

## COMP7015 Artificial Intelligence (S1, 2024-25)

# Lecture 11: Probabilistic Methods, Generative AI, and Course Review

Instructor: Dr. Kejing Yin ([cskjyin@hkbu.edu.hk](mailto:cskjyin@hkbu.edu.hk))

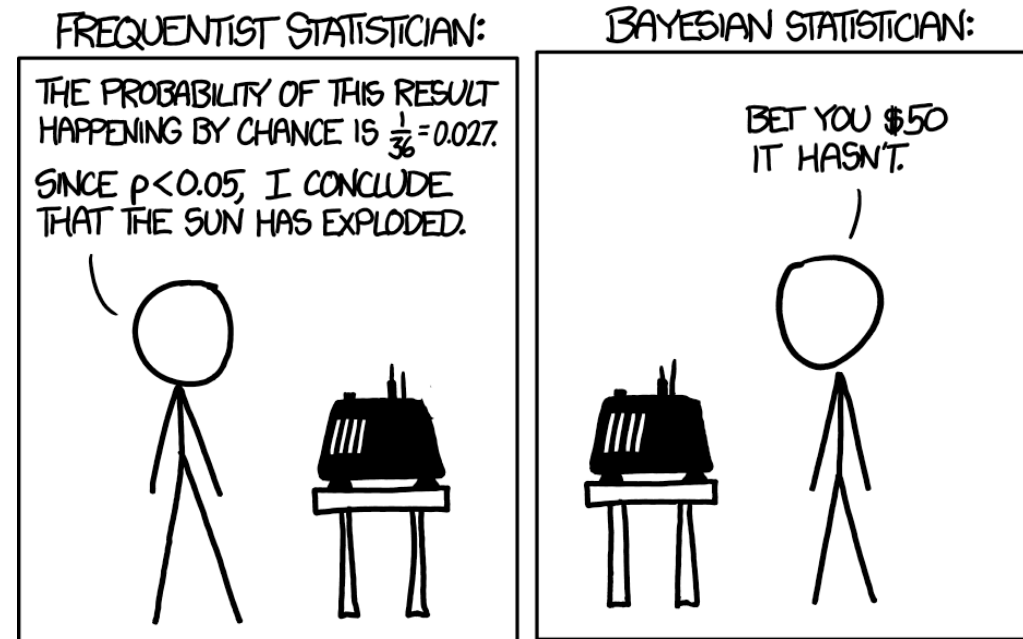
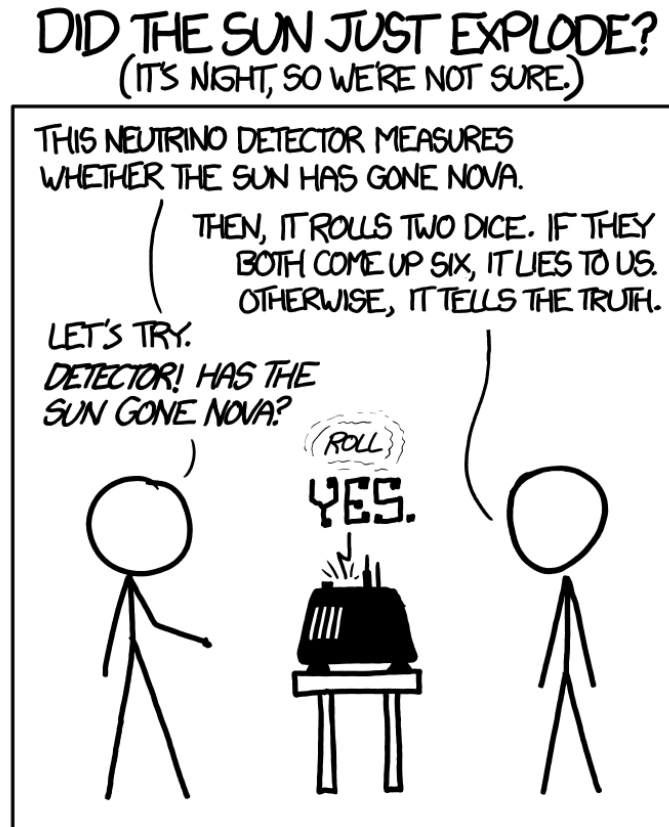
Department of Computer Science  
Hong Kong Baptist University

November 29, 2024

# Probabilistic Methods: Bayesian Decision Theory and Naïve Bayes Classifier

Credits: this section is partially adapted from Lecture 6 of CS440/CEC448 (UIUC) under CC-BY-4.0 license

# Frequentist Statistician vs. Bayesian Statistician



© <https://www.xkcd.com/1132/>

# Recap: Bayes' Rule

- The product rule gives us two ways to factor a joint probability:

$$P(A, B) = P(B|A)P(A) = P(A|B)P(B)$$

- Therefore,

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

- Why is this useful?
  - “A” is something we care about, but  $P(A|B)$  is really really hard to measure (example: the sun exploded)
  - “B” is something less interesting, but  $P(B|A)$  is easy to measure (example: the amount of light falling on a solar cell)
  - Bayes' rule tells us how to compute the probability we want ( $P(A|B)$ ) from probabilities that are much, much easier to measure ( $P(B|A)$ ).



Rev. Thomas Bayes  
(1702-1761)

By Unknown -  
[2][3], Public  
Domain,  
[https://commons.  
wikimedia.org/w/i  
ndex.php?curid=1  
4532025](https://commons.wikimedia.org/w/index.php?curid=14532025)

# Bayes Rule example

Eliot & Karson are getting married tomorrow, at an outdoor ceremony in the desert. Unfortunately, the weatherman has predicted rain for tomorrow.

- In recent years, it has rained (event  $R$ ) only 5 days each year ( $5/365 = 0.014$ ).

$$P(R) = 0.014$$

- When it actually rains, the weatherman forecasts rain (event  $F$ ) 90% of the time.

$$P(F|R) = 0.9$$

- When it doesn't rain, he forecasts rain (event  $F$ ) only 10% of the time.

$$P(F|\neg R) = 0.1$$

- What is the probability that it will rain on Eliot's wedding?

$$\begin{aligned} P(R|F) &= \frac{P(F|R)P(R)}{P(F)} = \frac{P(F|R)P(R)}{P(F, R) + P(F, \neg R)} = \frac{P(F|R)P(R)}{P(F|R)P(R) + P(F|\neg R)P(\neg R)} \\ &= \frac{(0.9)(0.014)}{(0.9)(0.014) + (0.1)(0.956)} = 0.116 \end{aligned}$$

# The More Useful Version of Bayes' Rule



Rev. Thomas Bayes  
(1702-1761)

By Unknown -  
[2][3], Public  
Domain,  
[https://commons.  
wikimedia.org/w/i  
ndex.php?curid=1  
4532025](https://commons.wikimedia.org/w/index.php?curid=14532025)

*This version is what you  
memorize.*

$$\rightarrow P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

- $P(B|A)$  is easy to measure (the probability that light hits our solar cell, if the sun still exists and it's daytime). Let's assume we also know  $P(A)$  (the probability the sun still exists).
- But suppose we don't really know  $P(B)$  (what is the probability light hits our solar cell, if we don't really know whether the sun still exists or not?)

*This version is what you  
often actually use.*

$$\rightarrow P(A|B) = \frac{P(B|A)P(A)}{P(B|A)P(A) + P(B|\neg A)P(\neg A)}$$

# The Misdiagnosis Problem

- 1% of women at age 40 who participate in routine screening have breast cancer.
- 80% of women with breast cancer will get positive mammographies.
- 9.6% of women without breast cancer will also get positive mammographies.
- Question: A woman in this age group had a positive mammography in a routine screening. What is the probability that she actually has breast cancer?

$$\begin{aligned} P(\text{cancer} \mid \text{positive}) &= \frac{P(\text{positive} \mid \text{cancer})P(\text{cancer})}{P(\text{positive})} \\ &= \frac{P(\text{positive} \mid \text{cancer})P(\text{cancer})}{P(\text{positive} \mid \text{cancer})P(\text{cancer}) + P(\text{positive} \mid \neg \text{cancer})P(\neg \text{cancer})} \\ &= \frac{0.8 \times 0.01}{0.8 \times 0.01 + 0.096 \times 0.99} = \frac{0.008}{0.008 + 0.095} = 0.0776 \end{aligned}$$

CHECK YOUR SYMPTOMS

FIND A DOCTOR

FIND LOWEST DRUG PRICES

SIGN IN

SUBSCRIBE

WebMD

HEALTH A-Z

DRUGS & SUPPLEMENTS

LIVING HEALTHY

FAMILY & PREGNANCY

NEWS & EXPERTS

SEARCH

ADVERTISEMENT

HEALTH INSURANCE AND MEDICARE HOME

News

Reference

Quizzes

Videos

Message Boards

Find a Doctor


Health Insurance and Medicare | Reference

Second Opinions


" # \$ ' & %

If your doctor tells you that you have a health problem or suggests a treatment for an illness or injury, you might want a second opinion. This is especially true when you're considering surgery or major procedures.


TODAY ON WEBMD



**Clinical Trials**  
What qualifies you for one?



**Working During Cancer Treatment**  
Know your benefits.



**Going to the Dentist?**  
How to save money.

COMP7015 (HKBU)

L11: Bayesian, AIGC, and Review

November 29, 2024



# Application in AI: Bayesian Decision Theory

- The agent is given some evidence,  $E$ .
- The agent has to make a decision about the value of an unobserved variable  $Y$ .  $Y$  is called the “class variable” or the “category” (“label”).
  - Partially observable, stochastic, episodic environment
  - Example:  $Y \in \{\text{spam}, \text{not spam}\}$ ,  $E$  = email message.
  - Example:  $Y \in \{\text{zebra}, \text{giraffe}, \text{hippo}\}$ ,  $E$  = image features

✗

Dear Sir.

First, I must solicit your confidence in this transaction, this is by virtue of its nature as being utterly confidential and top secret. ...

✗

TO BE REMOVED FROM FUTURE MAILINGS, SIMPLY REPLY TO THIS MESSAGE AND PUT "REMOVE" IN THE SUBJECT.

99 MILLION EMAIL ADDRESSES FOR ONLY \$99

✓

Ok, I know this is blatantly OT but I'm beginning to go insane. Had an old Dell Dimension XPS sitting in the corner and decided to put it to use, I know it was working pre being stuck in the corner, but when I plugged it in, hit the power nothing happened.



# Classification Using Probabilities

- Suppose you know that you have a toothache.
- Should you conclude that you have a cavity?
- Goal: make a decision that minimizes your probability of error.
- Equivalent: make a decision that maximizes the probability of being correct. This is called a MAP (maximum a posteriori) decision.
- You decide that you have a cavity if and only if

$$P(\text{Cavity} | \text{Toothache}) > P(\neg \text{Cavity} | \text{Toothache})$$

*Label Y*                      *Evidence E*

# Bayesian Decisions

- What if we don't know  $P(\text{Cavity}|\text{Toothache})$ ? Instead, we only know  $P(\text{Toothache}|\text{Cavity})$ ,  $P(\text{Cavity})$ , and  $P(\text{Toothache})$ ?
- Then we choose to believe we have a Cavity if and only if

$$P(\text{Cavity}|\text{Toothache}) > P(\neg\text{Cavity}|\text{Toothache})$$

Which can be re-written as

$$\frac{P(\text{Toothache}|\text{Cavity})P(\text{Cavity})}{P(\text{Toothache})} > \frac{P(\text{Toothache}|\neg\text{Cavity})P(\neg\text{Cavity})}{P(\text{Toothache})}$$

# MAP (Maximum A Posteriori) Decision

The action, “a”, should be the value of C that has the highest posterior probability given the observation  $E = e$ :

$$\begin{aligned} a = \operatorname{argmax} P(Y = a|E = e) &= \operatorname{argmax} \frac{P(E = e|Y = a)P(Y = a)}{P(E = e)} \\ &= \operatorname{argmax} P(E = e|Y = a)P(Y = a) \end{aligned}$$

$$\underbrace{P(Y = a|E = e)}_{\text{posterior}} \propto \underbrace{P(E = e|Y = a)}_{\text{likelihood}} \underbrace{P(Y = a)}_{\text{prior}}$$

# The Bayesian Terms

- $P(Y)$  is called the “**prior**” (*a priori*, in Latin) because it represents your belief about the query variable before you see any observation.
- $P(Y|E)$  is called the “**posterior**” (*a posteriori*, in Latin), because it represents your belief about the query variable after you see the observation.
- $P(E|Y)$  is called the “**likelihood**” because it tells you how much the observation,  $E=e$ , is like the observations you expect if  $Y=y$ .
- $P(E)$  is called the “**evidence distribution**” because  $E$  is the evidence variable, and  $P(E)$  is its marginal distribution.

$$P(Y = y|E = e) = \frac{P(E = e|Y = y)P(Y = y)}{P(E = e)}$$

# Naïve Bayes Classifier

- Suppose we have many different types of observations (symptoms, features)  $E_1, \dots, E_n$  that we want to use to obtain evidence about an underlying hypothesis  $Y$ .
- MAP decision:

$$P(Y|E_1, \dots, E_n) \propto P(Y)P(E_1, \dots, E_n|Y)$$

- THE BIG PROBLEM: If each feature  $E_i$  can take on  $K$  values, how many entries are in the probability table  $P(E_1, \dots, E_n|Y)$ ? How can we calculate this from finite training samples?

# Naïve Bayes Classifier

- Suppose we have many different types of observations (symptoms, features)  $E_1, \dots, E_n$  that we want to use to obtain evidence about an underlying hypothesis  $Y$ .
- The naïve Bayes classifier makes the “**attribute conditional independence assumption**”: given any known class, assume all attributes are independent of each other. (*assume that each attribute is influenced by the class labels independently.*)

$$\begin{aligned} a &= \operatorname{argmax} p(Y = a | E_1 = e_1, \dots, E_n = e_n) \\ &= \operatorname{argmax} p(Y = a) p(E_1 = e_1, \dots, E_n = e_n | Y = a) \\ &\approx \operatorname{argmax} p(Y = a) p(E_1 = e_1 | Y = a) \dots p(E_n = e_n | Y = a) \end{aligned}$$

# Naïve Bayes Classifier Example

- You are a robot in an animal shelter and must learn to discriminate Dogs from Cats. You are given the following examples. Classify a new example (Sound=Meow, Fur=Fine, and Color=Black) using naïve Bayes classifier.

Example	Sound	Fur	Color	Class
Example #1	Meow	Coarse	Brown	Dog
Example #2	Bark	Fine	Brown	Dog
Example #3	Bark	Coarse	Black	Dog
Example #4	Bark	Coarse	Black	Dog
Example #5	Meow	Fine	Brown	Cat
Example #6	Meow	Coarse	Black	Cat
Example #7	Bark	Fine	Black	Cat
Example #8	Meow	Fine	Brown	Cat



# Naïve Bayes Classifier Example

- You are a robot in an animal shelter and must learn to discriminate Dogs from Cats. You are given the following examples. Classify a new example (Sound=Meow, Fur=Fine, and Color=Black) using naïve Bayes classifier.

Example	Sound	Fur	Color	Class
Example #1	Meow	Coarse	Brown	Dog
Example #2	Bark	Fine	Brown	Dog
Example #3	Bark	Coarse	Black	Dog
Example #4	Bark	Coarse	Black	Dog
Example #5	Meow	Fine	Brown	Cat
Example #6	Meow	Coarse	Black	Cat
Example #7	Bark	Fine	Black	Cat
Example #8	Meow	Fine	Brown	Cat

Priors:  $P(\text{class}=\text{Cat}) = 4/8 = 0.5$ ;  $P(\text{class}=\text{Dog}) = 4/8 = 0.5$

Conditional probability of the features:

$$P(\text{Sound}=\text{Meow} \mid \text{class}=\text{Cat}) = 3/4 = 0.75$$

$$P(\text{Sound}=\text{Meow} \mid \text{class}=\text{Dog}) = 1/4 = 0.25$$

$$P(\text{Fur}=\text{Fine} \mid \text{class}=\text{Cat}) = 3/4 = 0.75$$

$$P(\text{Fur}=\text{Fine} \mid \text{class}=\text{Dog}) = 1/4 = 0.25$$

$$P(\text{Color}=\text{Black} \mid \text{class}=\text{Cat}) = 2/4 = 0.5$$

$$P(\text{Color}=\text{Black} \mid \text{class}=\text{Dog}) = 2/4 = 0.5$$

$$P(c=\text{Dog})P(\text{Sound}=\text{Meow} \mid c=\text{Dog}) P(\text{Fur}=\text{Fine} \mid c=\text{Dog}) P(\text{Color}=\text{Black} \mid c=\text{Dog}) = 0.5 * 0.25 * 0.25 * 0.5 = 0.016$$

$$P(c=\text{Cat})P(\text{Sound}=\text{Meow} \mid c=\text{Cat}) P(\text{Fur}=\text{Fine} \mid c=\text{Cat}) P(\text{Color}=\text{Black} \mid c=\text{Cat}) = 0.5 * 0.75 * 0.75 * 0.5 = 0.14$$

Since  $0.14 > 0.016$ , the naïve Bayes classifier will classify this new example as a Cat.

# Naïve Bayes Classifier Exercise

- Given the following dataset, what would be the prediction generated by a naïve Bayes classifier for the new sample (blue, round, small)?

	color	shape	size	class
1	red	square	big	+
2	blue	square	big	+
3	red	round	small	-
4	green	square	small	-
5	red	round	big	+
6	green	round	big	-

# Naïve Bayes Classifier: Pros and Cons

- Pros:

- Naïve Bayes classifier is fast and simple. We just calculate the probabilities.
- Easy to implement.
- Incremental learning: just update the probabilities when new data arrive.
- Suitable for large datasets: since it is very efficient.

- Cons:

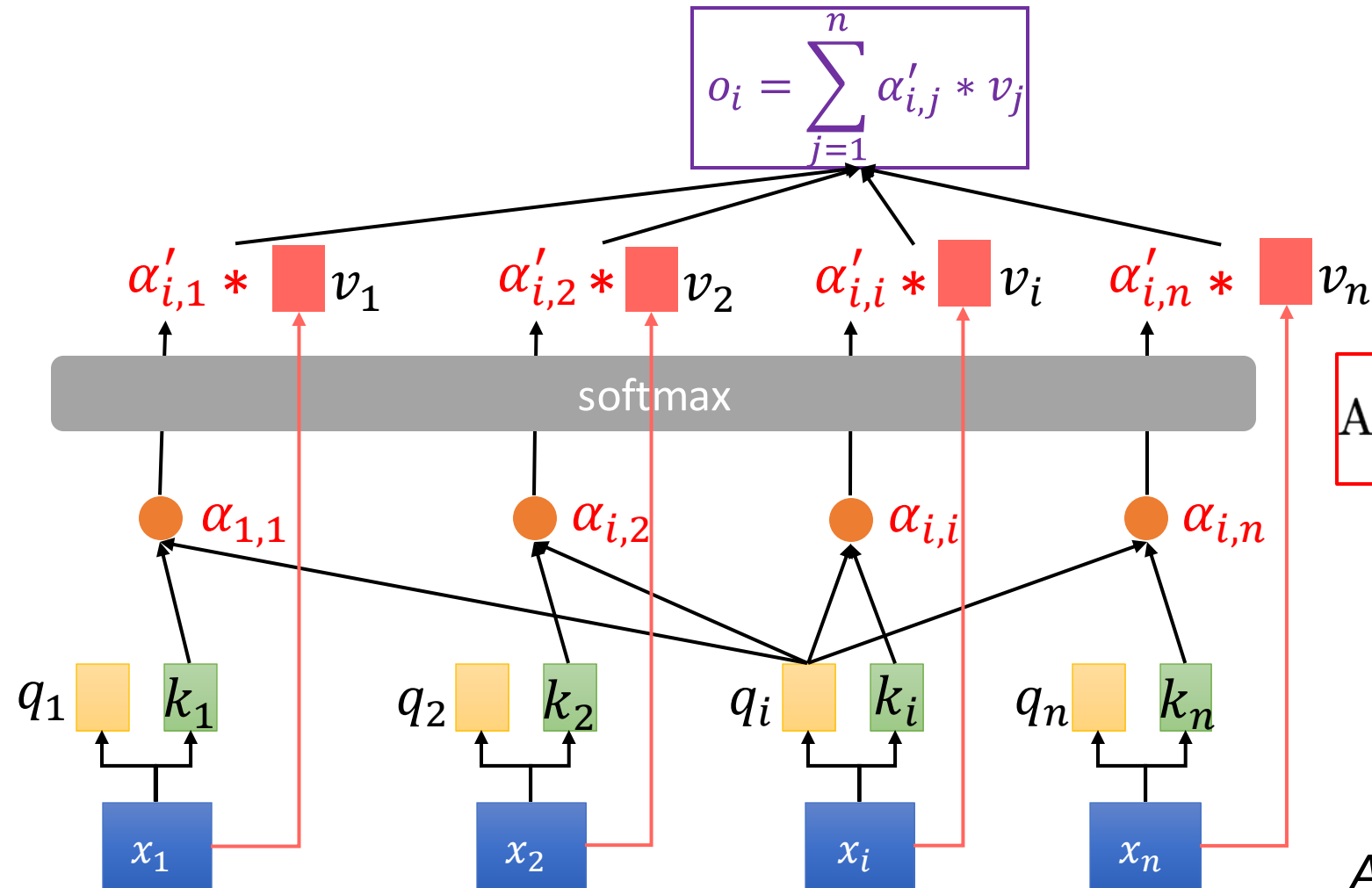
- The attribute conditional independence assumption is a strong assumption and may not be satisfied by real data.
- Cannot do regression (can only make predictions for categorical labels).
- Zero frequency problem: cannot handle new values of a feature in the test set as naïve Bayes classifier will assign zero probability (Color=white in the previous example).

# Generative AI Models

Credits: this section is partially adapted from Lecture 6 of CS440/CEC448 (UIUC) under CC-BY-4.0 license

# Recap: Attention is All You Need (forget about RNNs)

- Self-attention computes attention between words in a sentence



Scaled dot-product attention

Rewrite in matrix form:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}}\right)\mathbf{V}$$

$d$ : hidden dimension

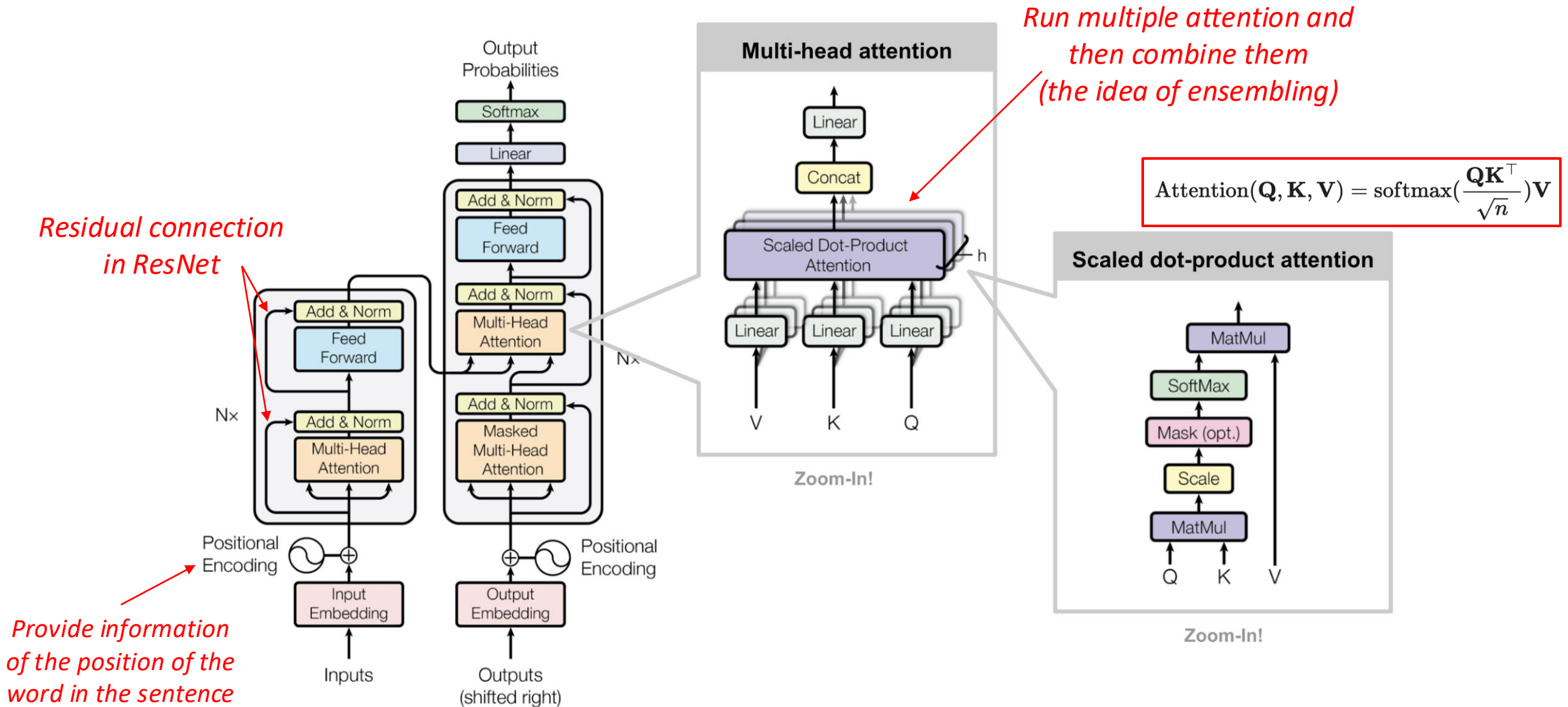
$$\mathbf{Q} = \mathbf{X}\mathbf{W}^Q$$

$$\mathbf{K} = \mathbf{X}\mathbf{W}^K$$

$$\mathbf{V} = \mathbf{X}\mathbf{W}^V$$

A core element of Transformer model

# Recap: Transformer Architecture



# BERT

**Bert:** Pre-training of deep bidirectional transformers for language understanding

J Devlin, [MW Chang](#), [K Lee](#), [K Toutanova](#) - arXiv preprint arXiv ..., 2018 - [arxiv.org](#)

... Unlike Peters et al. (2018a) and Radford et al. (2018), we do not use traditional left-to-right or right-to-left language models to **pre-train BERT**. Instead, we **pre-train BERT** using two unsupervised tasks, described in this section. This step is presented in the left part of Figure 1 ...

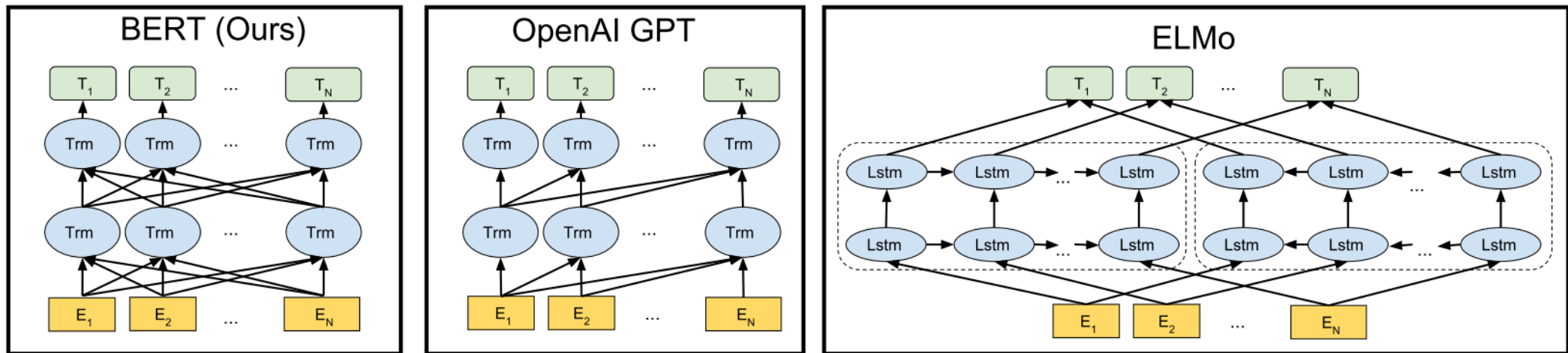
☆ 77 Cited by 28707 Related articles All 30 versions ⇔

- BERT is a pre-training model using deep bidirectional transformers for **language understanding**. (*encoder part of Transformer*)
- It uses the idea of **self-supervised learning**, rather than training on any specific NLP task.
- After we obtain the BERT pre-trained model, we can fine-tune it for a specific NLP task. (*given a sentence, we can obtain a representation from BERT and use it for downstream tasks*)



# BERT Model Architecture

- BERT's model architecture is a multi-layer bidirectional Transformer encoder.
- The input is word embedding and output is context sensitive word representation.





# Input representation

- Positional embeddings are learnable, rather than fixed magic number as in the Transformer paper.
- Each input sequence is a pair of sentences, separated by the token `[SEP]`. It adopted two learnable embeddings to each sentence.
- `[CLS]` is the a special classification embedding for the first token of every sequence.

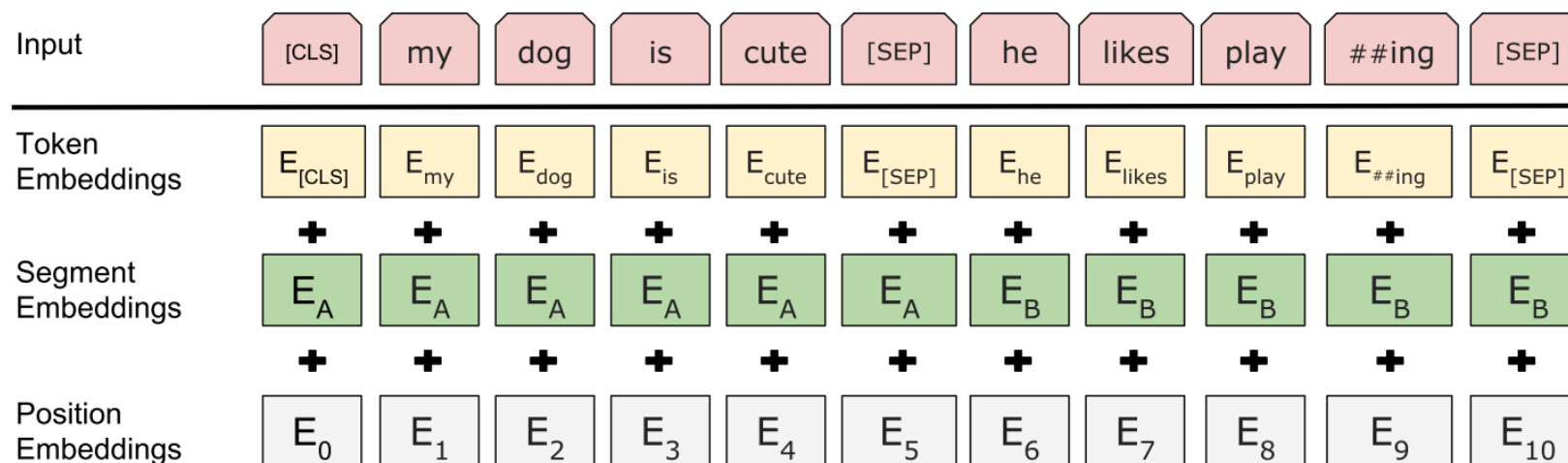


Image source: Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

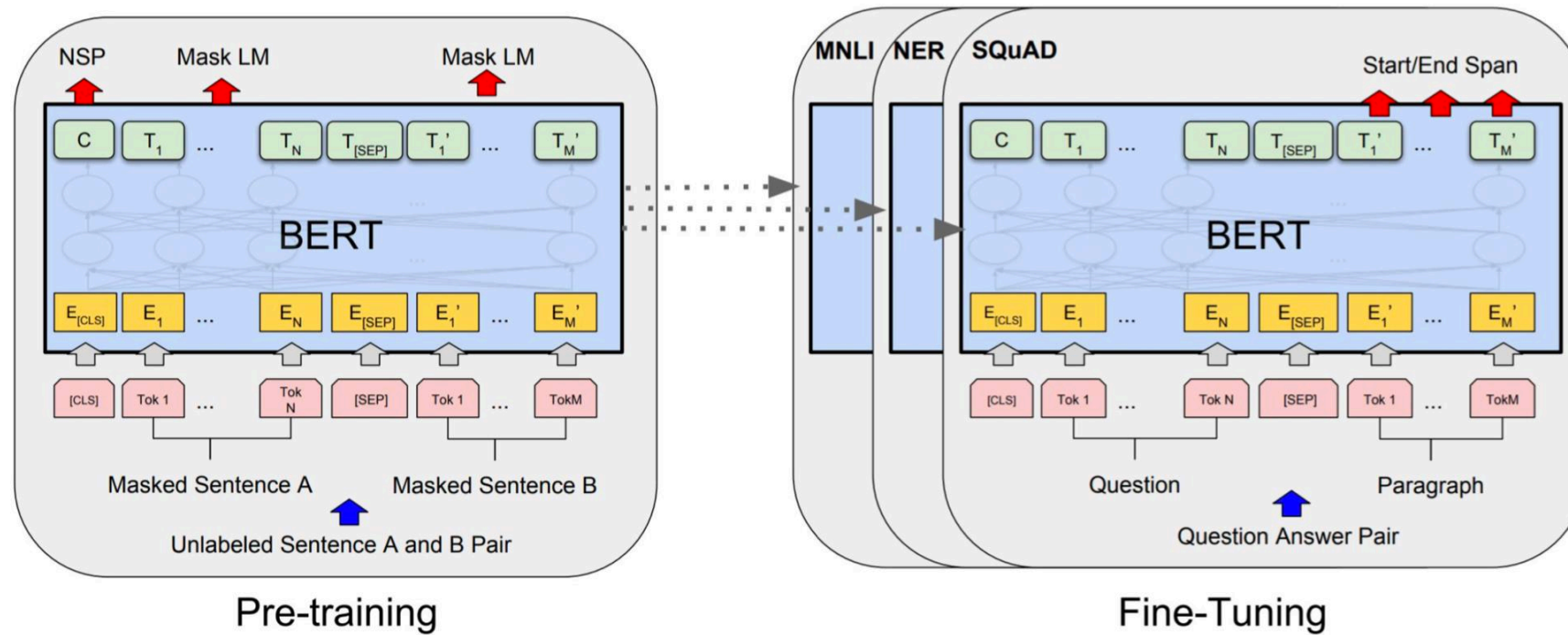
# Pre-Training Task 1: Masked LM

- Mask some percentage of the input tokens at random, and then predicting only those masked tokens.
- Use [MASK] token to replace 15% tokens randomly, and use the real token as the label to make it predict.
- However, the [MASK] token is never seen during fine-tuning. The authors proposed the following strategy:
  - 80% of the time: Replace the word with the [MASK] token.
    - e.g., my dog is hairy → my dog is [MASK].
  - 10% of the time: Replace the word with a random word.
    - e.g., my dog is hairy → my dog is apple.
  - 10% of the time: Keep the word unchanged. The purpose of this is to bias the representation towards the actual observed word.
    - e.g., my dog is hairy → my dog is hairy.

# Pre-Training Task 2: Next Sentence Prediction

- Make the model understand the relationship between two text sentences.
- Choose the sentences A and B for each pre-training example.
  - 50% of the time B is the actual next sentence that follows A.
  - 50% of the time it is a random sentence from the corpus.
- Example:
  - **Input** = [CLS] the man went to [MASK] store [SEP] he bought a gallon [MASK] milk [SEP]  
**Label** = IsNext
  - **Input** = [CLS] the man [MASK] to the store [SEP] penguin [MASK] are flight ##less birds [SEP]  
**Label** = NotNext

# Pre-Train and Fine-Tune



# GPT-3

## Language models are few-shot learners

[TB Brown](#), [B Mann](#), [N Ryder](#), [M Subbiah](#)... - arXiv preprint arXiv ..., 2020 - arxiv.org

Recent work has demonstrated substantial gains on many NLP tasks and benchmarks by pre-training on a large corpus of text followed by fine-tuning on a specific task. While typically task-agnostic in architecture, this method still requires task-specific fine-tuning datasets of thousands or tens of thousands of examples. By contrast, humans can generally perform a new language task from only a few examples or from simple instructions—something which current NLP systems still largely struggle to do. Here we show that scaling ...

☆ 🔖 Cited by 180 Related articles All 2 versions 🔗

- **Generative** Pre-trained Transformer 3 (GPT-3) is also a Transformer-based pre-trained language model. (**decoder part** of Transformer)
- It is the third generation in the GPT-n series created by OpenAI.
- It only uses the decoder part of Transformer (“**decoder-only**”).
- It has 175 billion parameters.

# GPT-n

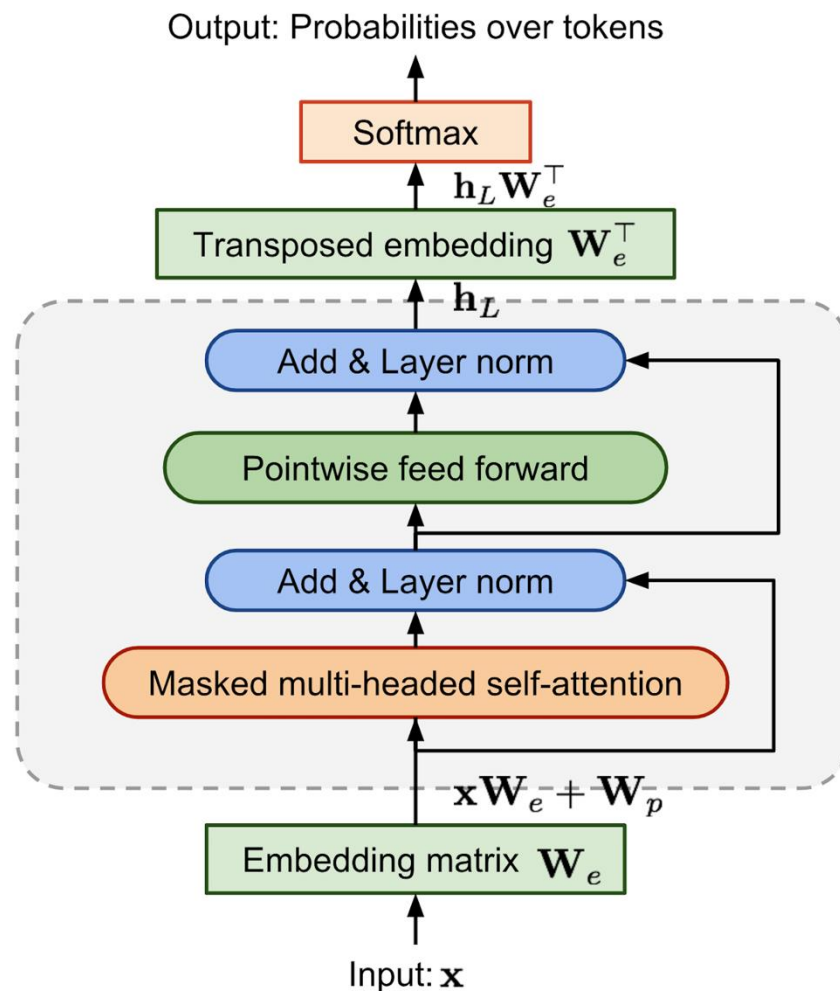
- Input is only one sentence.
- Predict the next word using previous words. Loss:

$$\mathcal{L}_{\text{LM}} = - \sum_i \log p(x_i \mid x_{i-k}, \dots, x_{i-1})$$

- During inference, input a text prompt, e.g.,

Once upon a time in a faraway  
land,

- GPT then autoregressively predicts the next word.



**Transformer Block**  
Repeat x L=12

$$\mathbf{h}_\ell = \text{transformer\_block}(\mathbf{h}_{\ell-1})$$
$$\ell = 1, \dots, L$$

# GPT-n

Have almost identical (but larger) architecture

ChatGPT: similar architecture with GPT, finetuned for conversations

OpenAI's *GPT-n* series

Model	Architecture	Parameter count	Training data	Release date	Training cost
GPT-1	12-level, 12-headed Transformer decoder (no encoder), followed by linear-softmax.	117 million	<a href="#">BookCorpus</a> : <sup>[39]</sup> 4.5 GB of text, from 7000 unpublished books of various genres.	June 11, 2018 <sup>[9]</sup>	30 days on 8 P600 <a href="#">GPUs</a> , or 1 petaFLOP/s-day. <sup>[9]</sup>
GPT-2	GPT-1, but with modified normalization	1.5 billion	WebText: 40 GB of text, 8 million documents, from 45 million webpages upvoted on <a href="#">Reddit</a> .	February 14, 2019 (initial/limited version) and November 5, 2019 (full version) <sup>[40]</sup>	"tens of petaflop/s-day", <sup>[41]</sup> or 1.5e21 FLOP. <sup>[42]</sup>
GPT-3	GPT-2, but with modification to allow larger scaling	175 billion <sup>[43]</sup>	499 billion tokens consisting of <a href="#">CommonCrawl</a> (570 GB), WebText, English Wikipedia, and two books corpora (Books1 and Books2).	May 28, 2020 <sup>[41]</sup>	3640 petaflop/s-day (Table D.1 <sup>[41]</sup> ), or 3.1e23 FLOP. <sup>[42]</sup>
GPT-3.5	Undisclosed	175 billion <sup>[43]</sup>	Undisclosed	March 15, 2022	Undisclosed
GPT-4	Also trained with both text prediction and <a href="#">RLHF</a> ; accepts both text and images as input. Further details are not public. <sup>[38]</sup>	Undisclosed. Estimated 1.7 trillion. <sup>[44]</sup>	Undisclosed	March 14, 2023	Undisclosed. Estimated $2.1 \times 10^{25}$ FLOP. <sup>[42]</sup>

Source: [https://en.wikipedia.org/wiki/Generative\\_pre-trained\\_transformer](https://en.wikipedia.org/wiki/Generative_pre-trained_transformer)

# ChatGPT: Chatbot Built Upon GPT-3.5/GPT-4

- There are three stages of training ChatGPT:
  - Generative pre-training: predicting  $p_{\theta}(X_{t+1} = x_{t+1} | x_1, \dots, x_t)$ .
  - Supervised fine-tuning (using “prompts”).
  - Reinforcement learning from human feedback.
- Explanation with visualization: <https://www.youtube.com/watch?v=VPRSBzXzavo>



# Prompt Engineering in ChatGPT

**Prompt:**

Q: Explain what is meant by time complexity.

A: Time complexity refers to the amount of time an algorithm takes to run as a function of the size of the input. It is typically expressed using big O notation.

Q: Explain how the bubble sort algorithm works.

A:

# Prompt Engineering in ChatGPT

- Increasingly important in practical use of generative AI

## Bloomberg

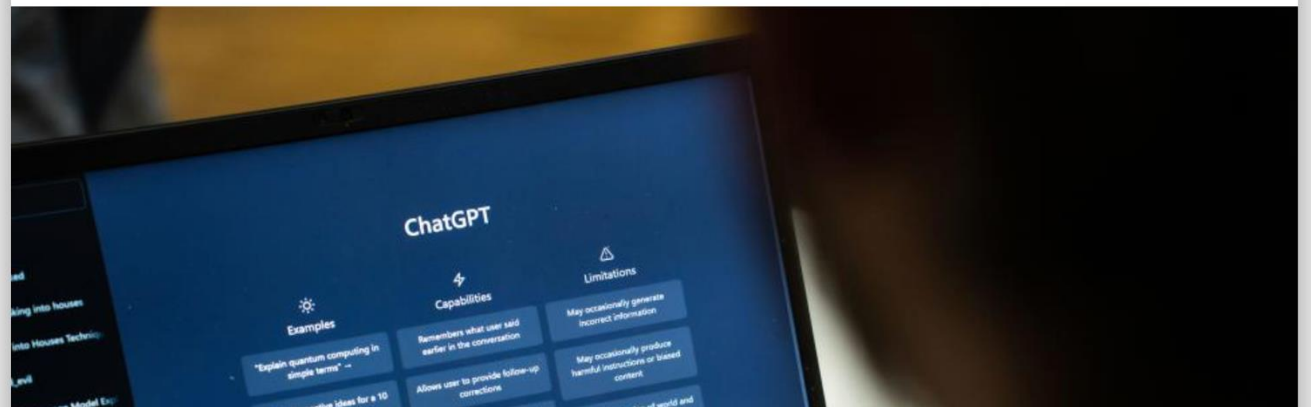
Work Shift

### **\$335,000 Pay for 'AI Whisperer' Appears in Red-Hot Market**

The fast-growing apps have created a seller's market for artificial intelligence grads — capable of manipulating its output.

### **AI 'prompt engineer' jobs can pay up to \$375,000 a year and don't always require a background in tech**

May 1, 2023



# Prompt Engineering in ChatGPT

- Prompt engineering: writing and optimizing prompts in a structured way.
- Goal: communicate with LLM to steer its behavior for desired outcomes *without* updating the model weights.
- Best practices:
  - Clear instructions
  - Adopt a persona
  - Specify the format
  - Avoid leading the answer
  - Limit the scope

"Isn't it true that electric cars are far better for the environment than traditional gasoline cars?"

✗ **Figure: Bad example - The prompt is leading and suggests a specific answer**

"What are the environmental benefits and drawbacks of electric cars compared to traditional gasoline cars?"

✓ **Figure: Good example - The prompt is neutral and encourages a balanced comparison**

# Prompt Engineering in ChatGPT

- Basic Prompting

- Zero-shot: simply feed in the task test to the model and ask for results.

Text: i'll bet the video game is a lot more fun than the film.  
Sentiment:

- Few-shot: presenting some high-quality demonstrations first. The model first sees good examples, it can better understand the human intention and expectation.

Text: (lawrence bounces) all over the stage, dancing, running, sweating, mopping his face and generally displaying the wacky talent that brought him fame in the first place.  
Sentiment: positive

Text: despite all evidence to the contrary, this clunker has somehow managed to pose as an actual feature movie, the kind that charges full admission and gets hyped on tv and purports to amuse small children and ostensible adults.  
Sentiment: negative

Text: for the first time in years, de niro digs deep emotionally, perhaps because he's been stirred by the powerful work of his co-stars.  
Sentiment: positive

Text: i'll bet the video game is a lot more fun than the film.  
Sentiment:

# Prompt Engineering in ChatGPT

- Instruction Prompting

- Few-shot examples can be expensive in token usage and limits our input length.
- Give instructions directly to explain our intent to the model.

Please label the sentiment towards the movie of the given movie review. The sentiment label should be "positive" or "negative".  
Text: i'll bet the video game is a lot more fun than the film.  
Sentiment:

- Be *specific* and *precise*!
- Combining instruction and few-shot examples:

Definition: Determine the speaker of the dialogue, "agent" or "customer".  
Input: I have successfully booked your tickets.  
Output: agent

Definition: Determine which category the question asks for, "Quantity" or "Location".  
Input: What's the oldest building in US?  
Output: Location

Definition: Classify the sentiment of the given movie review, "positive" or "negative".  
Input: i'll bet the video game is a lot more fun than the film.  
Output:

# Prompt Engineering in ChatGPT

- Chain-of-Thought (CoT) Prompting

- Reasoning chains: a sequence of short sentences to describe reasoning logics step by step.
- CoT is useful for complicated reasoning tasks (simple tasks benefit less).

- Zero-shot CoT:

Question: ...  
Answer: Let's think step by step.

- Few-shot CoT: a few examples containing high-quality reasoning chains.

Question: Tom and Elizabeth have a competition to climb a hill. Elizabeth takes 30 minutes to climb the hill. Tom takes four times as long as Elizabeth does to climb the hill. How many hours does it take Tom to climb up the hill?

Answer: It takes Tom  $30 \times 4 = 120$  minutes to climb the hill.

It takes Tom  $120/60 = 2$  hours to climb the hill.

So the answer is 2.

===

Question: Jack is a soccer player. He needs to buy two pairs of socks and a pair of soccer shoes. Each pair of socks cost \$9.50, and the shoes cost \$92. Jack has \$40. How much more money does Jack need?

Answer: The total cost of two pairs of socks is  $9.50 \times 2 = 19$ .

The total cost of the socks and the shoes is  $19 + 92 = 111$ .

Jack need  $111 - 40 = 71$  more.

So the answer is 71.

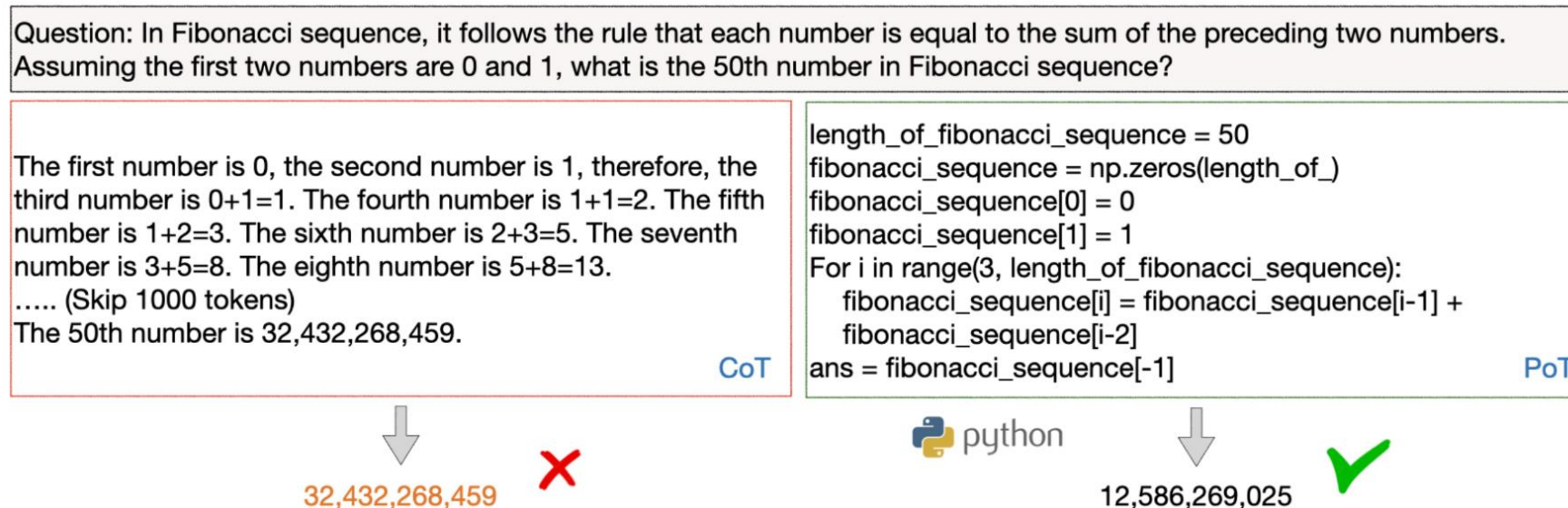
===

Question: Marty has 100 centimeters of ribbon that he must cut into 4 equal parts. Each of the cut parts must be divided into 5 equal parts. How long will each final cut be?

Answer:

# Prompt Engineering in ChatGPT

- Program of Thoughts (PoT) Prompting
  - Ask LLM to generate programs to solve a natural language reasoning problem.
  - Offloads the solution step to a runtime like a Python interpreter.



# Using ChatGPT APIs

- Application Programming Interface (API): a way for two or more computer programs to communicate with each other.
- You can enable your own applications to communicate with ChatGPT using API.
- Integrate the power of ChatGPT into your own applications or products.
- More precise control over the conversation flow.



# Using ChatGPT APIs

- Get the API Key

- Go to <https://genai.hkbu.edu.hk/> and log in.
- Choose “API” from the menu.
- Press the “Generate” button and copy the key.
- Do NOT disclose the key! Keep it safe.

The screenshot displays the HKBU GenAI Platform interface. At the top, the HKBU logo and 'HKBU GenAI Platform' are visible. A user is signed in as 'cskjyin@hkbu.edu.hk'. A dropdown menu is open, showing options like 'FAQ', 'API' (highlighted with a red box and number 2), 'Prompt Engineering & Temperature', 'Terms & Conditions', 'Privacy Policy', 'Feedback', 'Contact Us', and 'Sign out'. Below the menu, a section titled 'HKBU GenAI Platform API Service' provides information about the REST API endpoint. It states that the key is a unique identifier and provides access to the account, and that the API usage will be monitored. A table lists supported versions of swagger specification for more guidance. The table has two columns: 'MODEL' and 'SPECIFICATION'. Under 'MODEL', 'GPT' is listed. Under 'SPECIFICATION', '2024-02-01' and '2024-05-01-preview' are listed, with '2024-02-01' highlighted by a red box and the text 'available models' next to it. Below the table, instructions state that to obtain a key, the user should click on the 'Generate' button, which is highlighted with a red box and number 1. The key is displayed in a text box, and a copy icon is highlighted with a red box and number 2. A 'Close' button is also visible.

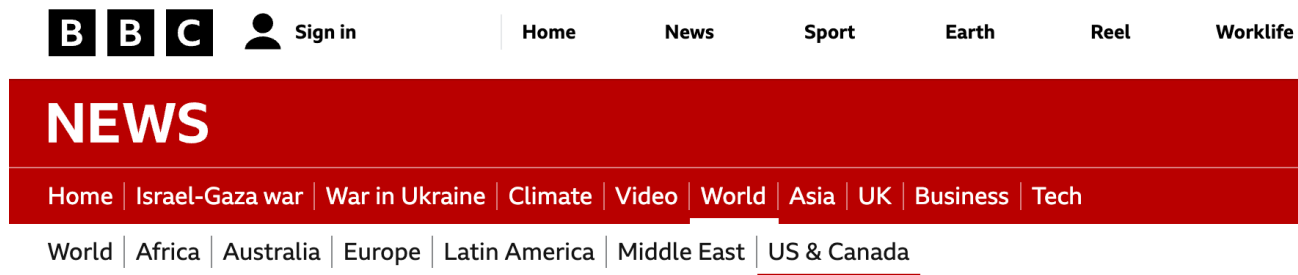
MODEL	SPECIFICATION
GPT	2024-02-01 2024-05-01-preview

# Using ChatGPT APIs

- Full documentation: <https://learn.microsoft.com/en-us/azure/ai-services/openai/reference>
- For Chat completions: <https://learn.microsoft.com/en-us/azure/ai-services/openai/reference#chat-completions>
- Python Example: <https://genai.hkbu.edu.hk/general/hkbu-chatgpt-api#python>
- *[live demo]*
- Rapid iteration: programming your ideas → Refining your prompts.

# Risks, Responsibilities and Ethics of Generative AI

- Generating content that is not factually correct: use with extra caution!
  - Who will be responsible for the consequences?
  - Use with extra caution! Never over trust generative AI models.



*Example: US lawyer caught citing a non-existent case in court.*

*Presenting false information to courts can be a criminal offense.*

# Risks, Responsibilities and Ethics of Generative AI

- Copyright, privacy, security, and legal issue
  - Who own the copyright of the generated content?
  - If the model plagiarizes someone else's work, who will be responsible?
  - Training data may contain private information.
    - Some GitHub repos contain SSH keys added to commit history by accident. Generative AI trained using GitHub resources could leak the SSH keys.
    - Private data is protected by law. Use without authorization is illegal: DPR, Hong Kong Personal Data (Privacy) Ordinance
- AIGC service provider may retain your data for model training! *Privacy risks!*

# Risks, Responsibilities and Ethics of Generative AI

- Distribution of harmful content
  - A scammer cloned the voice of a young girl to demand ransom from her family by faking a kidnapping.
  - How to avoid such unethical and unlawful content? *Open challenge...*
  - Detecting such harmful content using AI methods?
- We should only use AI in a way that moves humanity forward.
- Always think twice the potential ethical issues!

# Generative AI for Images: DALL-E 3



<https://www.youtube.com/watch?v=sqQrN0iZBs0>

# Generative AI for Images: Stable Diffusion



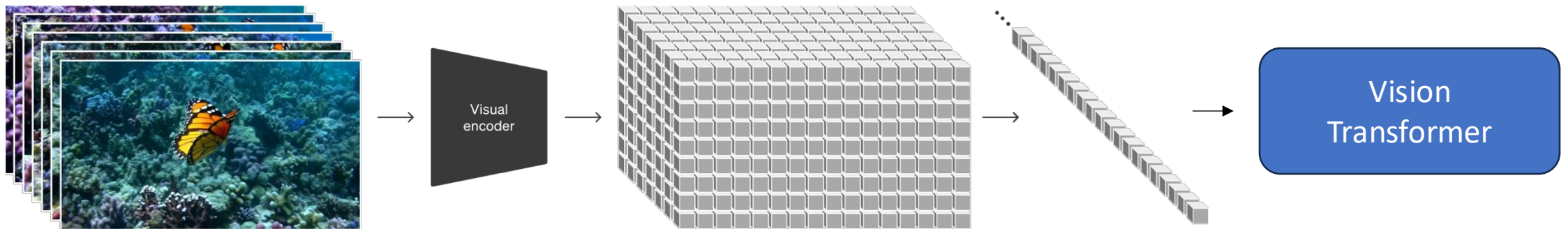
<https://www.youtube.com/watch?v=nVhmFski3vg>

Stable-diffusion-webUI: <https://github.com/AUTOMATIC1111/stable-diffusion-webui>



# Sora: Video Generation

- Demo: <https://openai.com/index/video-generation-models-as-world-simulators/>
- Divide video frames into patches. Each patch act as a “word” in Transformer.



Source: <https://openai.com/index/video-generation-models-as-world-simulators/>



# COMP7015 Course Review

Credits: this section is partially adapted from Lecture 6 of CS440/CEC448 (UIUC) under CC-BY-4.0 license

# Artificial Intelligence

- Searching
- Knowledge Representation & Logics
- Machine Learning
- Deep Learning
- Reinforcement Learning
- Bayesian Learning
- Generative AI

# I. Searching

- **How to formulate a search problem:** states, initial state, goal state, & actions
- **Uninformed Search (blind search):**
  - BFS, graph DFS, uniform-cost search
  - Tree-like search: tree-like DFS, depth-limited search, iterative deepening search
- **Informed Search (heuristic search; have hints about the how close a state is to the goal state):**
  - Heuristic functions: admissible? Which is better?
  - Greedy search & A\* search
- **Constraint Satisfaction Problems (CSPs):**
  - Variables and domains
  - Backtracking with forward checking & Arc consistency
- **Adversarial Search:**
  - Minimax search algorithm, Alpha-beta pruning & MCTS
- **Which algorithm to use?**

## II. Knowledge Representation and Logics

- Goal: Represent and store information and derive conclusions from the available information.  
(example: virtual assistant)
- Why logics? Natural language is not precise. Need a precise way to store and process knowledge.
- Ingredients of logic: Syntax, Semantics, and Inference Rules.
- Propositional Logics
  - Syntax: propositional atoms and propositional connectives
  - Semantics: models, satisfaction, truth table
  - Knowledge base: entailment, satisfiability, Tell and Ask operations.
  - Inference Rules: Modus Ponens, Resolution
- First-order Logics
  - Syntax and Semantics: quantifiers (universal and existential)
  - Inference Rules: generalized Modus Ponens rule

# III. Machine Learning (1/2)

- Supervised Machine Learning Algorithms

- Linear regression, logistic regression: loss minimization framework, gradient descent
- Decision tree (ID3) algorithm
  - Entropy, information gain, information gain ratio
  - Bi-partition for continuous attributes

- Generalization and Model Selection

- How to evaluate model performance? Hold-out, repeated hold-out, K-fold, leave-one-out

- Performance Measures for Classification Problems

- Confusion matrix
- Metrics: accuracy, precision, recall, f1-scores

- Ensemble Learning:

- Why and when ensemble learning could help?
- Random forest.

# III. Machine Learning (2/2)

- Multi-class classification
  - OvO and OvR approach
  - Performance Evaluation
- Feature Engineering
  - Hypothesis class: a feature extractor (or feature mapping) determines the hypothesis class.
- Bayesian Decision Theory
  - Bayes' rule
  - Naïve Bayes classifier

# IV. Deep Learning

- Neural Networks: Feature (representation) learning
- Feedforward Neural Network
  - Neurons, layers, nonlinear activation functions
  - Computational graphs and backpropagation
  - Stochastic gradient descent
- Regularization Techniques
- Convolutional Neural Networks
  - Convolution, padding, stride, receptive field, pooling
  - Why CNN?
- Recurrent Neural Networks
  - Sequential data
  - RNN, vanilla RNN, LSTM
- Language Models: word embeddings
- Pre-trained CNN and language models
- Fine-tuning
- Generative AI models

# V. Reinforcement Learning

- **Markov Decision Process**
  - Why? Uncertainty
  - State, chance nodes, transition function, reward.
  - Policy evaluation: how good a given policy is?
    - Utility, discounting, value, Q-value.
  - Value iteration: what is the optimal value and optimal policy?
    - Optimal value, optimal policy, iterative algorithm
- **Reinforcement Learning**
  - Why? No access to transition function and reward.
  - Framework
  - Monte Carlo methods
    - Model-based: estimate the MDP
    - Model-free: estimate the Q-value directly.
  - Q-learning
  - $\epsilon$ -greedy algorithm: exploration vs exploitation



# The Final Exam



- Materials for preparation:
  - Slides (you can safely skip parts we did not cover in class)
  - Lab materials
  - Assignments (written/programming)
  - Quiz
- You can use a calculator.
- Give intermediate steps! You can get partial scores even if the end results are incorrect, but some steps are correct.
- Scan through the questions before start: not ordered by difficulty.