

Natural Language Processing

Week 8

Agenda

- Text Generation
- Decoder-Only Models
- Encoder-Decoder Models

Text Generation

Text Generation

- Text generation is the task of generating new text given some starting text
- Covers **text-generation** (decoder-only) and **text-to-text** (encoder-decoder) models
- Text Generation
 - Excel at predicting the next tokens given only some start or seed tokens
 - Popular models: GPT, Llama, Mistral
- Text-to-Text Generation
 - Trained to map sequences of text to other sequences
 - Useful for translation, summarization, text-based question answering
 - Popular Models: Flan-T5, BART

Language Model Variants

- Base Models
 - Also called **foundation models**
 - Pre-trained on enormous corpus of text as a starting point for more specialized tasks (encompasses generative and non-generative models)
- Instruction-Tuned Models
 - Models that receive further training to follow complex and varied instruction
 - Allows the models to perform better on tasks that involve user-generated prompts calling for some action
- Human Feedback Models:
 - Models that receive further training to improve **alignment** with human preferences (honesty, brevity, helpfulness, etc.)

Decoder-Only Models

Text Generation

Prompt (Human-written)

In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

Completion
(Machine-generated, GPT 2)


The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science.

Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved.

Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow.

Text Generation

- Recall that the task of text generation is called **autoregressive / causal language modeling**
- Language models give us the ability to **assign conditional probability to every possible next word**, giving us a distribution over the entire vocabulary


$$P(w_{1:T} | W_0) = \prod_{t=1}^T P(w_t | w_{1:t-1}, W_0), \text{ where } w_{1:0} = \emptyset$$

“The probability of a sequence of words, 1 through T, given some starting word(s) W_0 , is equal to the product of probabilities”

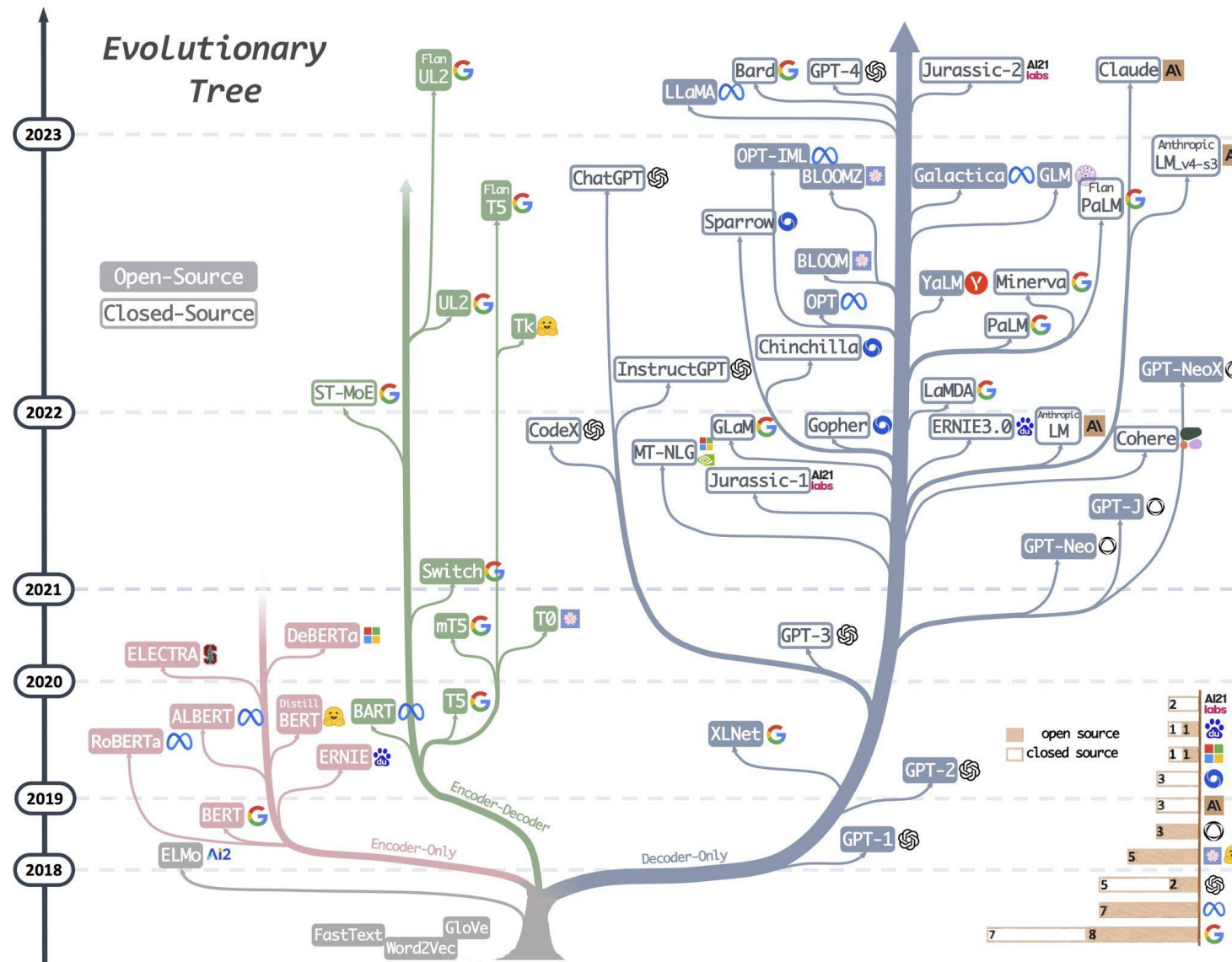


model vocabulary size
50,257



0.19850038	aardvark
0.7089803	aarhus
0.46333563	aaron
	...
	...
	...
	...
	...
-0.51006055	zyzzyva

Remember the Transformer Family Tree



Why are decoder-only models so much more popular?

- **Good zero-shot performance for various tasks**
- Zero-shot is a model's capability to generalize to **new tasks** without having been trained on that specific task (summarization, question-answering, classification, etc) during training

$P(\text{"positive"} \mid \text{"The sentiment of the sentence 'This pizza is fantastic!' is"})$

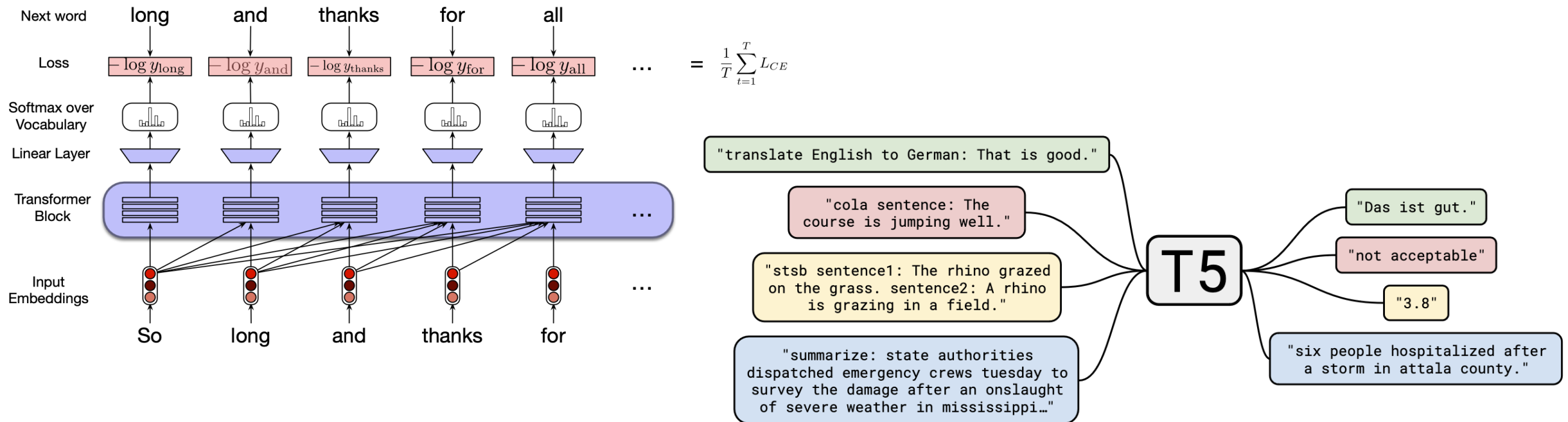
$>$

$P(\text{"negative"} \mid \text{"The sentiment of the sentence 'This pizza is fantastic!' is"})$

- During pre-training, language models are exposed to tasks where they need to predict the following tokens based on context

Why are decoder-only models so much more popular?

- **Ease of obtaining training data**
- Decoder-only models leverage self-supervised learning
- There are no explicit (human) labels, **the natural sequence of words serve as their own labels**



Training Decoder-Only Models

The University of Chicago is a private research university in Chicago, Illinois. The university has its main campus in Chicago's Hyde Park neighborhood.

Training data

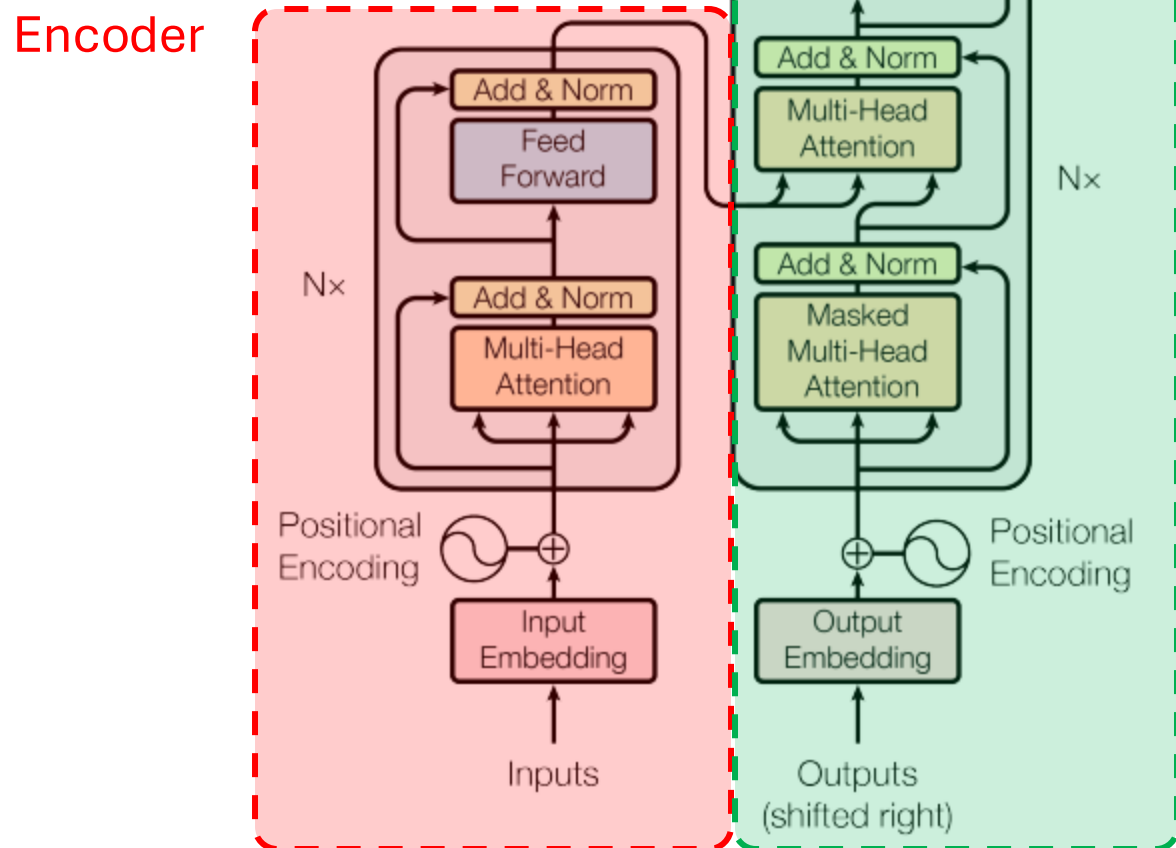
Row #1	The
Row #2	The University
Row #3	The University of
Row #4	The University of Chicago
Row #5	The University of Chicago is
Row #6	The University of Chicago is a
Row #7	The University of Chicago is a private
Row #8	The University of Chicago is a private research
Row #9	The University of Chicago is a private research university
Row #10	The University of Chicago is a private research university in
Row #11	The University of Chicago is a private research university in Chicago
Row #12	The University of Chicago is a private research university in Chicago ,
Row #13	The University of Chicago is a private research university in Chicago , Illinois
Row #14	The University of Chicago is a private research university in Chicago , Illinois .

Label

University
of
Chicago
is
a
private
research
university
in
Chicago
,
Illinois
.
<EOS>

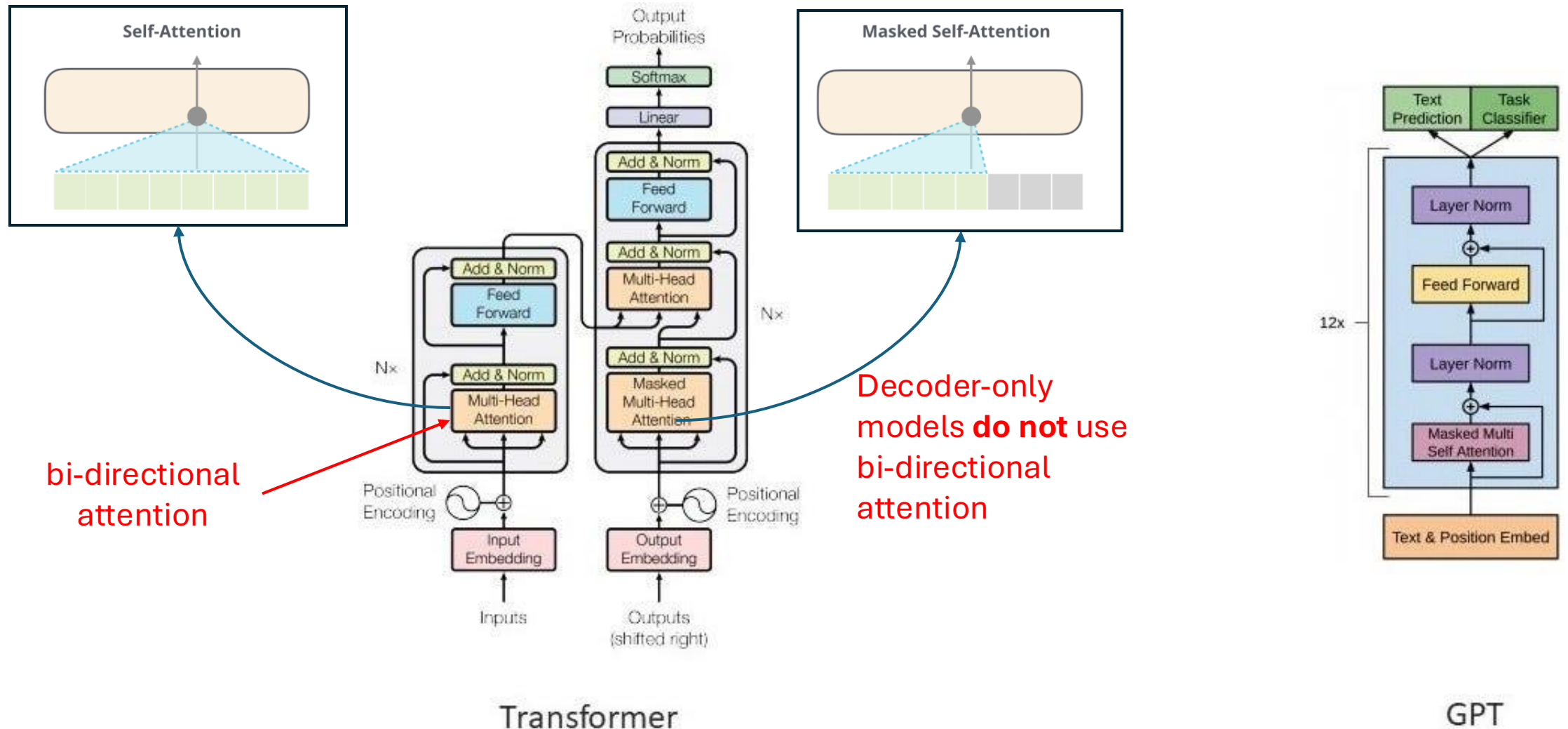
Max Context window: 1024

Original Transformer
model used an encoder-
decoder architecture



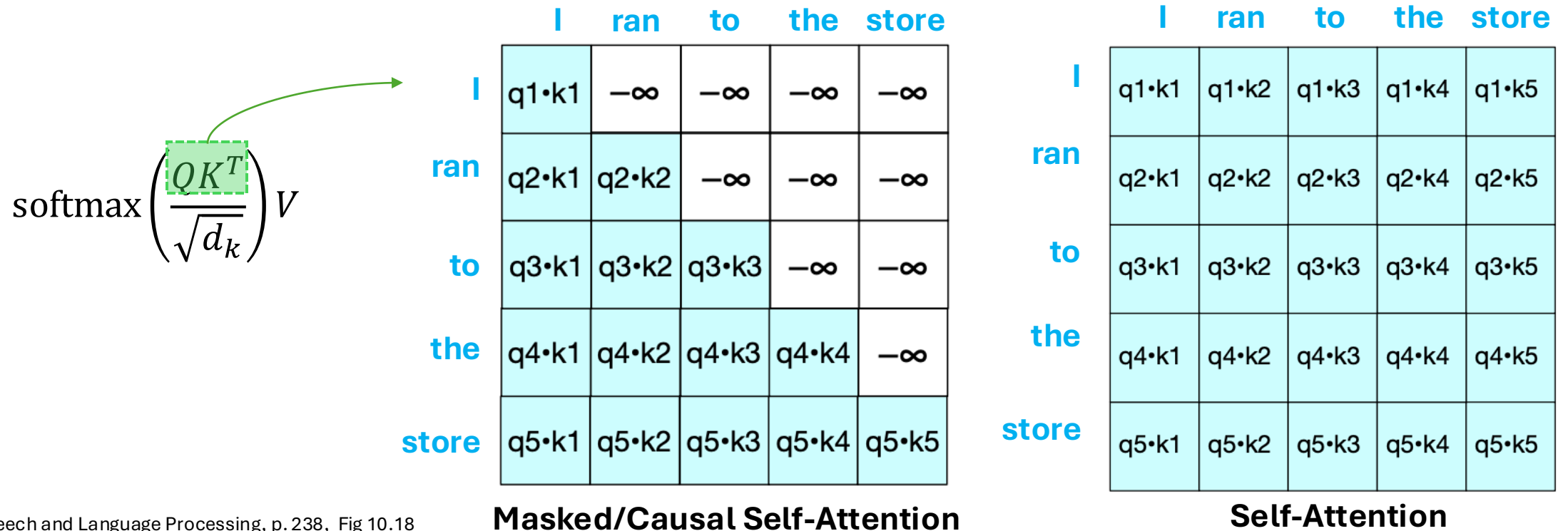
Decoder

Differentiating Encoder vs Decoder Attention



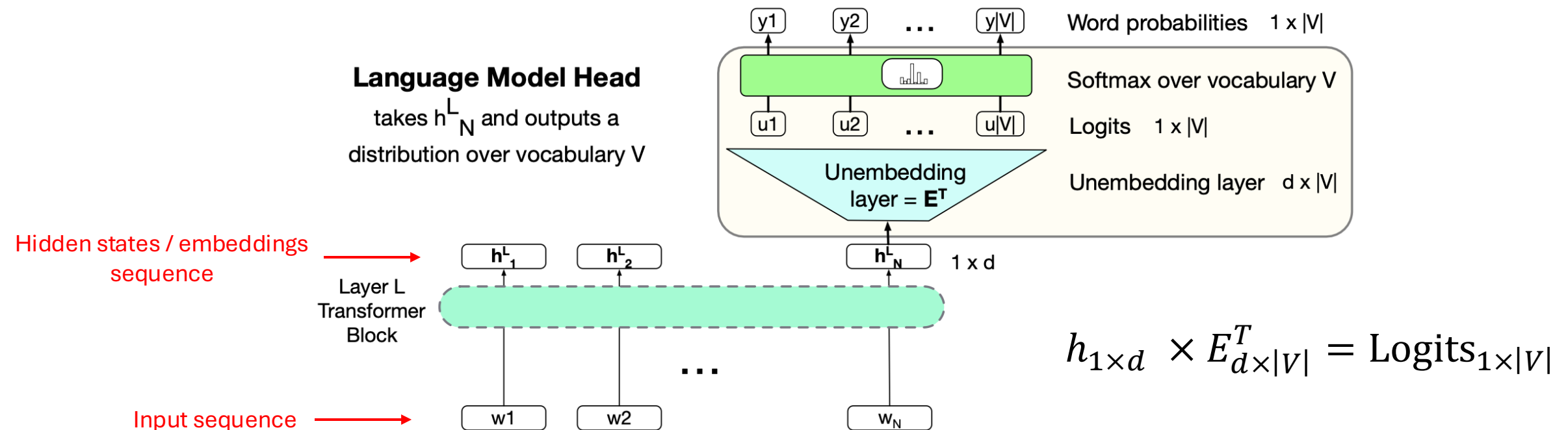
Masked Self-Attention

- Given the task of text generation (predicting the next word / token), we must modify the self-attention mechanism **to avoid leaking information about future tokens**
- Reminder:** Self-attention is the mechanism in transformer models that allows each part of an input sequence to consider / weigh every other part



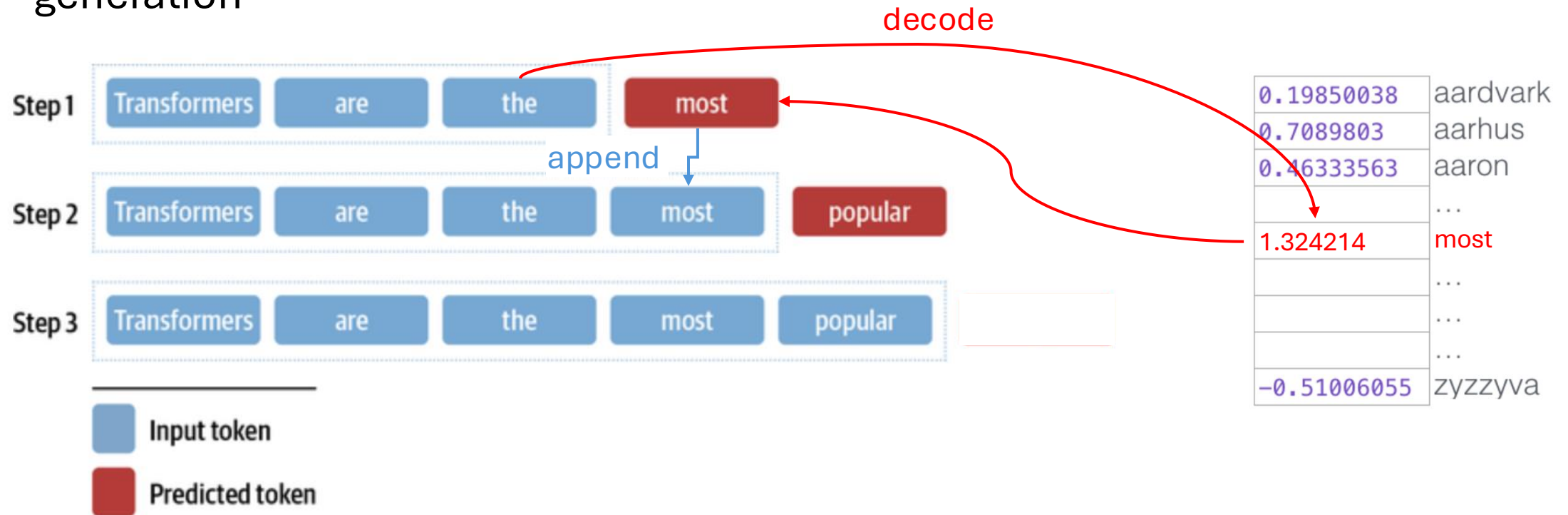
Language Modeling Head

- The job of the language modeling head is to use the final embedding of the last token in a sequence and use it to predict the next token.
- Uses a linear layer to take token embedding and produce probability distribution over the vocabulary



Decoding Methods for Inferencing

- The process of selecting the next word given a model's probabilistic output is called **decoding**
- Decoding is done **iteratively**
- The choice of decoding method dictates the **quality** and **diversity** of text generation



Greedy Search Decoding

- The simplest method is to select that the most probable token for each timestep

$$\hat{y}_t = \underset{y_t}{\operatorname{argmax}} P(y_t | y_{<t}, x)$$

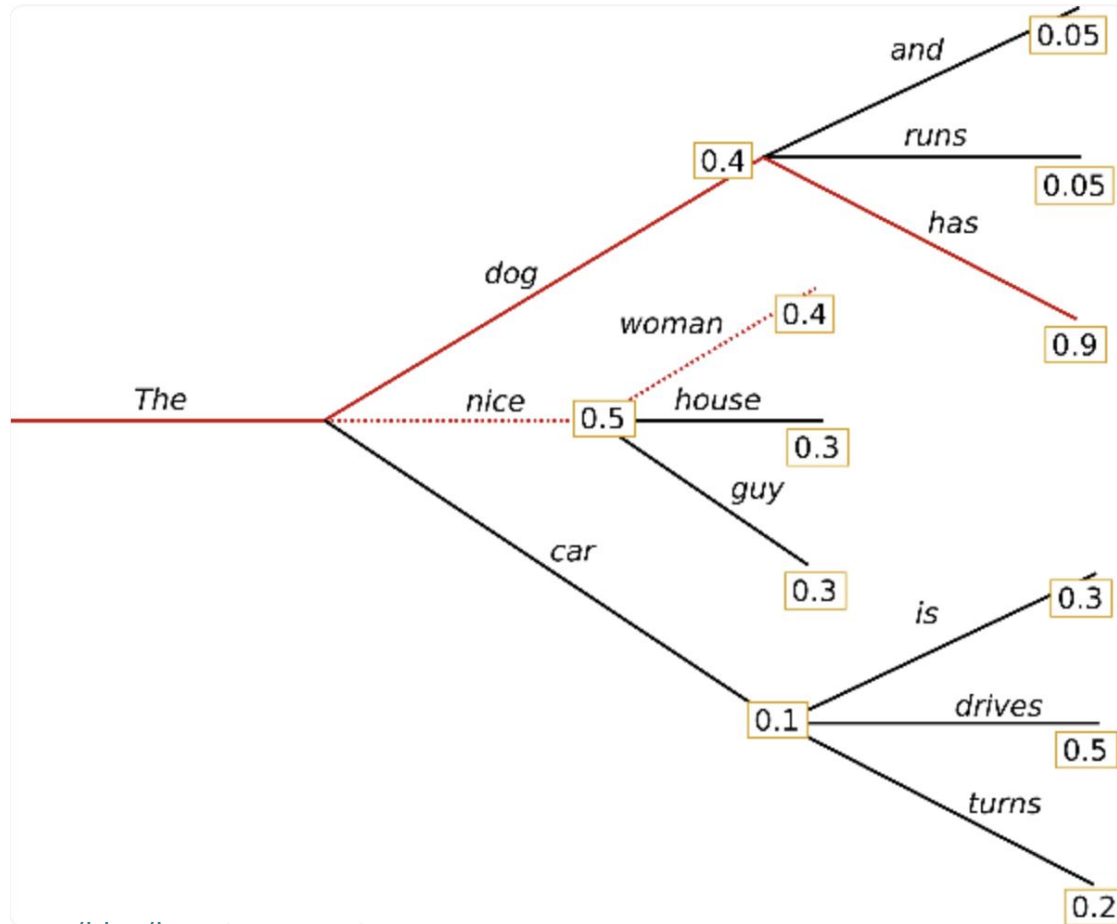
- Greedy search decoding tends to produce **repetitive text**
- They fail to give optimal solution because they can miss sequences **whose overall probability is higher** because high-probability words can follow low-probability words

Always
choosing this

Input	Choice 1	Choice 2	Choice 3	Choice 4	Choice 5
Transformers are	the (4.67%)	not (2.78%)	all (2.36%)	in (2.23%)	are (1.98%)
Transformers are the	most (5.00%)	the (2.37%)	are (1.86%)	best (1.38%)	ones (1.04%)
Transformers are the most	popular (13.99%)	important (4.10%)	of (3.41%)	effective (2.63%)	the (1.90%)
Transformers are the most popular	and (5.87%)	among (2.50%)	the (1.67%)	in (1.61%)	ly (1.53%)
Transformers are the most popular and	the (19.02%)	most (5.90%)	are (4.40%)	all (1.97%)	is (1.92%)
Transformers are the most popular and the	most (6.10%)	the (3.96%)	are (2.33%)	and (1.56%)	best (1.50%)
Transformers are the most popular and the most	popular (24.01%)	important (4.88%)	of (4.73%)	effective (1.69%)	likely (1.53%)
Transformers are the most popular and the most...	and (12.17%)	among (2.51%)	, (2.10%)	to (1.95%)	the (1.78%)

Beam Search Decoding

- Instead of decoding the token with the highest probability at each step, beam search keeps track of a predefined number of the best partial solutions (beams) at each step



- **Can lead to finding sequences with higher overall probability**
- More beams = slower generation
- Still suffers from repetition
- n-gram penalty: keeps track of n-grams seen and sets next token probability to zero if it would produce a previously seen n-gram

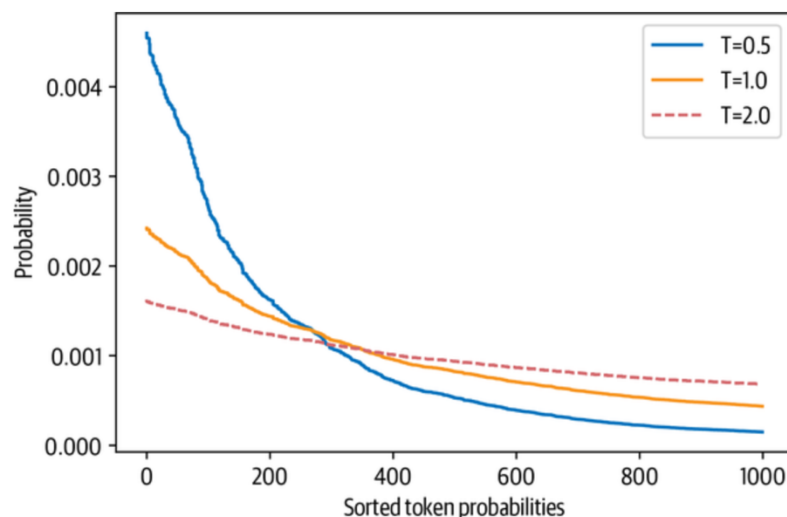
Overcoming Repetition - Sampling

- Instead of choosing tokens with highest probability, randomly sample from the probability distribution of the model's outputs

$$P(y_t = w_i | y_{<t}, x) = \text{softmax}(z_{t,i}) = \frac{\exp(z_{t,i})}{\sum_j^{|V|} \exp(z_{t,j})}$$

- We can modify the distribution by rescaling it by a temperature parameter, T

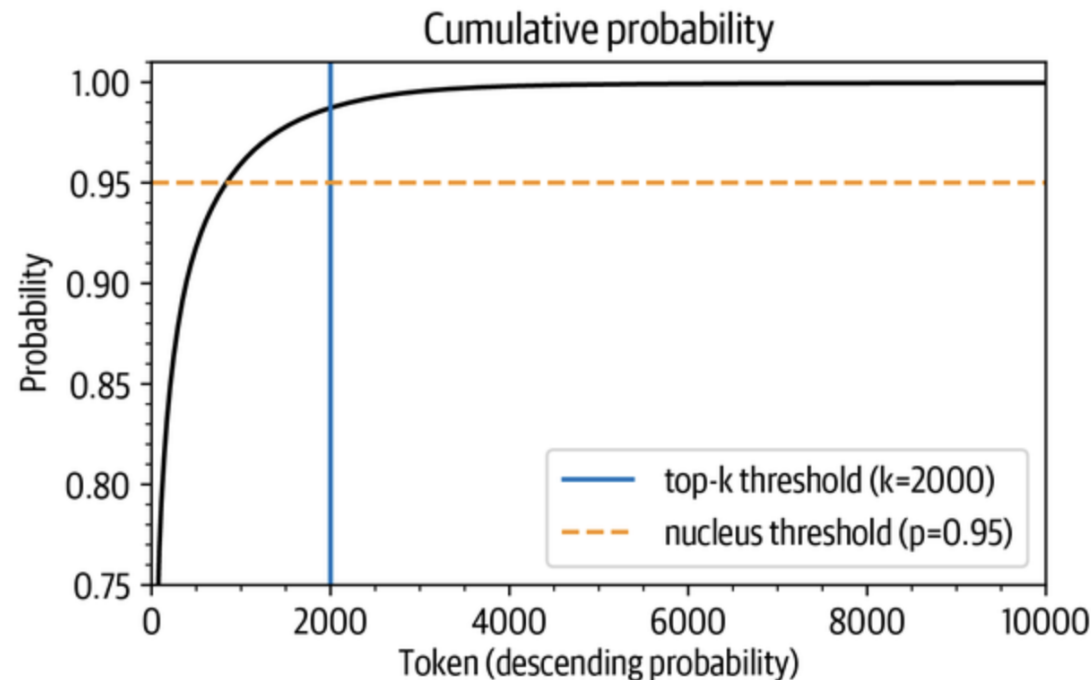
$$P(y_t = w_i | y_{<t}, x) = \text{softmax}(z_{t,i}) = \frac{\exp(z_{t,i}/T)}{\sum_j^{|V|} \exp(z_{t,j}/T)}$$



- When $T \gg 1$, accentuates the tail (rare tokens)
- When $T \ll 1$, suppresses the tail
- Tradeoff between coherence (low T) and diversity (high T)

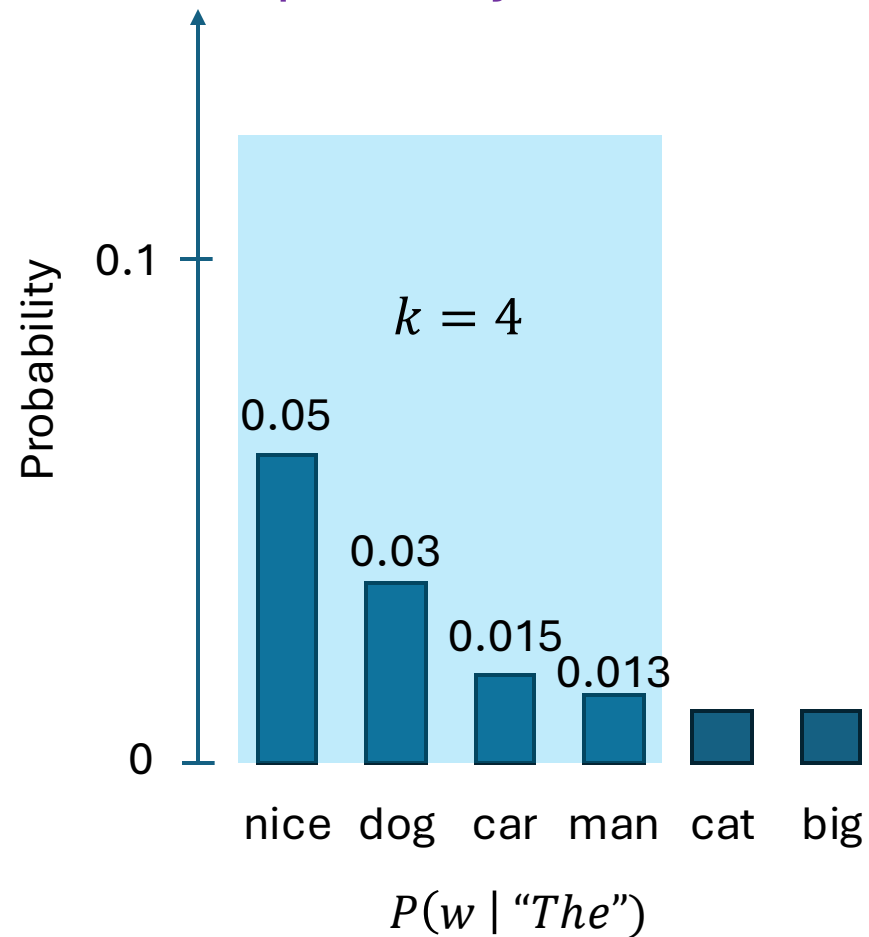
Overcoming Repetition – Top- k and Top- p (Nucleus Sampling)

- Instead of sampling from the entire vocabulary, we can limit the model to select from some number of candidate tokens
- **Top- k :** selects as candidates the top k most probably tokens
- **Top- p :** selects as candidates the tokens whose cumulative probability reaches a threshold, p

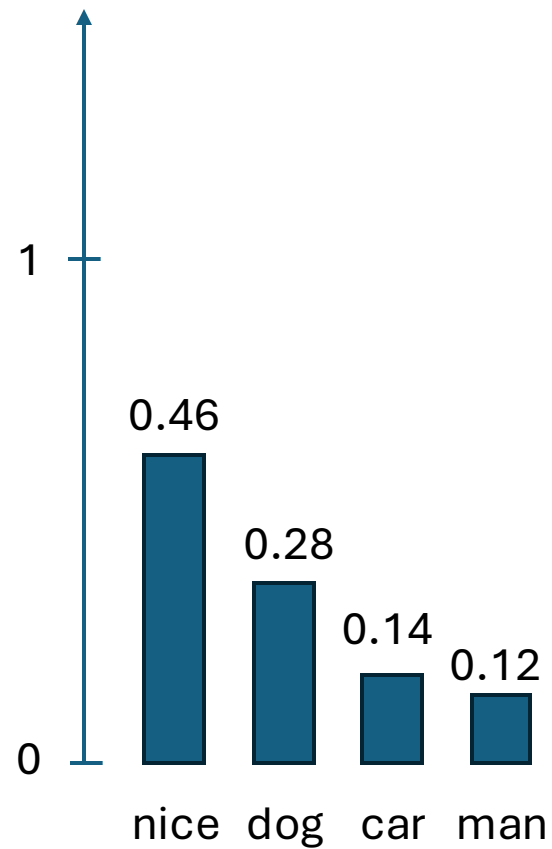


Overcoming Repetition – Top- k

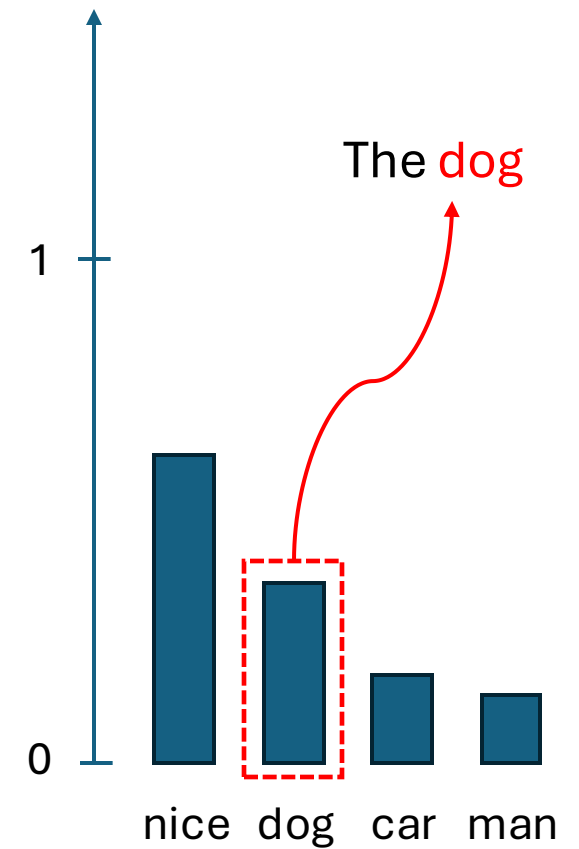
1. Select the top- k most probably tokens



2. Renormalize

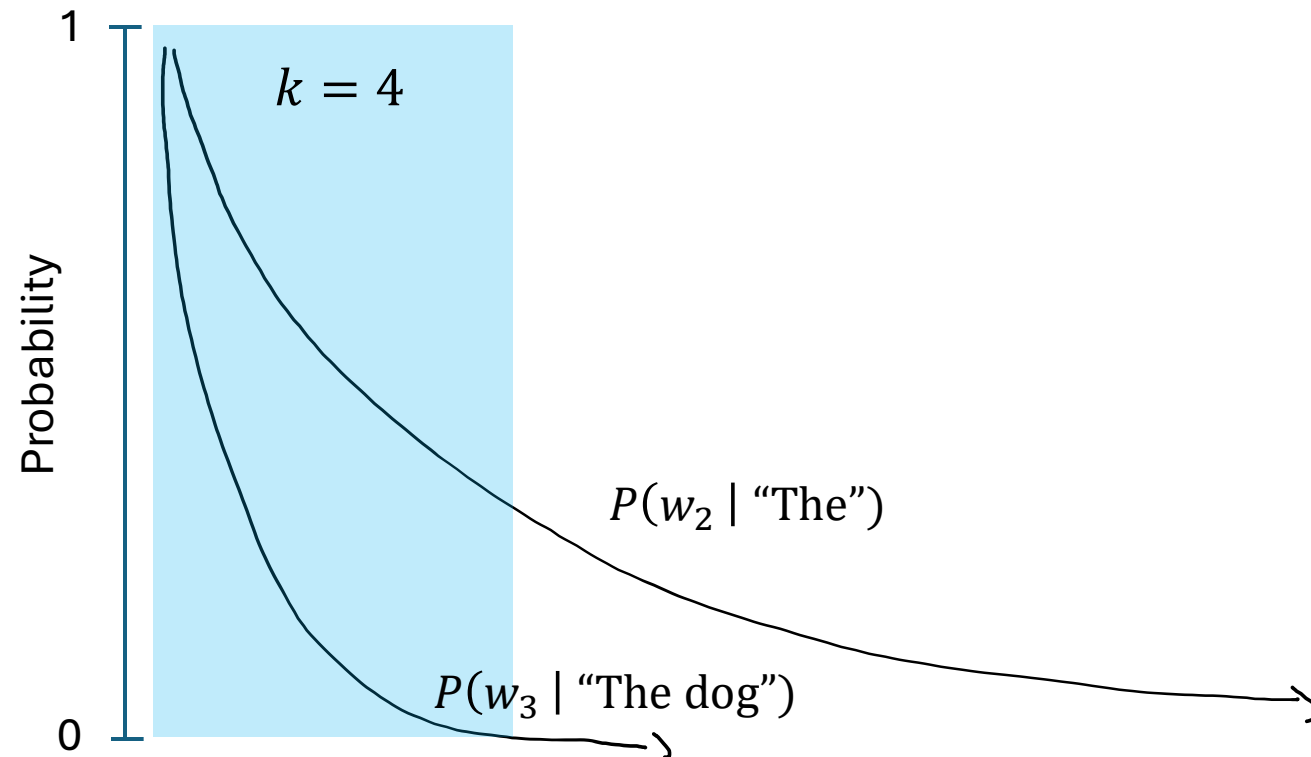


3. Sample



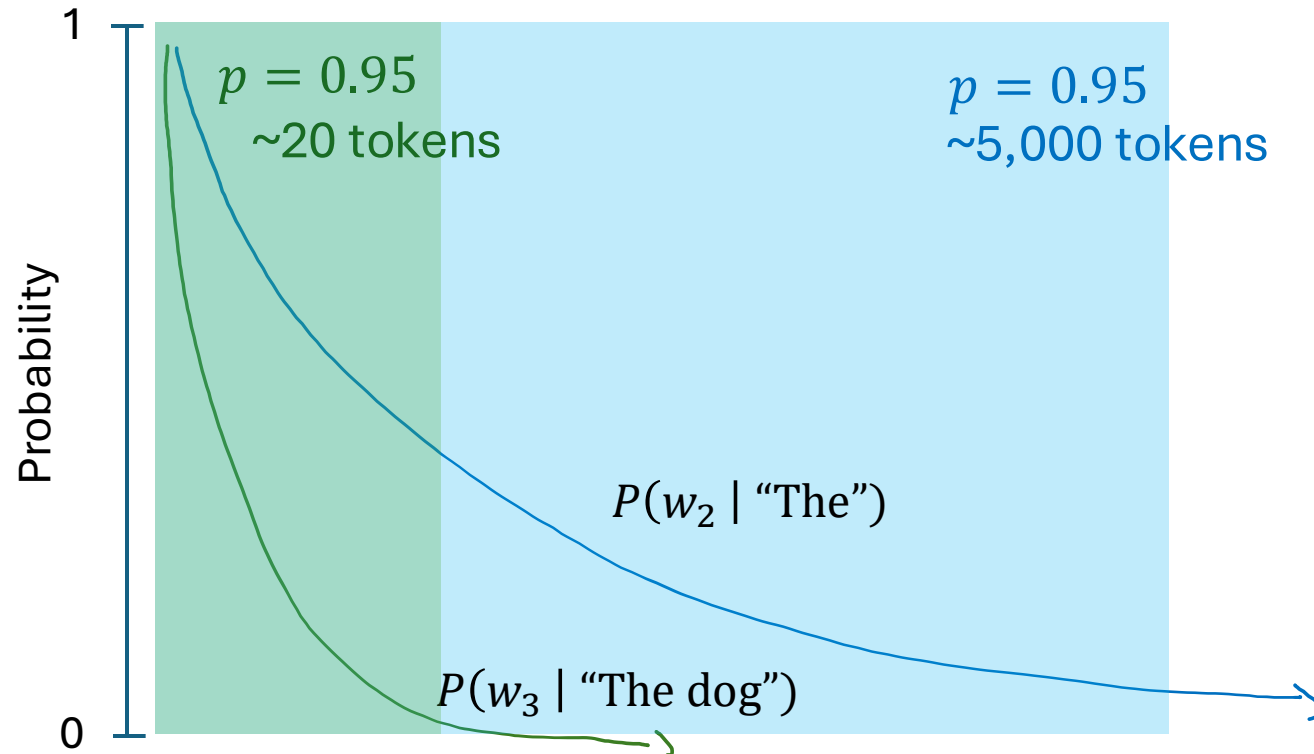
Overcoming Repetition – Top- k

- **Top- k always selects k words at each iteration, regardless of the word distribution**
- If the distribution is steep, that could mean including highly unlikely tokens as candidates



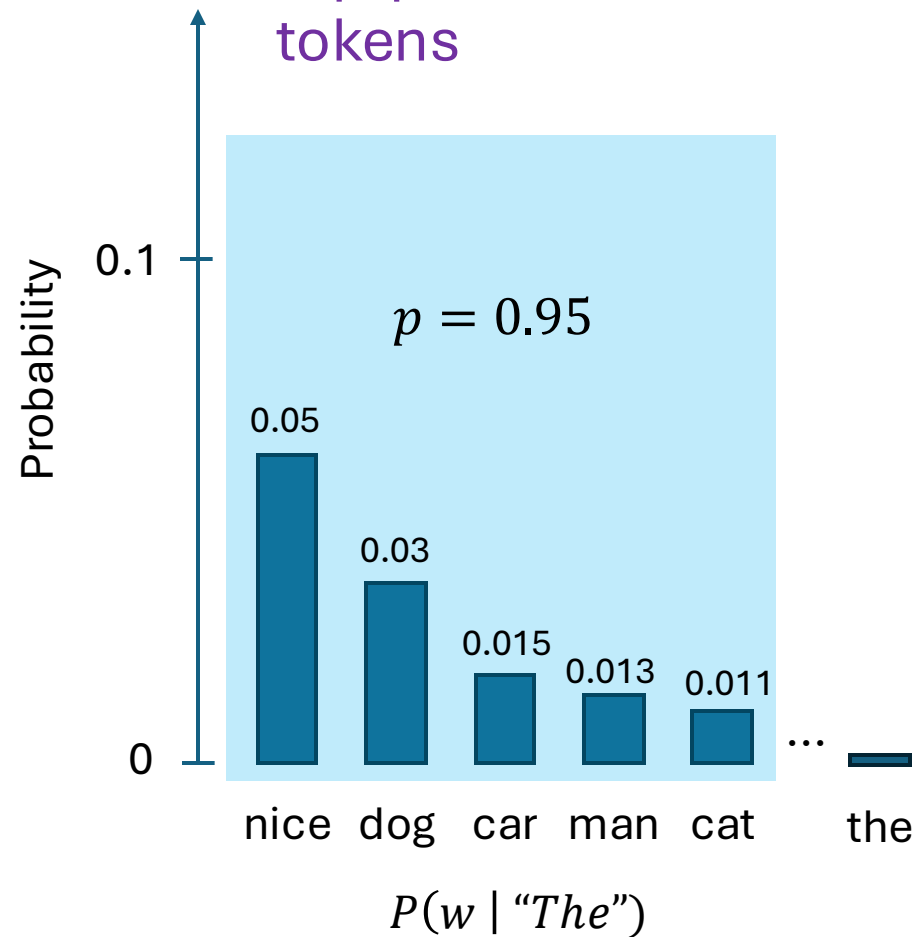
Overcoming Repetition – Top- p

- **Top- p chooses from the smallest possible set of words whose cumulative probability exceeds some probability p**

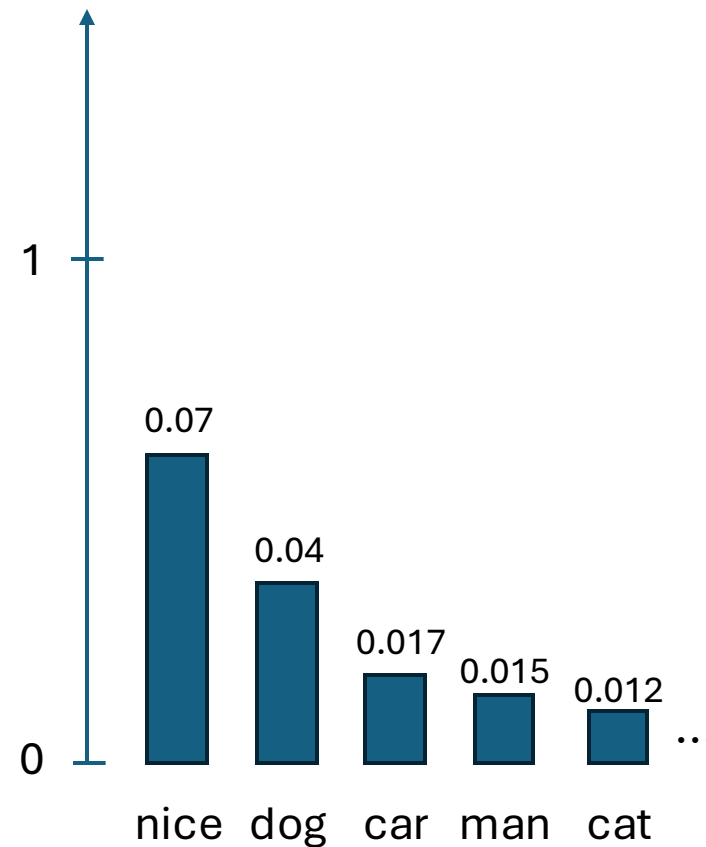


Overcoming Repetition – Top- p

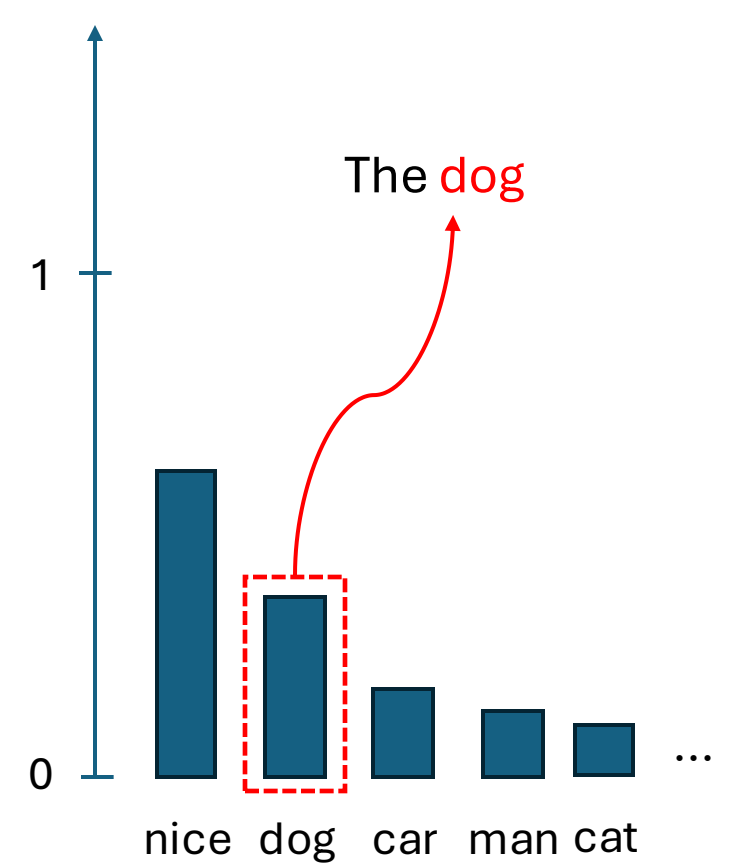
1. Select tokens within top- p of distribution tokens



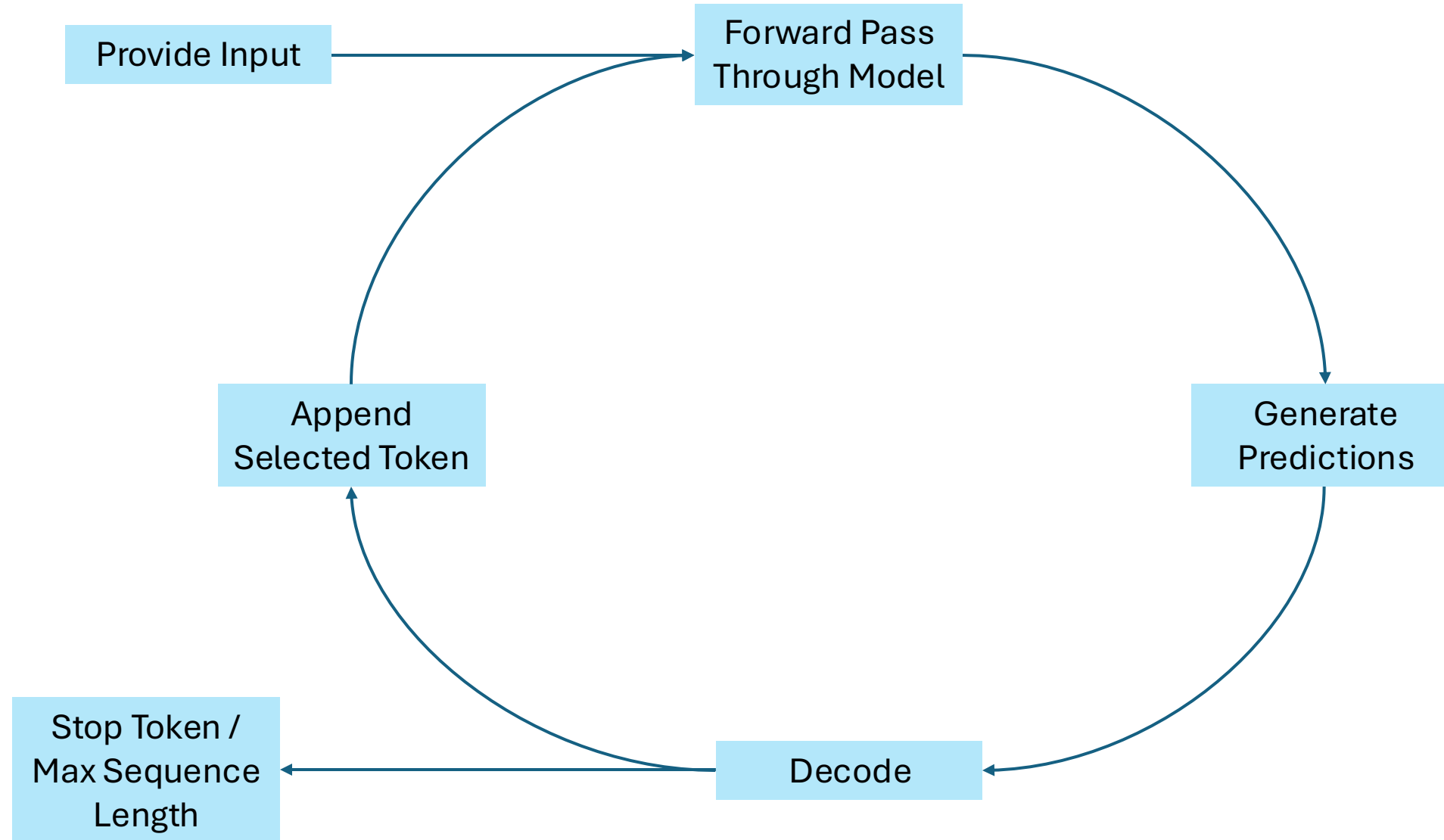
2. Renormalize



3. Sample

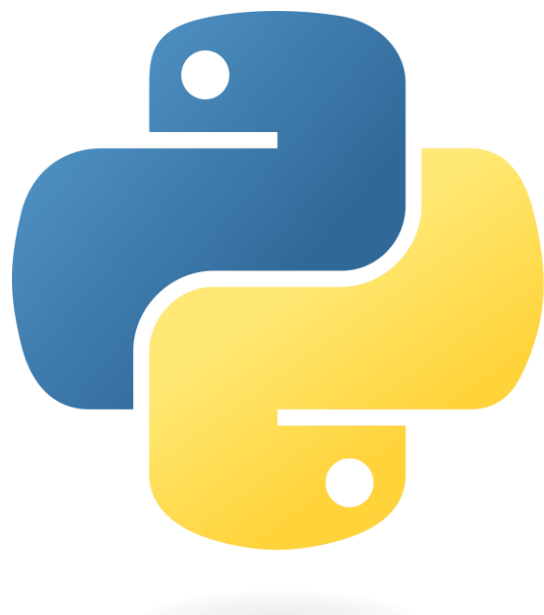


Text Generation Loop



Which decoding method is best?

- **As always, depends**
- If precision and factuality is important...
 - Lower the temperature
 - Use greedy / beam search to get most likely answer
- If creativity and longer texts are important
 - Increase the temperature
 - Use sampling methods (top k, top p)



Generative Pre-Trained Transformer (GPT)

History

GPT-2

- February 2019
- 10X training data of GPT-1, reaching 40GB of text
- 1.5B parameters
- **Proved that larger language model can perform better**

InstructGPT

- Mar 2022
- Added human label data (ranking) for supervised fine-tuning (SFT)
- 1.3B parameters
- Fine-tuned GPT-3 using SFT and RLHF, resulted **better alignment**

GPT-4

- Mar 2023
- OpenAI reveals little information
- First multimodal model, better reasoning

GPT-1

- June 2018
- Trained by 7,000 unpublished books ([BookCorpus](#) dataset)
- 117M parameters
- Overcame the manually labeling problem, introducing unsupervised pre-training step

GPT-3

- June 2020
- More data including public web archive data
<https://commoncrawl.org/>
- 175B parameters
- **Demonstrated superior coherence and creativity, but has misalignment issue**

GPT-3.5 series

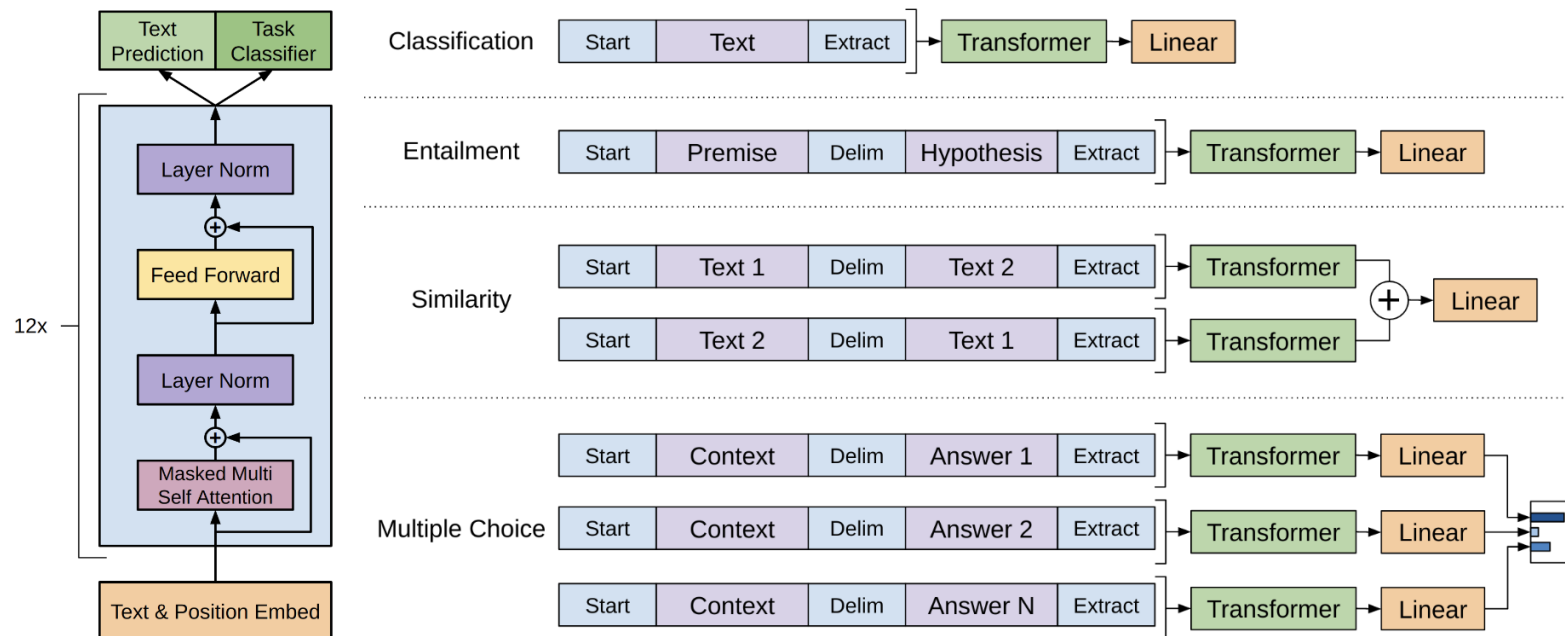
(Codex* and ChatGPT)

- March 2022
- Combined Instructions into GPT-3
- 175B parameters
- ChatGPT has been fine-tuned to excel at interactive dialogue

* Codex model was deprecated by OpenAI in March 2023. OpenAI recommends that users switch from Codex to GPT-3.5 Turbo or GPT-4. At the same time, GitHub released Copilot X, which is based on GPT-4

GPT-1

- [Improve Language Understanding by Generative Pre-Training](#) (2018)
- The paper notes:
 - The use of **language modeling objective** (self-supervised) on a large corpus of unlabeled data to learn the initial model parameters
 - Followed experiments done by **discriminative fine-tuning** (supervised) on specific NLP tasks



(left) Transformer architecture and training objectives used in this work.

(right) Input transformations for fine-tuning on different tasks. We convert all structured inputs into token sequences to be processed by our pre-trained model, followed by a linear + softmax layer.

GPT-1

- Model was able to perform better than models **specifically designed for that task**

Natural Language Inference

Method	MNLI-m	MNLI-mm	SNLI	SciTail	QNLI	RTE
ESIM + ELMo [44] (5x)	-	-	<u>89.3</u>	-	-	-
CAFE [58] (5x)	80.2	79.0	<u>89.3</u>	-	-	-
Stochastic Answer Network [35] (3x)	<u>80.6</u>	<u>80.1</u>	-	-	-	-
CAFE [58]	78.7	77.9	88.5	<u>83.3</u>		
GenSen [64]	71.4	71.3	-	-	<u>82.3</u>	59.2
Multi-task BiLSTM + Attn [64]	72.2	72.1	-	-	82.1	61.7
Finetuned Transformer LM (ours)	82.1	81.4	89.9	88.3	88.1	56.0

Question Answering

Method	Story Cloze	RACE-m	RACE-h	RACE
val-LS-skip [55]	76.5	-	-	-
Hidden Coherence Model [7]	<u>77.6</u>	-	-	-
Dynamic Fusion Net [67] (9x)	-	55.6	49.4	51.2
BiAttention MRU [59] (9x)	-	<u>60.2</u>	<u>50.3</u>	<u>53.3</u>
Finetuned Transformer LM (ours)	86.5	62.9	57.4	59.0

GPT-2

- [Language Models are Unsupervised Multitask Learners](#) (2019)
- Compared to GPT-1
 - More parameters (1.5B parameters vs 117M)
 - More data (2.6GB and 4-5M words vs 40GB and 20B words)
 - Better at handling longer text, improving performance on tasks like summarization

GPT-3

- [Language Models are Few Shot Learners \(2020\)](#)
- Take-aways:
 - "Just **scaling up** language models can greatly improve task-agnostic, few-shot performance, sometimes even reaching competitiveness with prior state-of-the-art fine-tuning approaches. "
 - Introducing **Meta-learning**: the model develops a broad set of skills and pattern recognition ability at training time, then uses them at inference to rapidly adapt to the desire.
 - Attempts to meta-learning by "**in-context learning**", using the test input of a pretrained LM as a form of task specification.

The three settings we explore for in-context learning

Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.



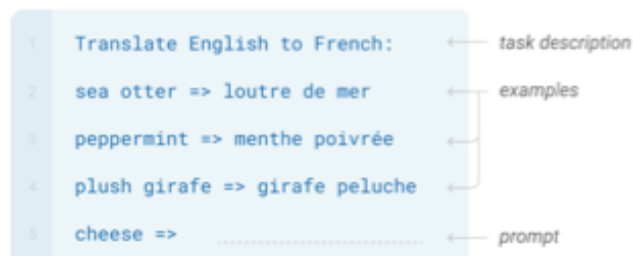
One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.



Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

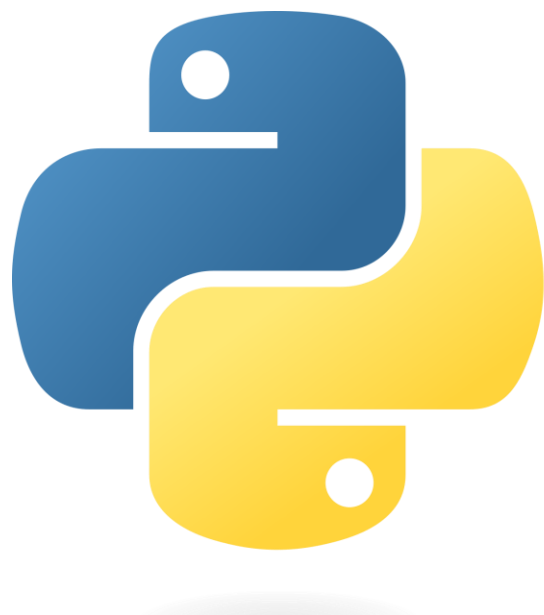


Traditional fine-tuning (not used for GPT-3)

Fine-tuning

The model is trained via repeated gradient updates using a large corpus of example tasks.





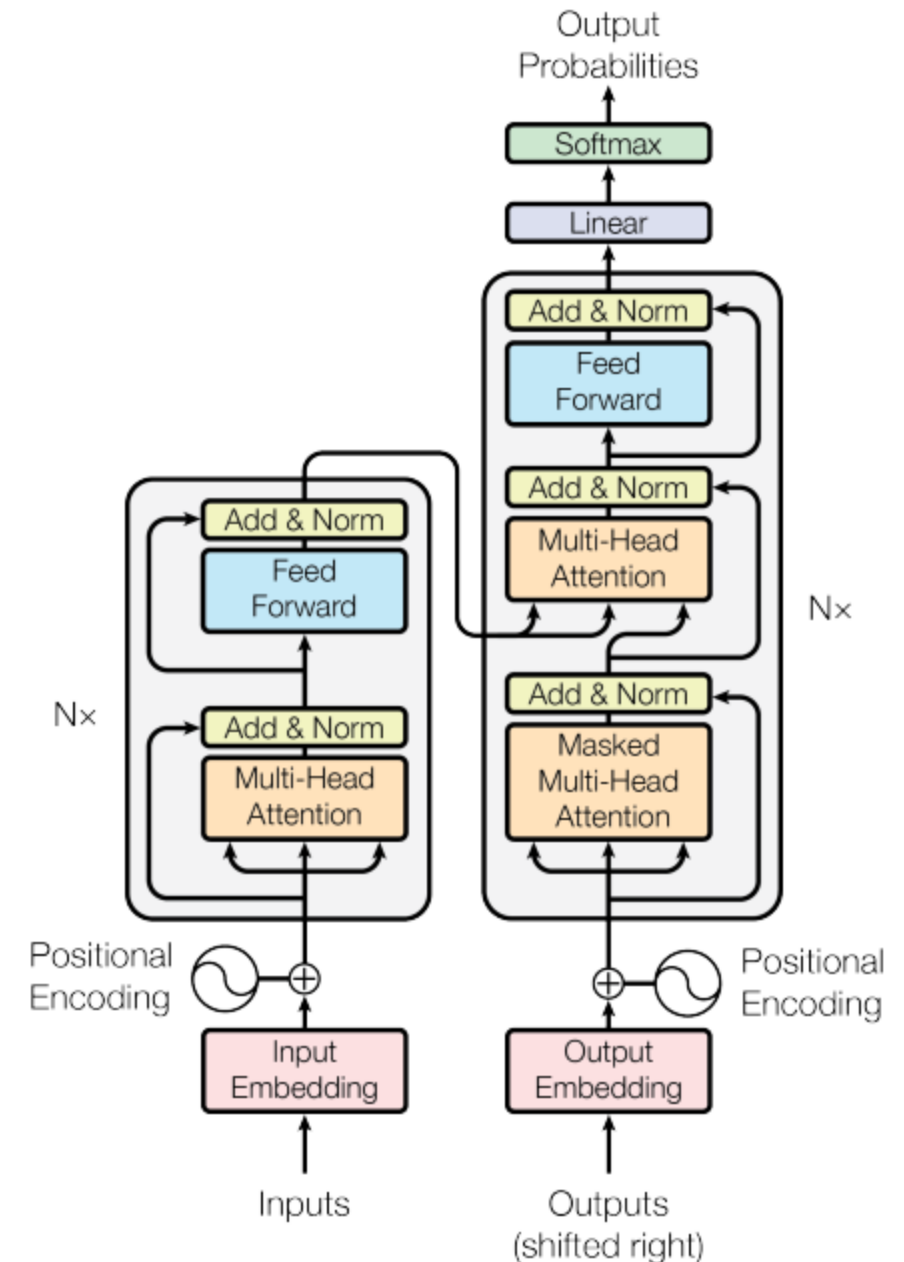
Encoder-Decoder Models (sequence-to-sequence)

Encoder-Decoder Tasks

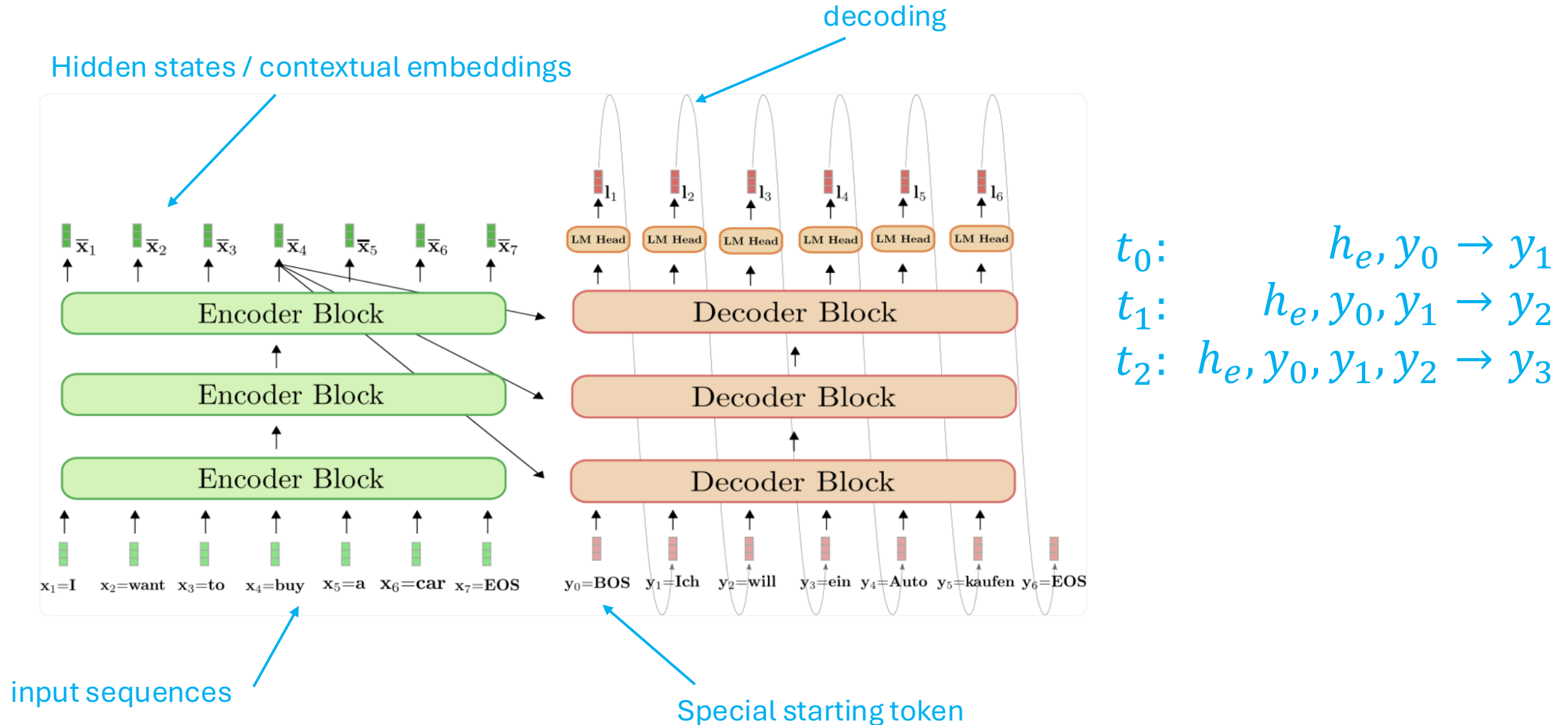
- These models are best suited for tasks revolving around generating new sequences of text depending on a given input
- **Abstractive Summarization**
 - Generating a summary of text while potentially using new phrases and sentences not found in the original text.
- **Translation**
 - Converts text from one language to another
- **Generative Question Answering**
 - Can use information stored in their learned parameters without context (though context may be required if information is proprietary / specialized)

Encoder-Decoder Models

- Use both parts of the original transformer architecture
- The input is processed by the encoder to produce a sequence of hidden states (contextual embeddings)
- The decoder generates an output based on:
 - All hidden states generated by the encoder
 - All **previous** outputs generated by the decoder

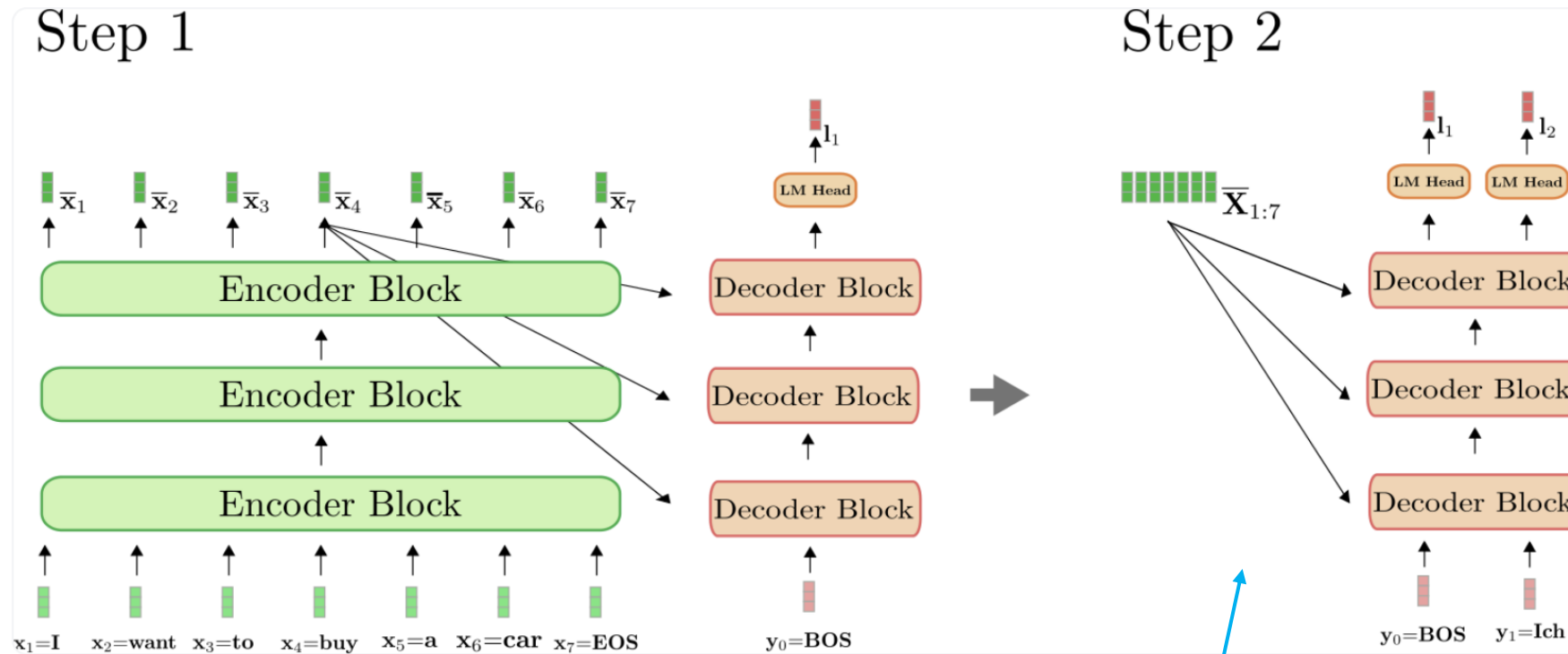


Encoder-Decoder Models



Encoder-Decoder Models

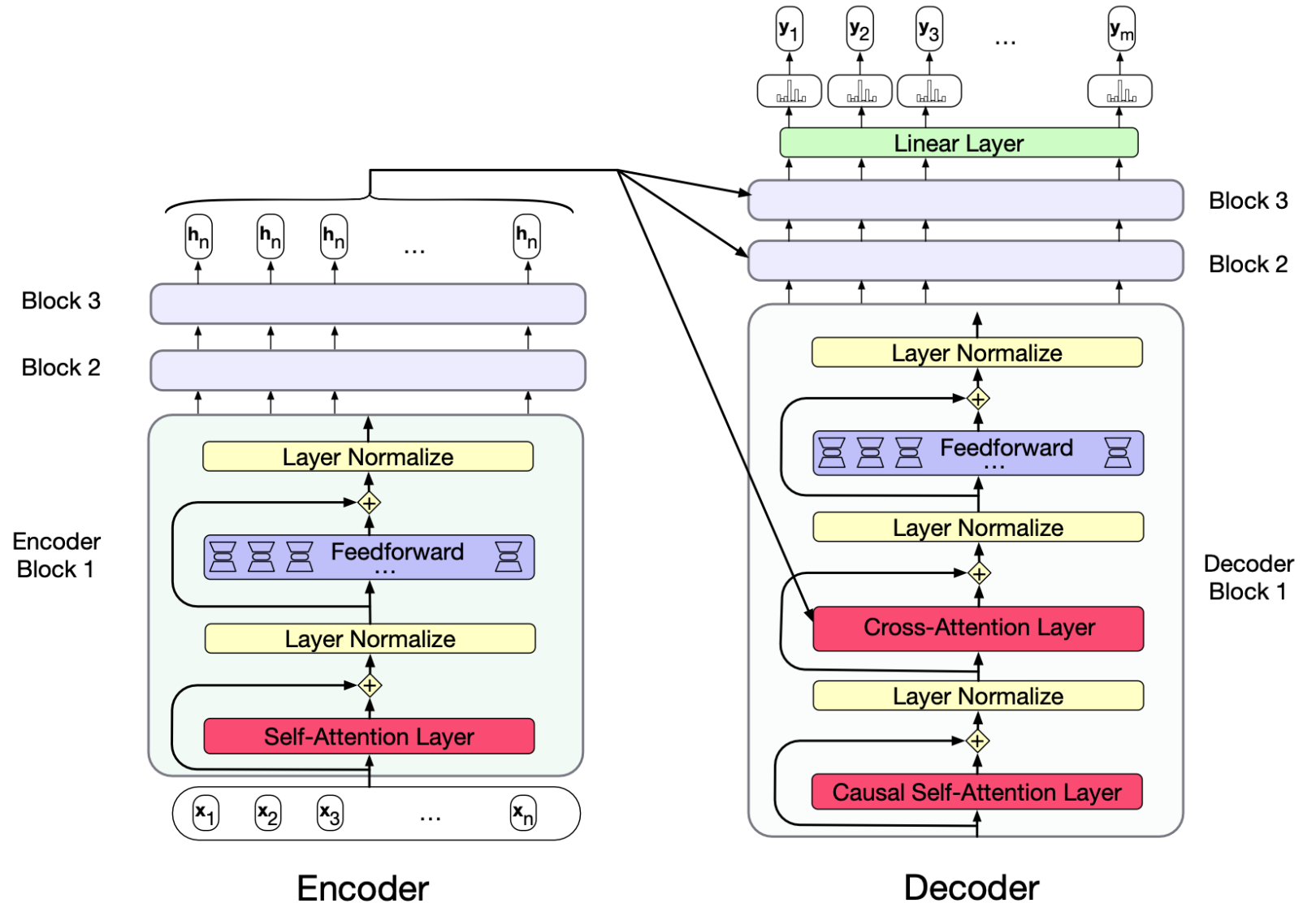
The encoder hidden states **are not** recalculated every time the decoder needs to generate a new token



Repeated forward passes only happen here

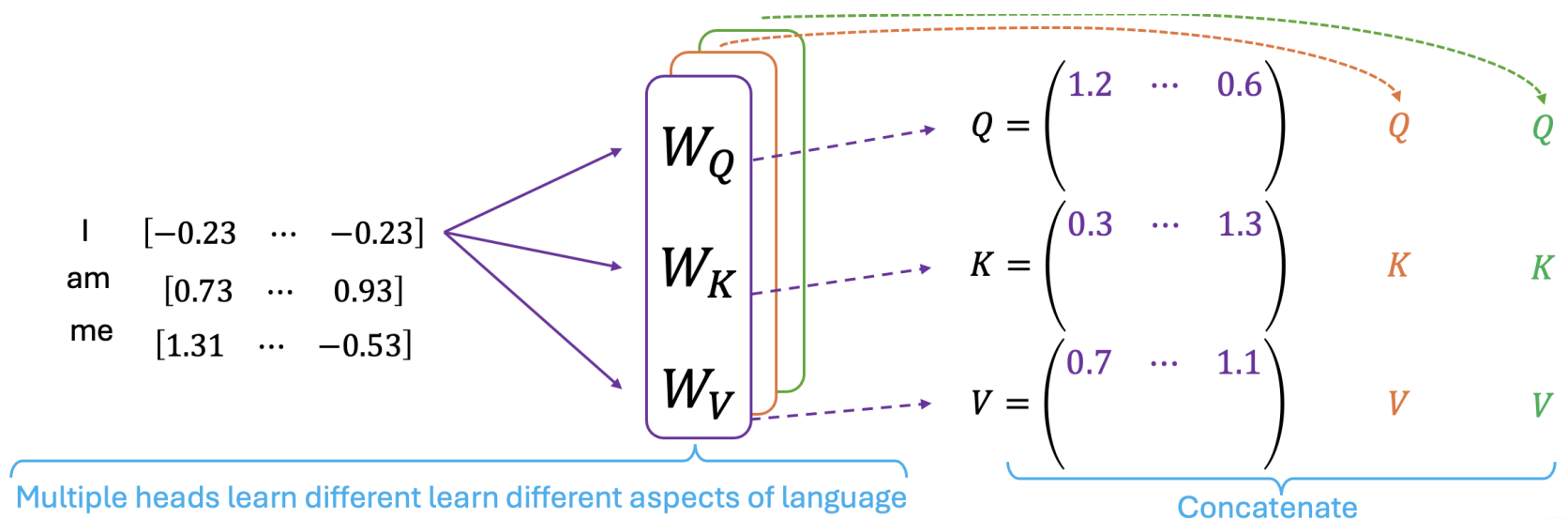
Cross Attention

- **Cross attention / encoder-decoder attention / source attention** allows the decoder to attend to the entire output of the encoder
- **Cross attention** is a (non-masked) attention layer that sits inside the decoder, and **uses the keys and values from the encoder**

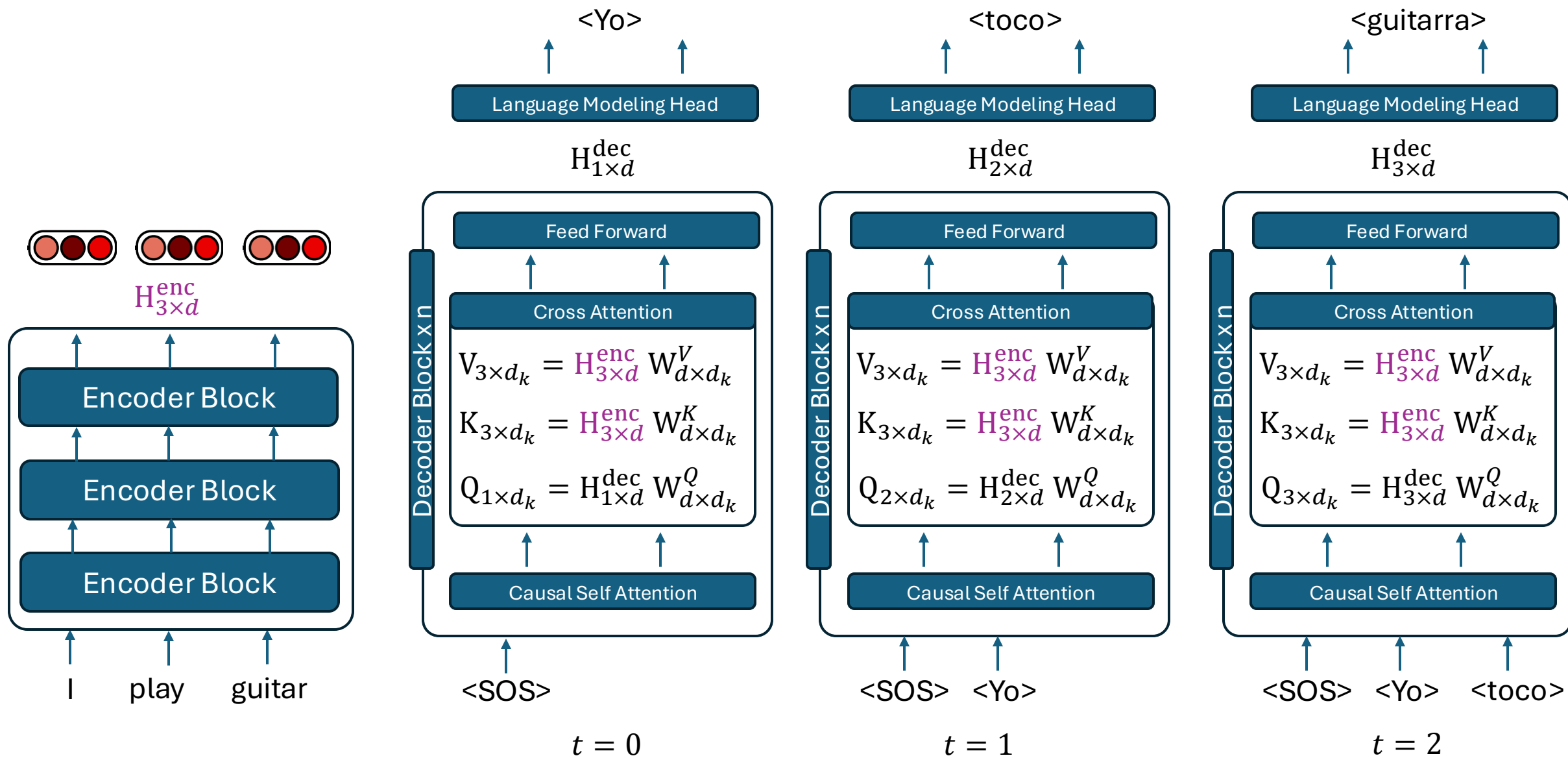


Cross Attention

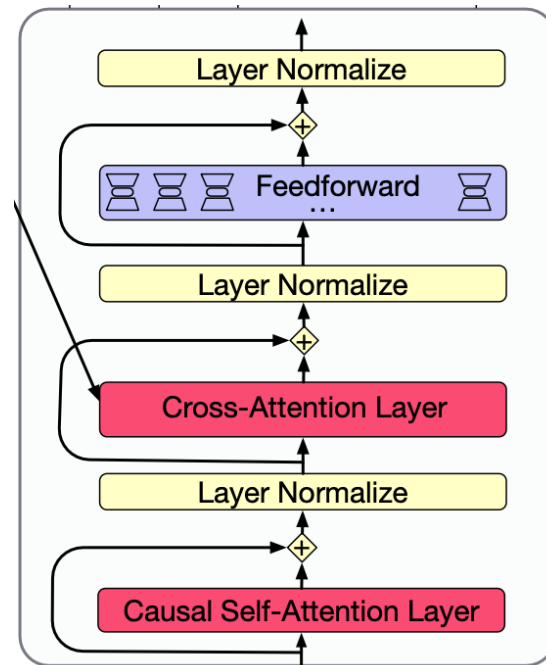
- **Reminder:** self-attention aims to dynamically weight inputs based on their contextual relevance
- Why use **Q** from decoder, while using **K** and **V** coming from encoder?
- **Q** comes from the **current state** of the decoder, contains text generated so far
- **K** helps to determine how each encoder element contributes to the attention weights
- **V** is the actual content from the encoder that gets weighted to produce the most relevant parts of the input



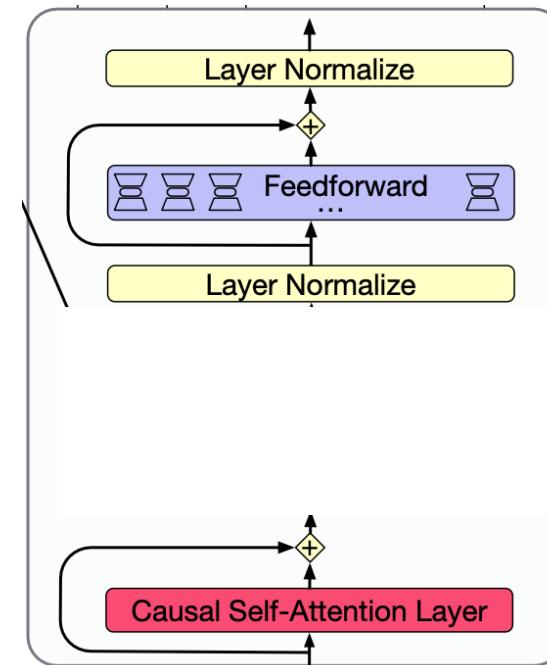
Cross Attention



Decoder of Decoder-Only vs Decoder of Encoder-Decoder



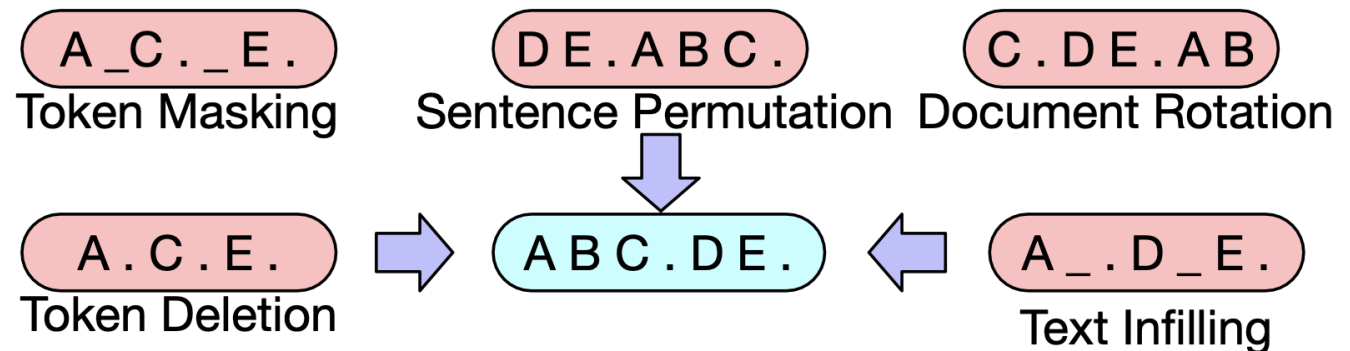
Encoder-Decoder



Decoder-Only

BART (**B**idirectional and **A**uto-**R**egressive **T**ransformers)

- Released in 2020 by Facebook/Meta [Publication](#)
- A denoising auto-encoder pre-trained by corrupting text and asking model to predict corrupted text. Corrupting text takes on various forms
 - **Masking**: randomly select and replace tokens with [MASK] token
 - **Deletion**: deletes random tokens, model must decide which positions are missing inputs
 - **Infilling**: spans of various lengths are replaced by single [MASK] token
 - **Permutation**: sentences are shuffled
 - **Rotation**: token is chosen at random, and document is rotated so it begins with this token



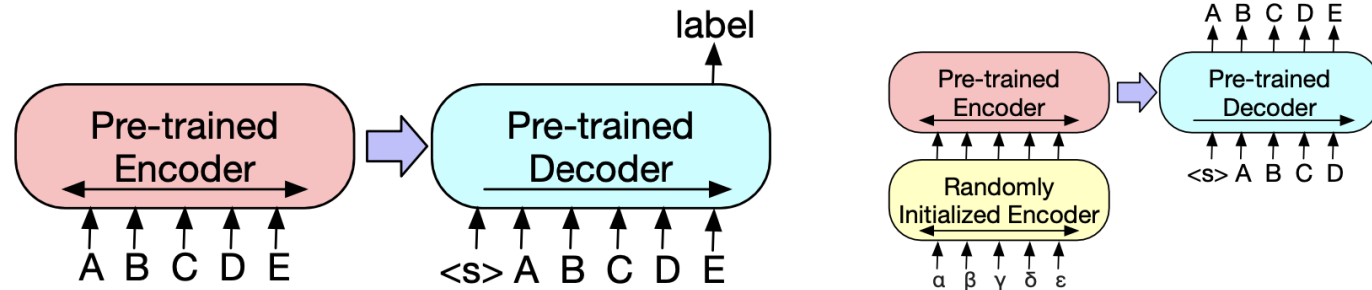
BART (**B**idirectional and **A**uto-**R**egressive **T**ransformers)

“BART shows large improvements on summarization metrics, of up to 3.5 points over the prior state-of-the-art”

Source Document (abbreviated)	BART Summary
The researchers examined three types of coral in reefs off the coast of Fiji ... The researchers found when fish were plentiful, they would eat algae and seaweed off the corals, which appeared to leave them more resistant to the bacterium <i>Vibrio coralliilyticus</i> , a bacterium associated with bleaching. The researchers suggested the algae, like warming temperatures, might render the corals' chemical defenses less effective, and the fish were protecting the coral by removing the algae.	Fisheries off the coast of Fiji are protecting coral reefs from the effects of global warming, according to a study in the journal Science.
Sacoolas, who has immunity as a diplomat's wife, was involved in a traffic collision ... Prime Minister Johnson was questioned about the case while speaking to the press at a hospital in Watford. He said, "I hope that Anne Sacoolas will come back ... if we can't resolve it then of course I will be raising it myself personally with the White House."	Boris Johnson has said he will raise the issue of US diplomat Anne Sacoolas' diplomatic immunity with the White House.

BART (**B**idirectional and **A**uto-**R**egressive **T**ransformers)

- Can be fine-tuned on other tasks as well.
- Sequence classification
- Token classification
- Abstractive question answering
- Abstractive summarization



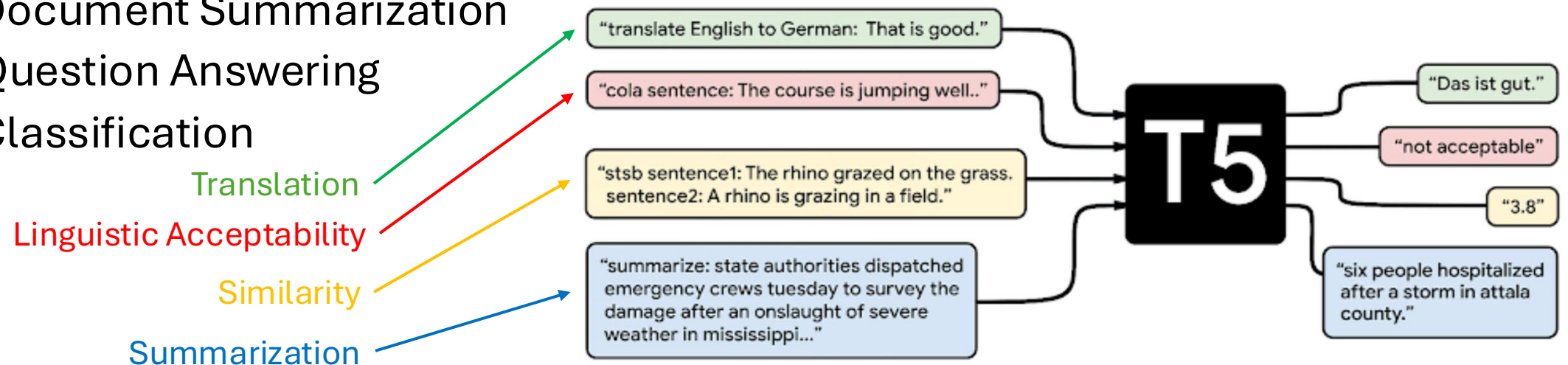
(a) To use BART for classification problems, the same input is fed into the encoder and decoder, and the representation from the final output is used.

(b) For machine translation, we learn a small additional encoder that replaces the word embeddings in BART. The new encoder can use a disjoint vocabulary.

Figure 3: Fine tuning BART for classification and translation.

T5 (Text-to-Text Transfer Transformer)

- Released in 2020 by [Google Research](https://arxiv.org/abs/1910.13461)
- Proposes reframing all NLP tasks into a unified text-to-text format where the input and output are always strings
 - Inputs signify what needs to be done
 - Outputs are strings
- Can use the same model, loss function, hyperparameters on any NLP task
 - Machine Translation
 - Document Summarization
 - Question Answering
 - Classification



T5 (Text-to-Text Transfer Transformer)

- Trained on purposefully developed dataset, **Colossal Clean Crawl Corpus** (C4)
 - Clean version of Common Crawl (deduplication, discarding incomplete sentences, discarding offensive or noisy content)
- Pre-trained using a **span-corruption** objective, wherein spans of text are masked with a single token
- Fine-tuned with certain task prefix, describing the nature of the task

Original text

Thank you ~~for inviting~~ me to your party ~~last~~ week.

Inputs

Thank you <X> me to your party <Y> week.

Targets

<X> for inviting <Y> last <Z>

D.1 CoLA

Original input:

Sentence: John made Bill master of himself.

Processed input: cola sentence: John made Bill master of himself.

Original target: 1

Processed target: acceptable

Abstractive Summarization

Summarization

- Summarization creates a shorter version of a text from a longer version, while trying to preserve most of the meaning of the original
- **Two types of summarization**
 - **Extractive:** Identify and extract most important sentences from text
 - **Abstractive:** Generate a summary, which may include words not present in original document

"Local farmers markets have become increasingly popular, offering fresh, organic produce directly from growers to consumers, which supports local economies."

Extractive: "Local farmers markets have become increasingly popular, offering fresh, organic produce."

Abstractive: "Farmers markets are gaining popularity by providing fresh organic produce and boosting local economies."

Use Cases

- Document summarization can be applicable to almost any downstream purpose
 - Meeting transcripts ([Zoom can do this.](#))
 - Emails, research papers, etc
- Can also be used as step in machine learning pipelines
 - Add summarization after topic modeling to better understand topics
 - Use summarization to reduce the length of documents before classification
 - Summarize text before creating textual embeddings to concentrate information
 - Summarization as a post-processing step for generated text

How are they fine-tuned?

- Most seq-2-seq models are pretrained on a corruption task
- If we take pre-trained model that hasn't been fine-tuned on task generation (like BART), and pass it through the generation pipeline, what will happen?
 - Will just return the text back as is
- **To fine-tune for summarization, we must provide it text-summary pairs: original text as input and our ideal summary as our target.**

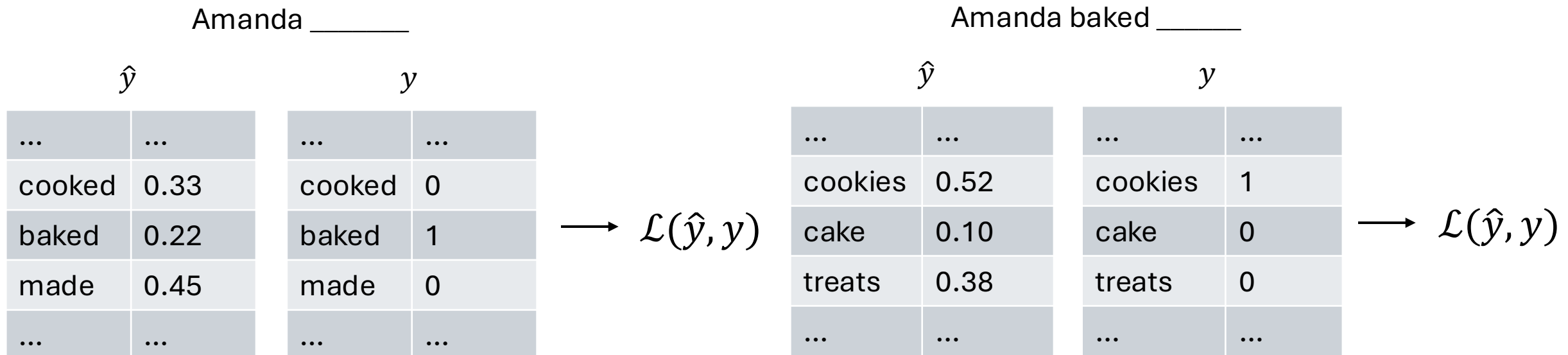
Original: The stock market experienced a significant drop yesterday, influenced by unexpected global economic news that led to widespread concern among investors.

Ideal Summary (Target / Reference): Global economic news caused a significant stock market drop yesterday.

How does the model learn?

- During training, the model generates an output that represents a summary
- At every token, the probability distribution for the next word is compared to the probability distribution of the correct word.
- Sum up the losses (or average) for the sequence

Ideal (Target / Refence) Summary: Amanda baked cookies and will bring Jerry some tomorrow.



Evaluation Metrics

- **Why are evaluation metrics for summarization difficult?**
- **Subjectivity** – what constitutes a good summary can be highly dependent on the reader
- **Multiple answers** – there can be any number of ways to formulate a summarization, all of which can be correct
- **Error of Omission** – Was the most relevant information captured or left out?
- **Hallucination** – The model can generate information not present in the original context

Evaluation Metrics – ROUGE-N

- **Recall-based** metric that focuses on the fraction of n-grams in the reference text that appear in the generated text

$$r_n = \frac{\text{Count}(\text{ref ngram} \in \text{gen})}{\text{Count}(\text{ref ngram})}$$

- The above can be combined with (unclipped) BLEU (via F1)
- There are several variations:
 - **ROUGE-L**: Uses the longest common subsequence instead of n-grams.
 - **ROUGE-S**: Uses skip-grams instead of n-grams

Evaluation Metrics - ROUGE

Original: “Solar and wind power reduce fossil fuel reliance”

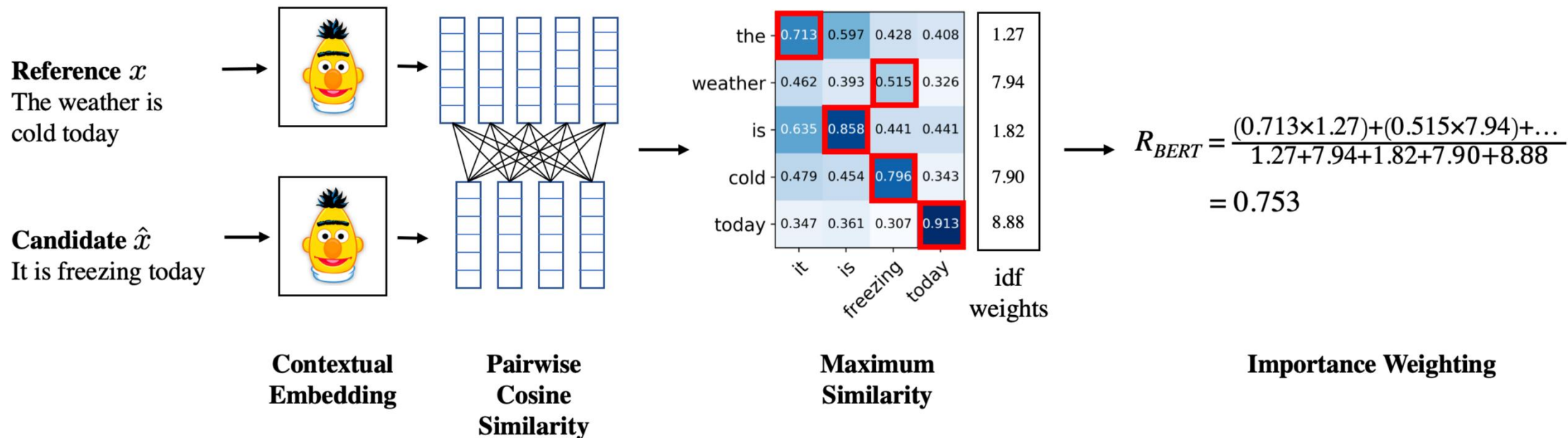
Reference: “Renewable energy cuts fossil fuels”

Candidate 1: “Renewable energy”

Candidate 2: “Renewables aid sustainability and reduce fossil fuels”

	Candidate 1	Candidate 2
# Unigrams In Ref.	5	5
# of Ref. grams in Candidate	2	2
ROUGE-1	$2/5 = 0.40$	$2/5 = 0.40$

Evaluation Metrics - BERTscore



$$R_{BERT} = \frac{1}{|x|} \sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} \mathbf{x}_i^\top \hat{\mathbf{x}}_j, \quad P_{BERT} = \frac{1}{|\hat{x}|} \sum_{\hat{x}_j \in \hat{x}} \max_{x_i \in x} \mathbf{x}_i^\top \hat{\mathbf{x}}_j, \quad F_{BERT} = 2 \frac{P_{BERT} \cdot R_{BERT}}{P_{BERT} + R_{BERT}}$$

Evaluation Metrics - Challenges

- In practice, tokenization method will affect counts (important if comparing across different models)
- Some methods don't consider semantic similarity of words
- Measuring the quality of generated text is still an area of research
- **Human judgement remains the best metric**

Summarization Challenges: Document Length

- Transformer models have finite context windows; how to overcome?
- **Segmentation/Windowing**
 - Split documents into chunks that can fit inside context window and summarize each individually
 - Can also use sliding window approach with overlap
- **Extractive Pre-Summarization**
 - Select most important sentences from document via extractive summarization, then use abstractive summarization to generate coherent summaries
- **Use Domain Knowledge**
 - If we know most important information is found somewhere in the document, we can use that as opposed to entire document

Summarization Challenges: Contextual Integrity

- Does the generated summary contain accurately reflect original content?
- **Human-in-the-loop**
 - A more advanced strategy that involves people evaluating the model outputs with actual summaries and providing scores from which models can learn
- **Entity Recognition**
 - Incorporating cross checking of named entities. Are there entities mentioned in the generated summary that are not present in the original (hallucination)? Are there entities mentioned in the original that are not in the summary (recall problem)?
- **Semantic Similarity Checks**
 - Incorporating other models to flag summaries that seem semantically much different than originals

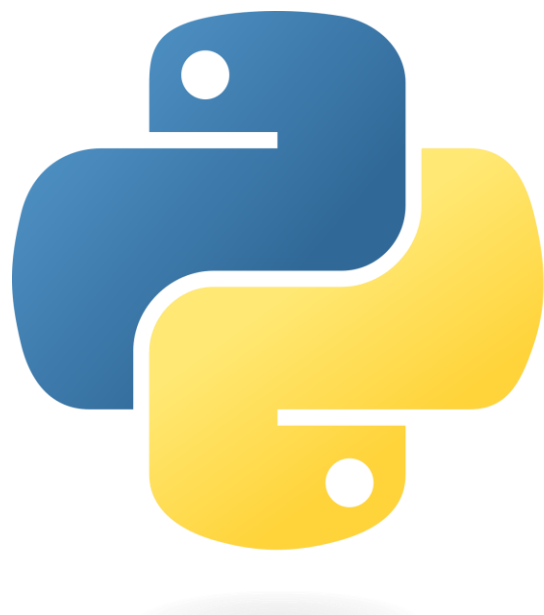
Benchmark Datasets

- **CNN/Daily Mail**

- Articles from CNN and Daily Mail
- Contains human generated, multi-sentence summaries
- Articles tend to be longer, and summaries are abstractive (but not overly so)

- **XSum** (Extreme Summarization)

- Articles from BBC with single sentence summaries written by journalists
- Intended for abstractive summarization



Abstractive Question Answering

Question Answering

- Extractive vs Abstractive
 - **Extractive:** given a question and some context, the answer is a span of text from the context the model must extract
 - **Abstractive:** given a question and some context, the answer is generated from the context, with a high possibility of wording the answer in a way that is not present in the context
- Closed domain vs open domain:
 - **Closed domain:** deals with questions under a **specific** domain OR when only a limited type of questions are accepted
 - **Open domain:** deals with questions about nearly anything and can only rely on general ontologies and world knowledge

Abstractive Question Answering

- **Extractive** question-answering is a **span-classification** problem wherein the model assigns start and end logits to every token
- **Abstractive** question-answering is a **seq-to-seq** task, wherein the model receives question + context as the input, and translates that to an appropriate output

Context: "The Great Wall of China is an ancient series of walls and fortifications located in northern China, built around 500 years ago during the Ming Dynasty in response to invasions from Mongol forces. The Great Wall is actually a succession of multiple walls spanning approximately 13,000 miles, making it one of the most impressive architectural feats in history."

Question: "Why was the Great Wall of China originally built?"

Answer: "The Great Wall of China was constructed to protect against invasions by Mongol forces during the Ming Dynasty."

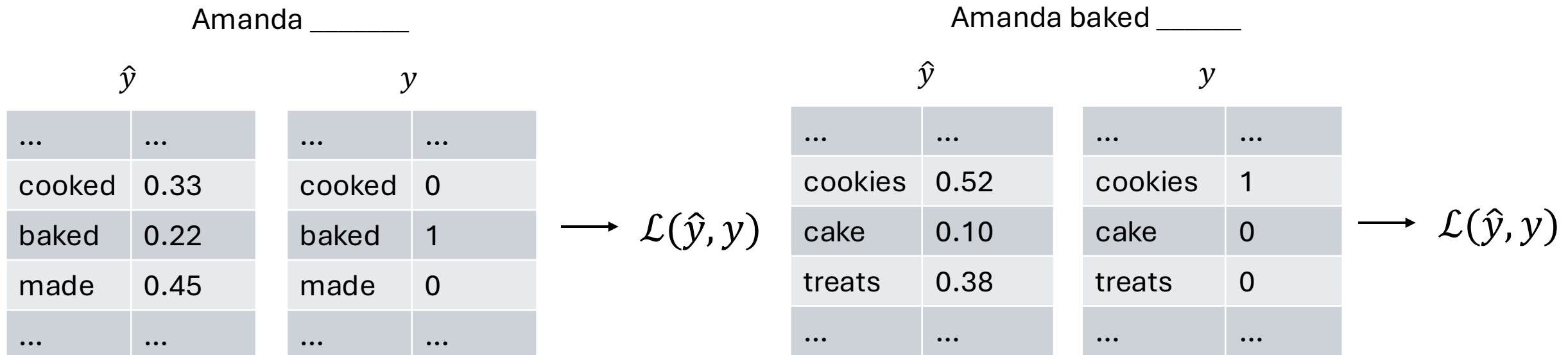
Use Cases

- **Extractive QA** is useful when precise, verbatim answers are required, and source material contains explicit answers
 - Legal documents, medical record, technical manuals
- **Abstractive QA** is useful when more human-like / natural / conversational responses that can paraphrase, summarize, or blend information is preferred

How does the model learn?

- During training, the model generates an output that represents an answer
- At every token, the probability distribution for the next word is compared to the probability distribution of the correct word.
- Sum up the losses (or average) for the sequence

Ideal (Target) Summary: Amanda baked cookies and will bring Jerry some tomorrow.



Evaluation Metrics

- **Why are evaluation metrics for question answering difficult?**
- **Error of Omission** – Was the most relevant information captured or left out?
- **Hallucination** – The model can generate information not present in the original context

Evaluation Metrics - BLEU

- **Precision-based** metric that looks at the fraction of generated n-grams that are correct (that also appear in the reference)
- Count the # of n-grams in generated text that occur in the reference and divide it by the # of words n-grams in the generation
 - Important detail: an n-gram is only counted **at most as many times as it appears in the reference text (clipping)**

$$p_n = \frac{\text{Count}_{clip}(\text{gen ngram} \in \text{ref})}{\text{Count}(\text{gen ngram})}$$

- Precision favors shorter texts here (shorter generated summaries have less chance to make mistakes)

$$BP = \min(1, e^{1 - l_{ref}/l_{gen}})$$

Evaluation Metrics - BLEU

- Putting everything together, we get:

$$BP \times \left(\prod_{n=1}^N p_n \right)^{1/N} = \min(1, e^{1 - l_{ref}/l_{gen}}) \left(\prod_{n=1}^N p_n \right)^{1/N}$$

Original: “Solar and wind power reduce fossil fuel reliance”

Reference: “Renewable energy cuts fossil fuels”

Candidate 1: “Renewable energy”

Candidate 2: “Renewables aid sustainability and reduce fossil fuels”

	Candidate 1	Candidate 2
Unigrams Total (Ideal)	5	5
Matches	2	2
Precision	$2/2 = 100\%$	$2/6 \approx 33\%$
Brevity Penalty (BP)	$e^{1 - 5/2} = e^{-1.5} = 0.2231$	1 ($l_{gen} \geq l_{ref}$)
BLEU Score	22.31% (0.2231 x 100%)	33%

