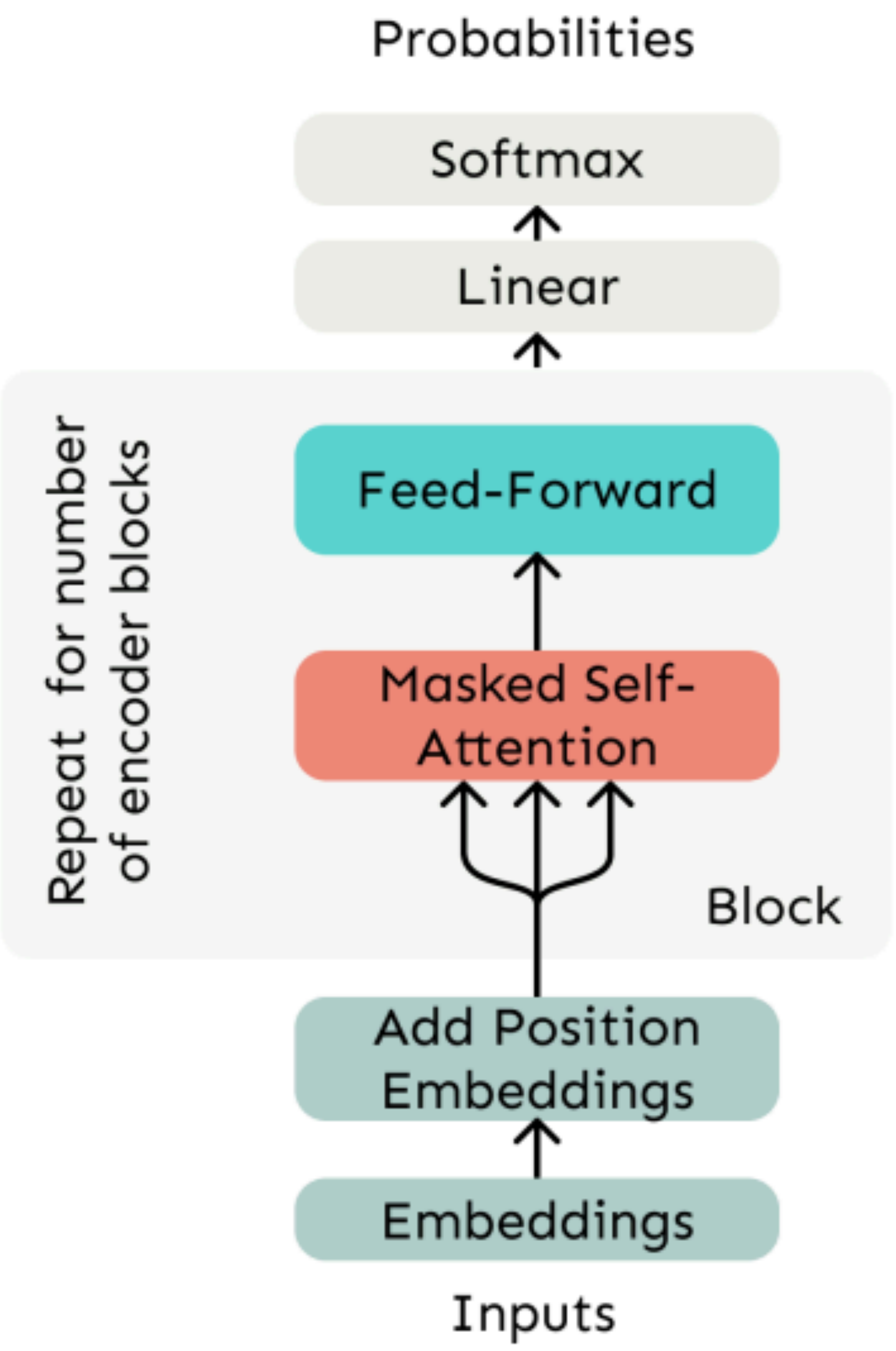
$$\boldsymbol{e}_{ij} = \boldsymbol{q}_i^\top \boldsymbol{k}_j \qquad \boldsymbol{\alpha}_{ij} = \frac{\exp(\boldsymbol{e}_{ij})}{\sum_{j'} \exp(\boldsymbol{e}_{ij'})}$$

3. Compute output for each word as weighted sum of values

$$\boldsymbol{o}_i = \sum_j \boldsymbol{\alpha}_{ij}\, v_i$$

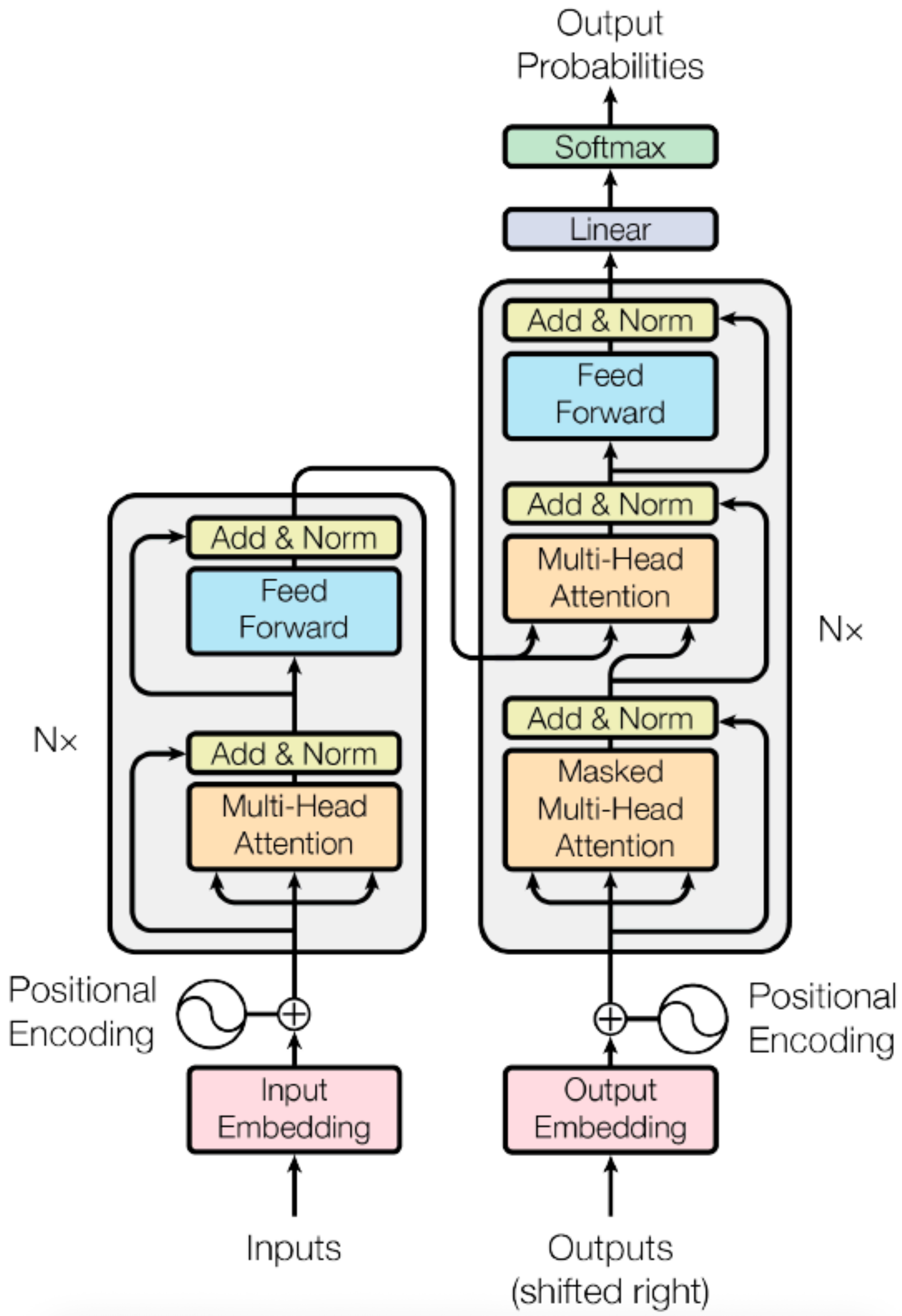
Self-Attention Transformer Building Block

- Self-attention:
  - the basis of the method; with multiple heads
- Position representations:
  - Specify the sequence order, since self-attention is an unordered function of its inputs.
- Nonlinearities:
  - At the output of the self-attention block
  - Frequently implemented as a simple feedforward network.
- Masking:
  - In order to parallelize operations while not looking at the future.
  - Keeps information about the future from "leaking" to the past.



Transformer Language Models

- Transformers are made up of stacks of transformer blocks, each of which is a multilayer network made by combining feedforward networks and **self-attention layers**, the key innovation of self-attention transformers
- The Transformer Decoder-only model corresponds to
  - a Transformer language model
- Lookup embeddings for tokens are usually randomly initialized

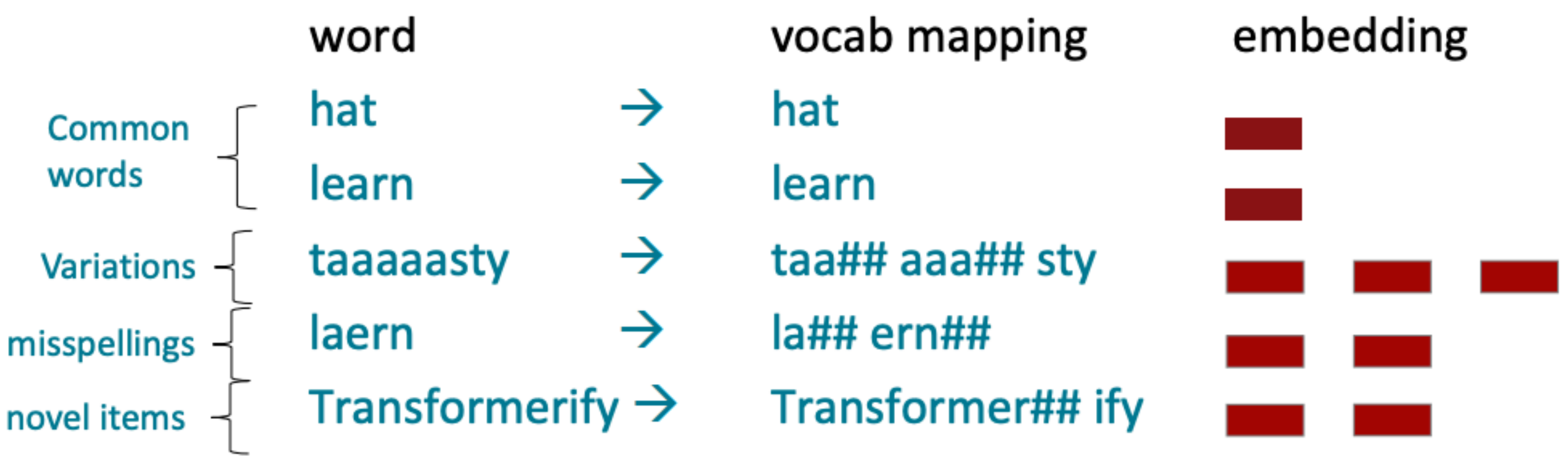


Attention is all you need (Vaswani et al., 2017)

Tokenization

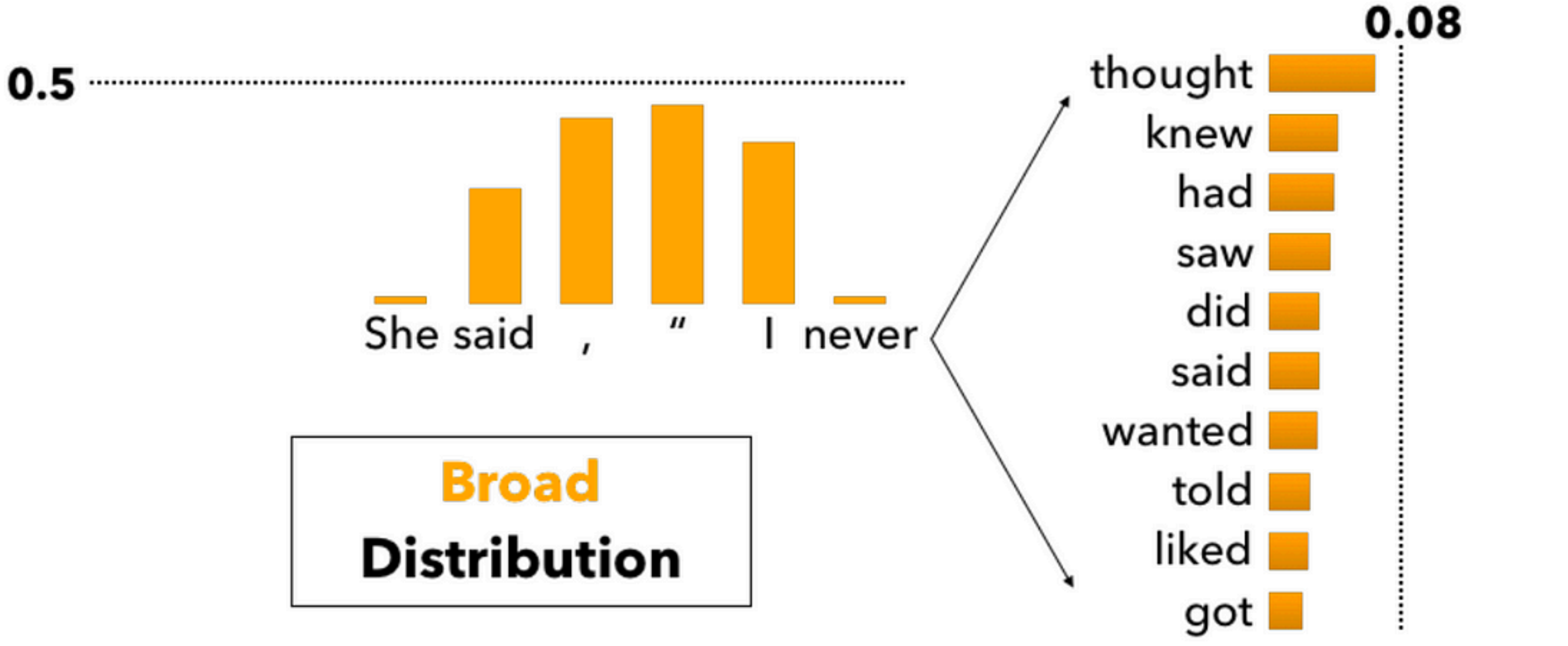
- Tokenization: how to get the most frequent tokens by merging characters or character sequences
- Byte Pair Encoding Algorithm:
  1. Start with a vocabulary containing only characters and an "end-of-word" symbol.
  2. Using a corpus of text, find *the most common adjacent characters* "a,b"; add "ab" as a subword
     - This is a learned operation! However, not a parametric function
     - Only combine pairs (hence the name!)
  3. Replace instances of the character pair with the new subword; repeat until desired vocabulary size.
- At test time, first split words into sequences of characters, then apply the learned operations to merge the characters into larger, known symbols

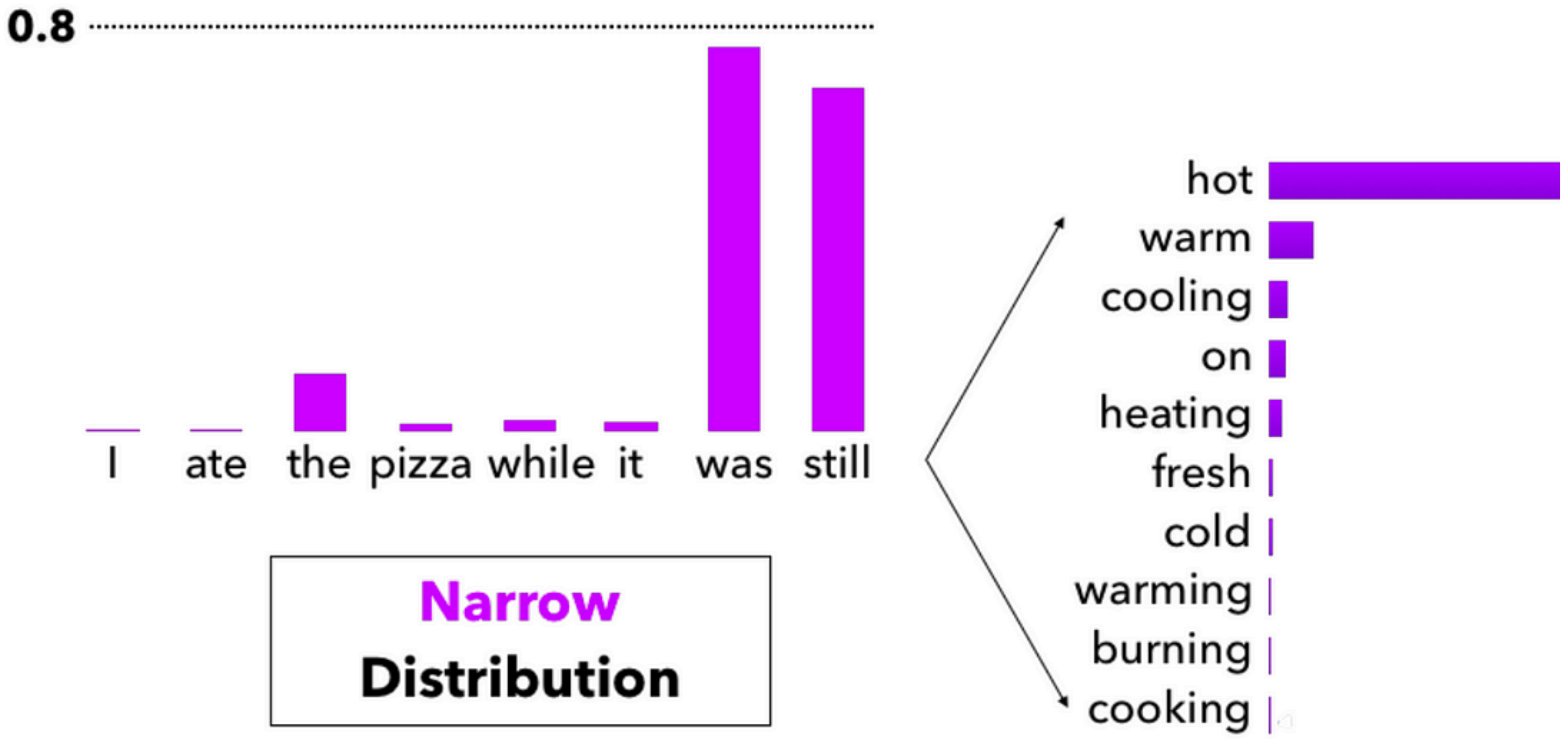
- **WordPiece Modeling:** Similar to BPE
  - Each word is initially split by adding a prefix (##) to all the characters inside a word



	word		vocab mapping	embedding
Common words	hat	→	hat	🟥
	learn	→	learn	🟥
Variations	taaaaasty	→	taa## aaa## sty	🟥 🟥 🟥
misspellings	laern	→	la## ern##	🟥 🟥
novel items	Transformerify	→	Transformer## ify	🟥 🟥

Generation





WebText

So what's new in my life?
09/11/18 - Just got back from vacation.

Beam Search, *b*=16

Just got back from vacation.
09/10/18 - Just got back from vacation. Just got back from vacation.
09/09/18 - Just got back from vacation. Just got back from vacation.
09/08/18 - Just got back from vacation. Just got back from vacation.

Pure Sampling

01/05/19 - Visited The Subaru Canada Heritage15th Star for my vehicle research. That was a wonderful experience. Thanks again for your knowledge and support.
05/23/19 - I was thrilled to receive my Subaru Legacy character! It fits me perfectly and makes me feel like a kid again!
06/15/19 - Just regained my licence for my different coloured 2016 WRX / STI...

Sampling, *t*=0.9

Just got back from vacation.
09/07/18 - Happy 9th birthday to me! Happy 9th birthday to me!
09/01/18 - Sofwidden also! Sofwidden also!
08/25/18 - Happy birthday to me! Happy birthday to me!
08/17/18 - Night shift! □□Night shift! □□
08/17/18 - Lucky me! Lucky me!

Top-*k*, *k*=640

Just got back from vacation.
09/03/18 - After more than 40 years, I might have an apartment in Virginia Beach.
After more than 40 years, I might have an apartment in Virginia Beach.
08/20/18 - Going for a hike at Mount Eerie in Northeast Virginia Spring
Going for a hike at Mount Eerie in Northeast Virginia Spring

Top-*k*, *k*=640, *t*=0.7

Just got back from vacation.
09/08/18 - I've gotten really sick. - I've gotten really sick.
09/07/18 - My wife and I are getting married in February.
- My wife and I are getting married in February.
09/06/18 - I'm so excited to go back to college this fall.
- I'm so excited to go back to college this fall.

Nucleus, *p*=0.95

Just got back from vacation.
07/12/18 - Happy birthday to Swingu, who is nearly 5 years old. I would like to say hi to him on the road as well as when I ride with him. You cannot go to work without feeling physically sick or psychologically exhausted because you can barely breathe. Even if you ride on rollercoaster even once, it is easy to recover from the physical side of it.

WebText

I just got back from a much needed and really great nine day vacation to my remote Arizona property. It was a really restful and relaxing visit. I got a lot accomplished while I was there, but still found time to just goof off and have fun too. I got to do some astronomy, even though the weather was pretty cloudy most of the time. Here is a 50 minute exposure of M101. It turned out pretty good.

Large Language Models

Three Stages of Training LLMs

Stage 1

- Pre-training on large corpus of text
- Produces a **Base Language Model**
- Continued Pre-training for domain adaptation (optional; sometimes called mid-training, stage 1.5)

Stage 2

- Post-training for Task Adaptation
- Seq2Seq Instruction Tuning (Supervised Finetuning, different meaning than BERT-style classification tasks)

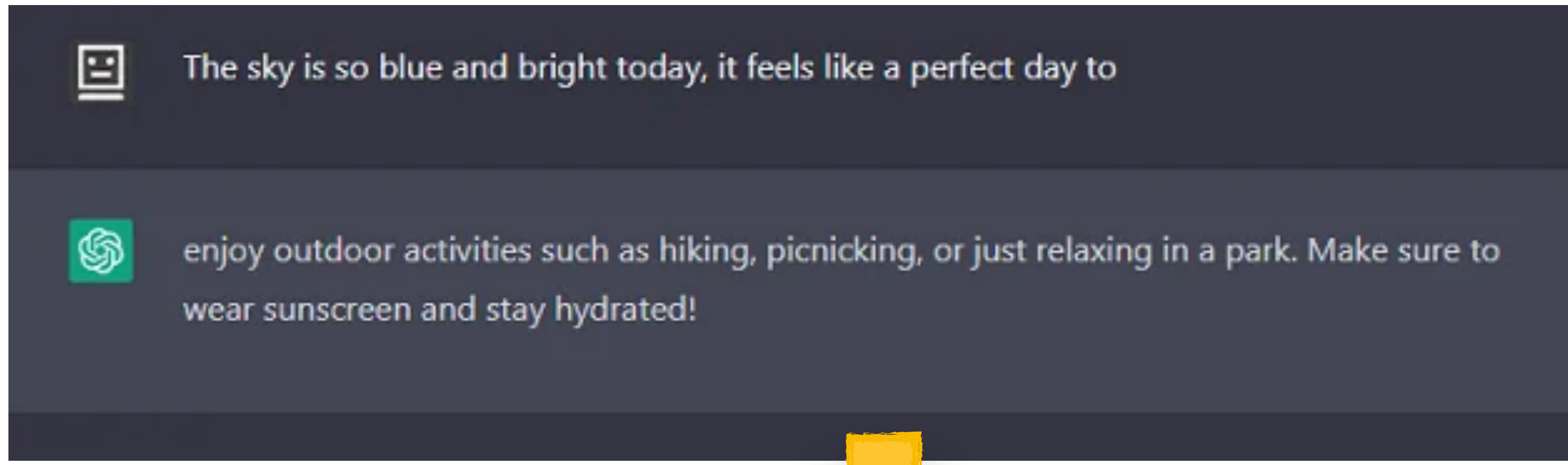
Stage 3

- Post-training for Preference Alignment
  - Either Reinforcement Learning with Human Feedback : Train a supervised classifier (reward model) on human demonstrations to provide feedback to LM. Reinforcement learning to maximize rewards given by reward model
  - Or, train with a preferred and a dispreferred generation (more popular now)

Inference: Prompting with Instructions and Demonstrations (also called examples, shots)

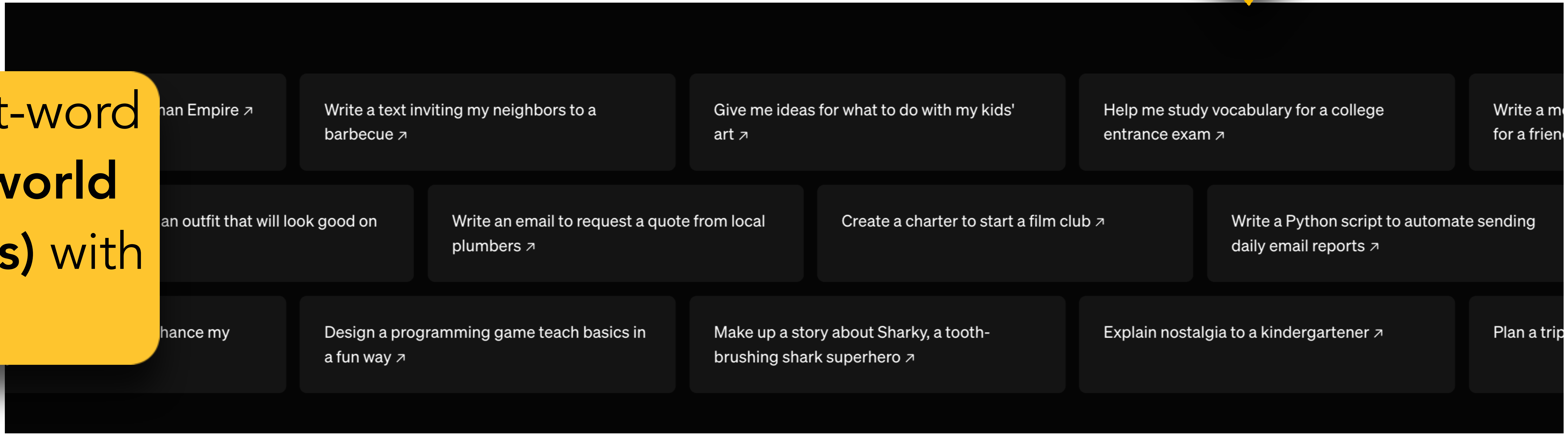
Pre-training and Post-training

- The term “training” is no longer specific enough :)
- Pre-training: Self-supervised, standard next token prediction
- Post-training: Supervision (sequence to sequence)
  - **Instruction-Tuning**: Supervision is not necessarily via labels, but sequence pairs. Labels in standard NLP benchmarks can be converted into sequence pairs
  - **Preference-Tuning**: Collects human judgments / preferences as rewards, or a pair of preferred / dispreferred generations



The sky is so blue and bright today, it feels like a perfect day to

enjoy outdoor activities such as hiking, picnicking, or just relaxing in a park. Make sure to wear sunscreen and stay hydrated!



an Empire ↗

Write a text inviting my neighbors to a barbecue ↗

Give me ideas for what to do with my kids' art ↗

Help me study vocabulary for a college entrance exam ↗

Write a m
for a frien

an outfit that will look good on

Write an email to request a quote from local plumbers ↗

Create a charter to start a film club ↗

Write a Python script to automate sending daily email reports ↗

hance my

Design a programming game teach basics in a fun way ↗

Make up a story about Sharky, a tooth-brushing shark superhero ↗

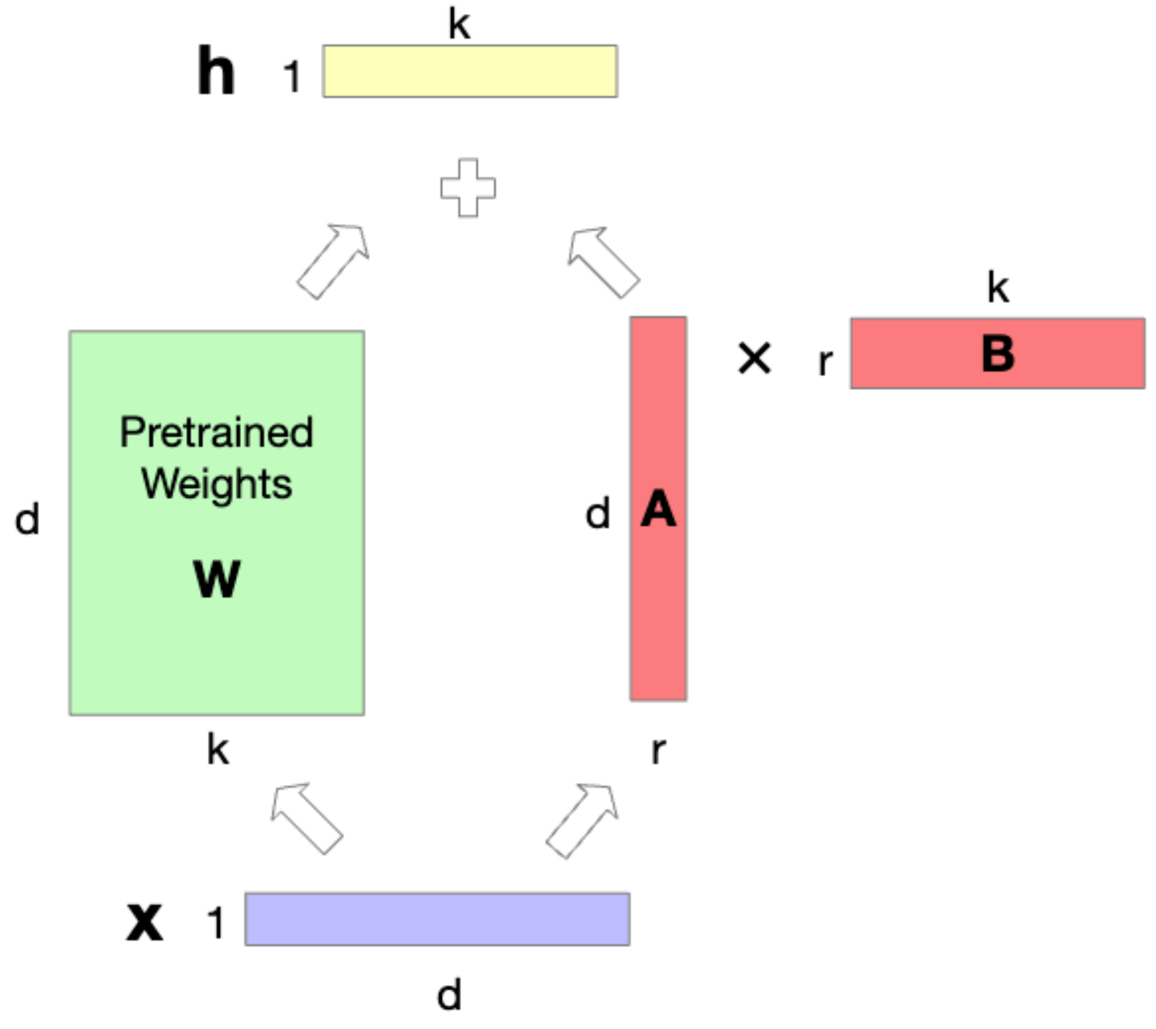
Explain nostalgia to a kindergartener ↗

Plan a trip

Post-training converts next-word completion models into **world models (agents, assistants)** with many capabilities!

LoRA: Low-Rank Adaptation

- Instead of updating weight matrices in attention layers during finetuning, Low-Rank Adaptation eases this by updating a low-rank approximation of the matrix : matrices which are far smaller
- LoRA freezes the pre-trained model weights and injects trainable rank decomposition matrices into each layer of the Transformer architecture
- Gradient updates are reparametrized as $\mathbf{W}_0 + \Delta\mathbf{W} = \mathbf{W}_0 + \mathbf{B}\cdot\mathbf{A}$
  - Where $\mathbf{B}$ and $\mathbf{A}$ are low-rank matrices, the only matrices to be updated
- LoRA can reduce the number of trainable parameters by 10,000 times and the GPU memory requirement by 3 times for GPT-3 175B
- Usually comes at a (small) cost to performance



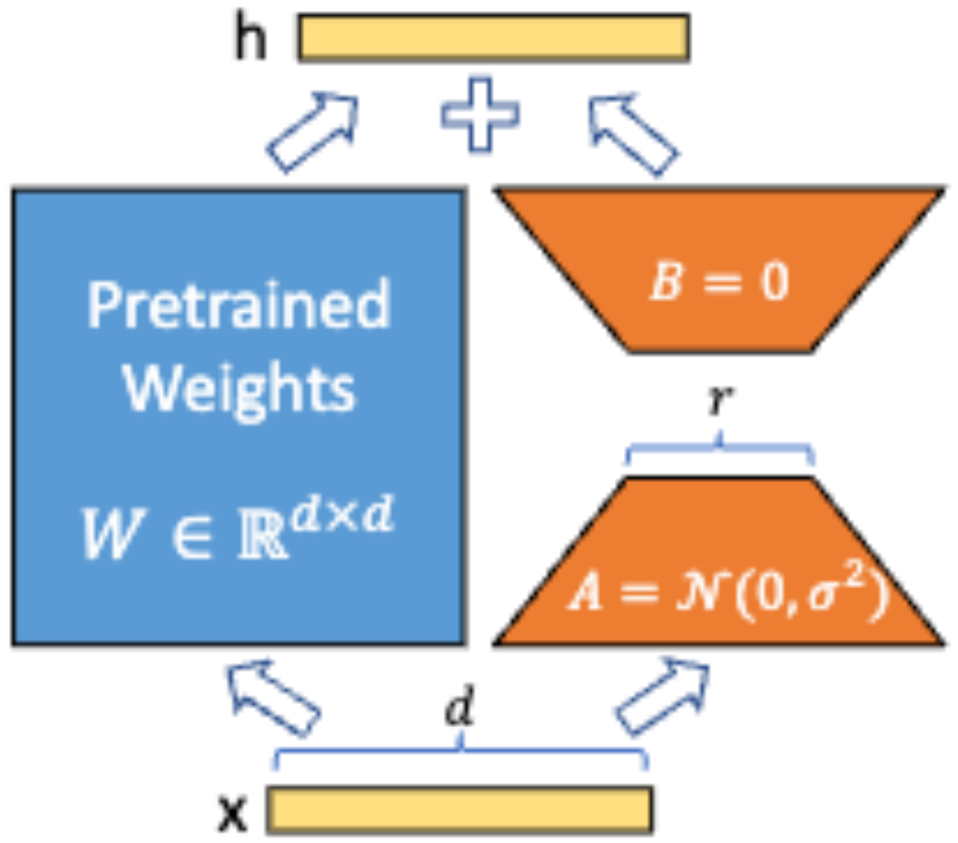
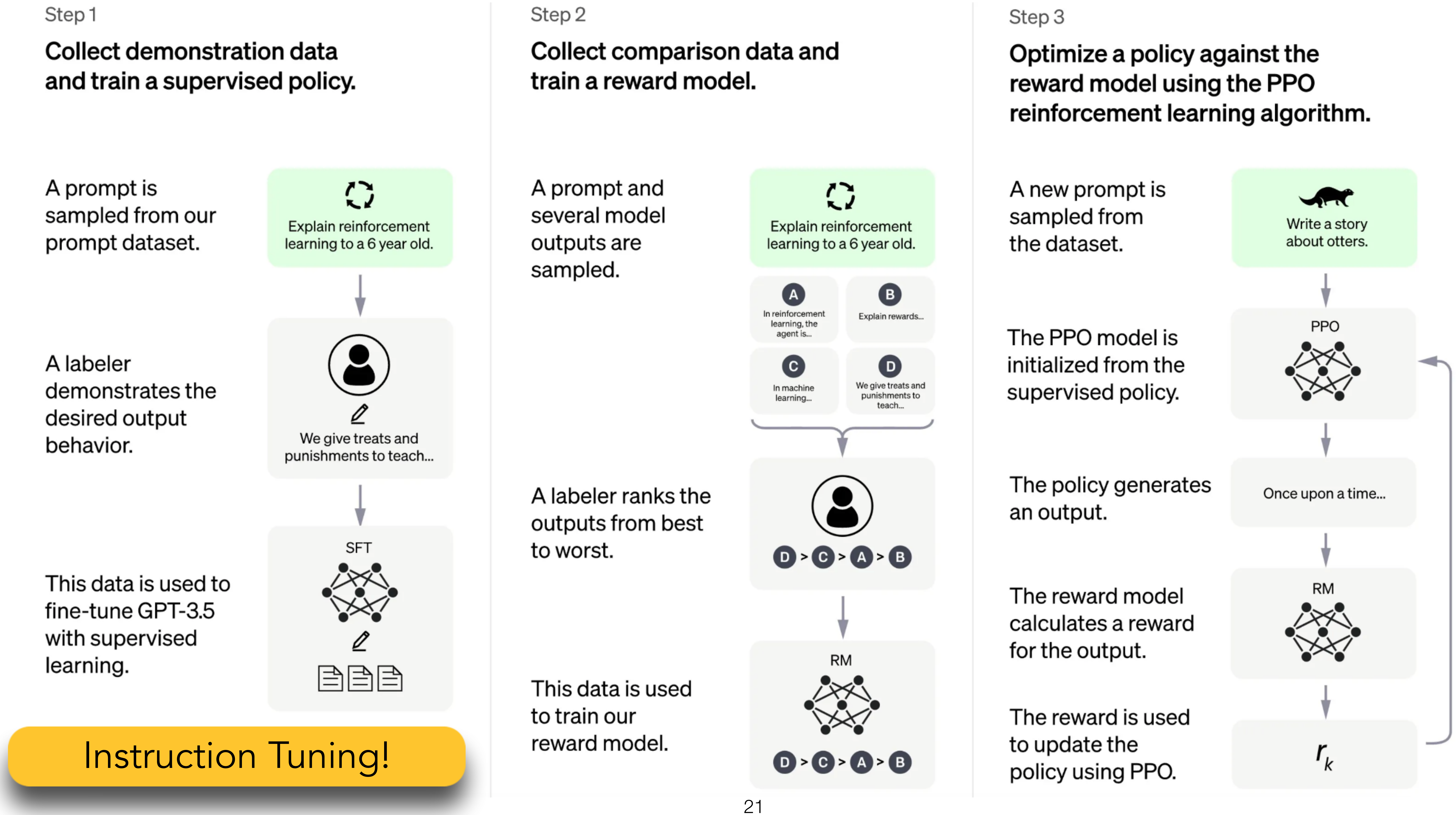


Figure 1: Our reparametrization. We only train A and B.

LoRA. Hu et al., 2021



Step 1

Collect demonstration data and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3.5 with supervised learning.

Explain reinforcement learning to a 6 year old.
We give treats and punishments to teach...
SFT

Step 2

Collect comparison data and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.

Explain reinforcement learning to a 6 year old.
A
In reinforcement learning, the agent is...
B
Explain rewards...
C
In machine learning...
D
We give treats and punishments to teach...
D > C > A > B
RM
D > C > A > B

Step 3

Optimize a policy against the reward model using the PPO reinforcement learning algorithm.

A new prompt is sampled from the dataset.

The PPO model is initialized from the supervised policy.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.

Instruction Tuning!

Evaluation

GSM8K

MMLU

BigBench

HumanEval

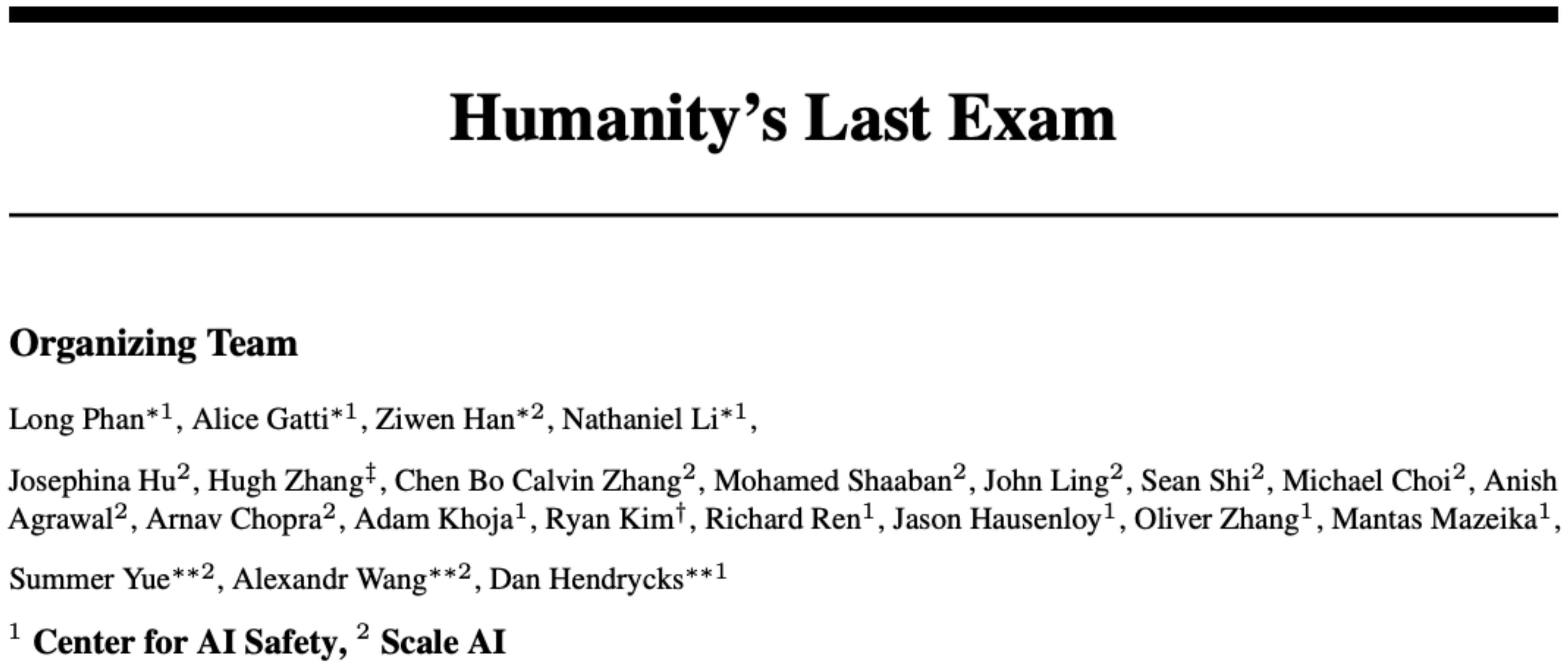
GPQA

Math

HELM

Alpaca2.0

OpenLLMLeaderboard



Humanity's Last Exam

Organizing Team

Long Phan[*1], Alice Gatti[*1], Ziwen Han[*2], Nathaniel Li[*1],

Josephina Hu[2], Hugh Zhang[‡], Chen Bo Calvin Zhang[2], Mohamed Shaaban[2], John Ling[2], Sean Shi[2], Michael Choi[2], Anish Agrawal[2], Arnav Chopra[2], Adam Khoja[1], Ryan Kim[†], Richard Ren[1], Jason Hausenloy[1], Oliver Zhang[1], Mantas Mazeika[1],

Summer Yue[**2], Alexandr Wang[**2], Dan Hendrycks[**1]

[1] **Center for AI Safety,** [2] **Scale AI**

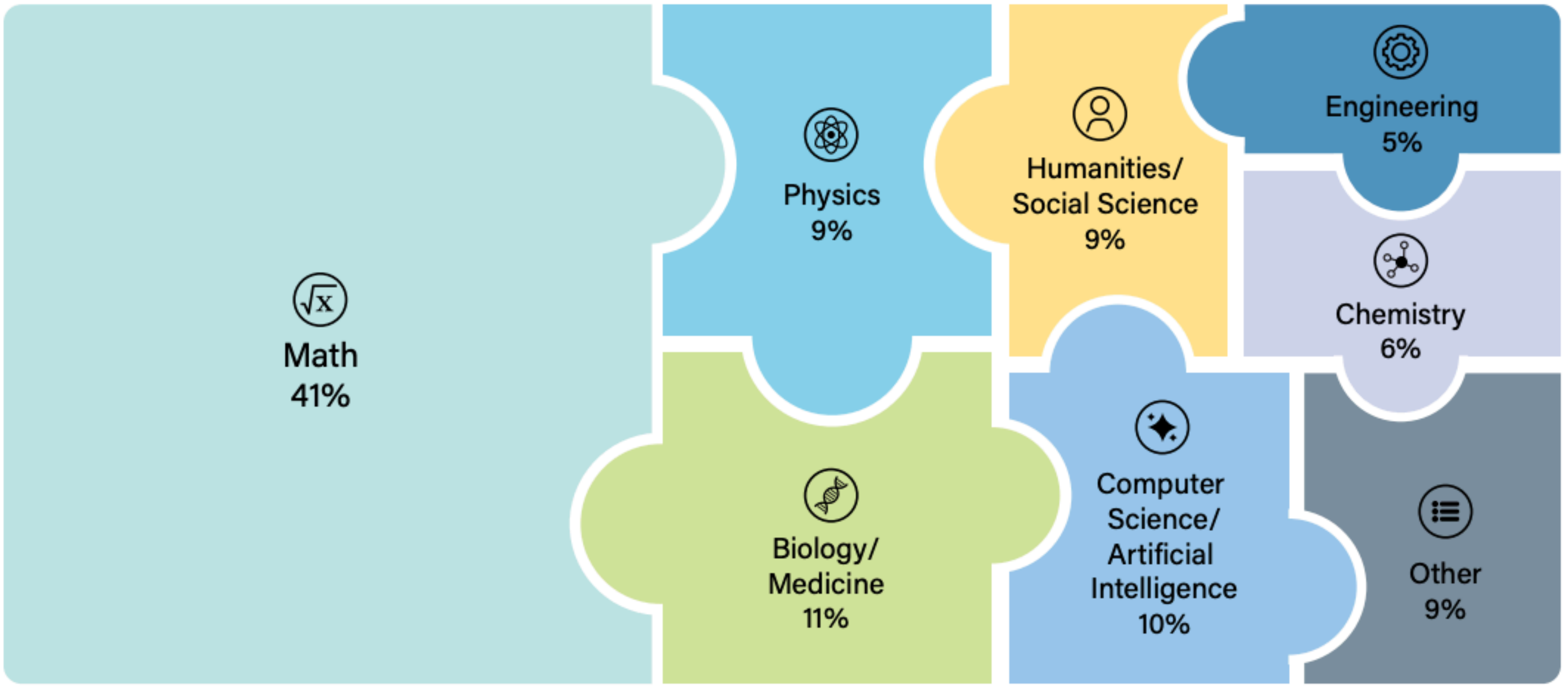
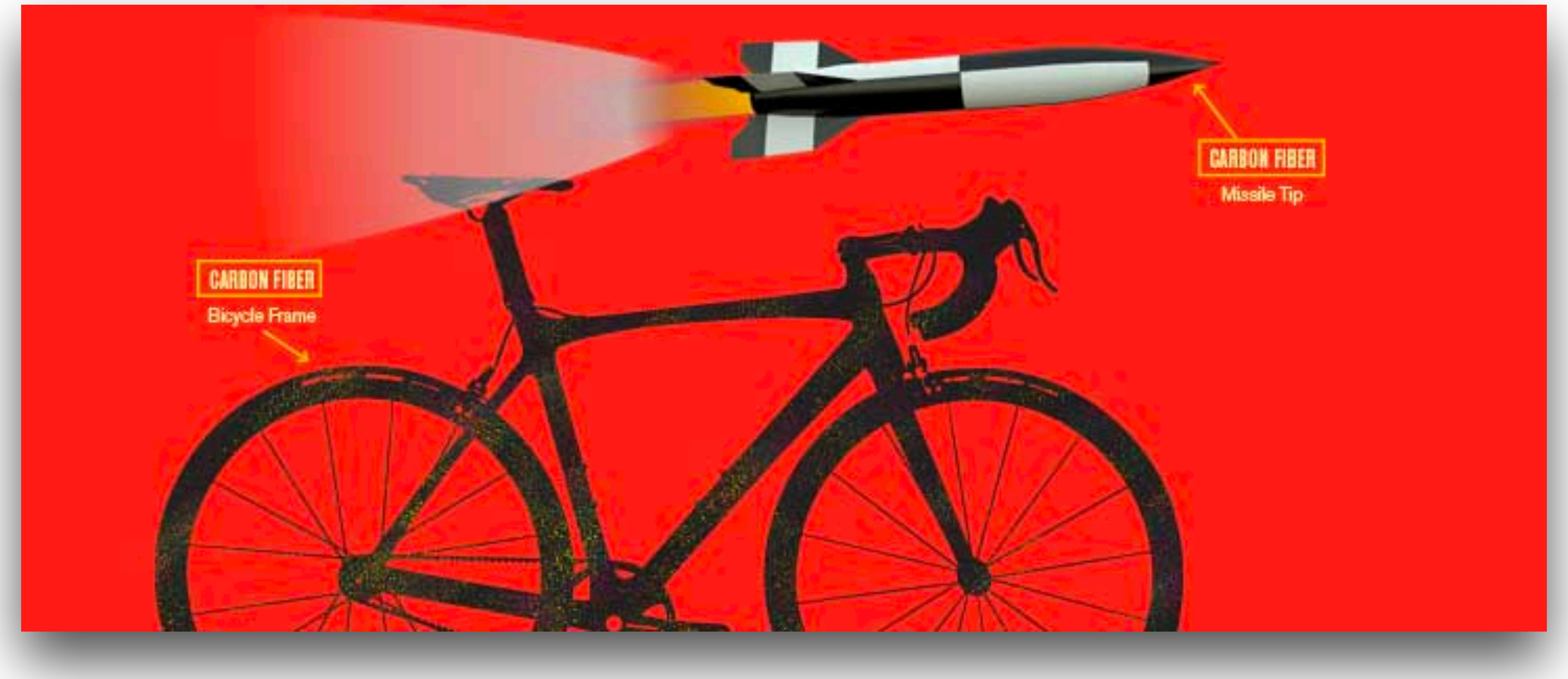


Figure 3: HLE consists of 2,700 exam questions in over a hundred subjects, grouped into high level categories here. We provide a more detailed list of subjects in Appendix B.3.

Responsible Language Modeling

- Category 1: Allocational and Representational Harms
  - Performance Disparities
  - Social biases and Stereotypes
- Category 2: Behavioral Harms
  - Hallucinations, Misinformation and Misguiding
  - Toxicity of Generated Content
  - Emotional Dependence, Cognitive Decline
- Category 3: Security and Privacy risks
  - Copyright and legal protections
- Category 4: Environmental Impact
- Category 5: Centralization of Power
  - Access due to high costs
  - Only a few key players can build LLMs



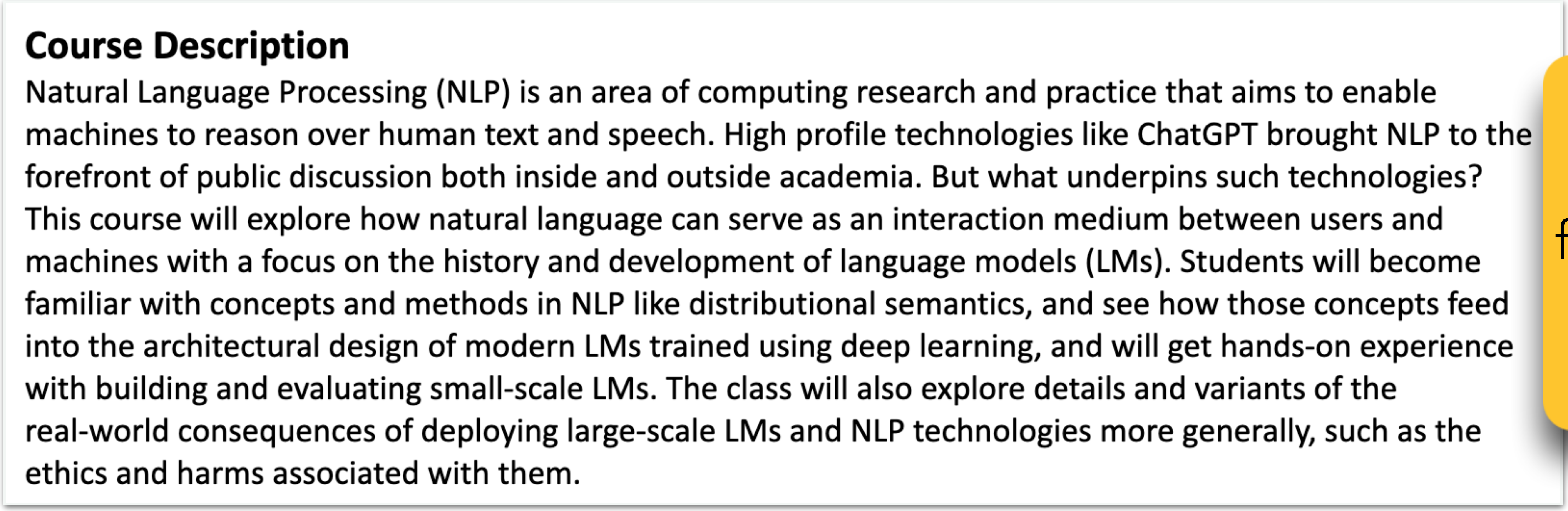
Benefits versus harms. With any technology, it's important to consider the tradeoff between benefits and harms

Quiz 5:
Password = alignment

Bonus Quiz

Password = bonus

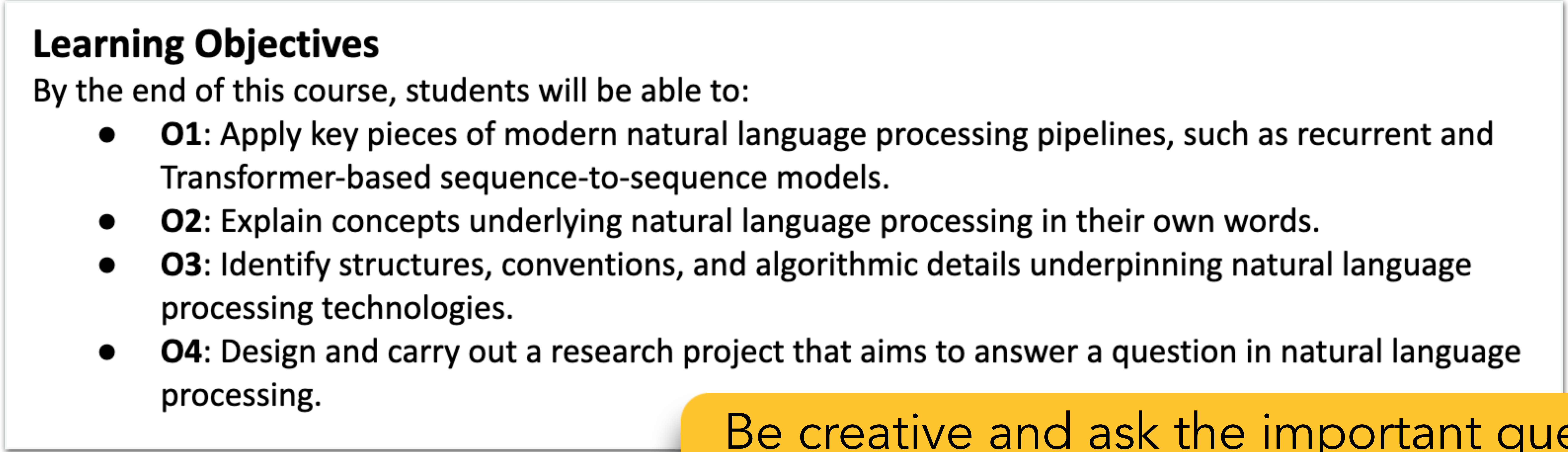
To wrap it up…



Course Description

Natural Language Processing (NLP) is an area of computing research and practice that aims to enable machines to reason over human text and speech. High profile technologies like ChatGPT brought NLP to the forefront of public discussion both inside and outside academia. But what underpins such technologies? This course will explore how natural language can serve as an interaction medium between users and machines with a focus on the history and development of language models (LMs). Students will become familiar with concepts and methods in NLP like distributional semantics, and see how those concepts feed into the architectural design of modern LMs trained using deep learning, and will get hands-on experience with building and evaluating small-scale LMs. The class will also explore details and variants of the real-world consequences of deploying large-scale LMs and NLP technologies more generally, such as the ethics and harms associated with them.

Language Models are a fundamental and foundational technology, and here to stay for a long long time!



Learning Objectives

By the end of this course, students will be able to:

- **O1**: Apply key pieces of modern natural language processing pipelines, such as recurrent and Transformer-based sequence-to-sequence models.
- **O2**: Explain concepts underlying natural language processing in their own words.
- **O3**: Identify structures, conventions, and algorithmic details underpinning natural language processing technologies.
- **O4**: Design and carry out a research project that aims to answer a question in natural language processing.

Be creative and ask the important questions as you use this technology

Other Resources

- Spring 2025 classes at USC
  - CSCI 544 by Robin Jia / Xuezhe Ma - Applied NLP
- Other Institutes
  - ETH Zürich - Large Language Models: https://rycolab.io/classes/llm-s23/
  - Stanford - Large Language Models: https://stanford-cs324.github.io/winter2022/
- Constantly evolving field
  - Keep up via Twitter, YouTube and other social media (but be cautious!)
  - e.g. Very accessible LM tutorial: https://www.youtube.com/watch?v=k9DnQPrfJQs&ab_channel=HarvardDataScienceInitiative

Thank You!

Go forth and generate…