通过文心 4.5 学习大语言模型

大语言模型概述科普

什么是语言模型?

P(“I have a cute cat.”) = ?

- 可以**估算任意一句话的概率**的机器。
- 语言模型是一种对**词语序列概率相关性分布**的建模，用于评估文本序列的合理性。

P(“I have a cute cat.”) > P(“My cat can fly.”)

$$P(w_1,w_2,\cdots,w_n)=P(w_1)\cdot P(w_2 \mid w_1)\cdot P(w_3 \mid w_1,w_2)\cdot\cdots\cdot P(w_n \mid w_1,w_2,\cdots,w_{n-1})$$

语言模型能干什么?

- 下一个词预测： P("job" | "I love my")
- 情感分类： P(sentiment=Positive | "I love your product.")
- 主题分类： P(category="Sports" | "We won the soccer game.")
- 机器翻译： P(target="我爱我的猫" | source="I love my cat.")
- 对话生成： P(assistant="I am a robot." | user="who are you?")
- 相关性排序： P(relevance=True | query="best restaurants", document="Top 10 Restaurants in Beijing.")

语言模型是完成上述所有任务的基础。然而，为了在每个具体任务上取得最佳效果，我们通常**需要一个在此基础上构建的专用模型**。

大语言模型带来的范式转变：一切都是文本生成！

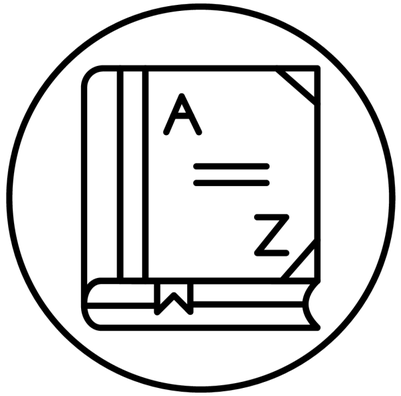
- P("I love my job")
- P("I love your product. This sentiment is: Positive")
- P("We won the soccer game. The topic is: Sports")
- P("Translate 'I love my cat.' to Chinese: 我爱我的猫")
- P("User: who are you? Assistant: I am a robot.")
- P("Query: best restaurants. Document: Top 10 Restaurants in Beijing. Are they relevant? Yes")

大语言模型不再需要为每个任务设计专用模型，而是可以**把所有问题都重新表述为文本生成的任务**。即：AGI曙光初现！

大语言模型如何生成文本?

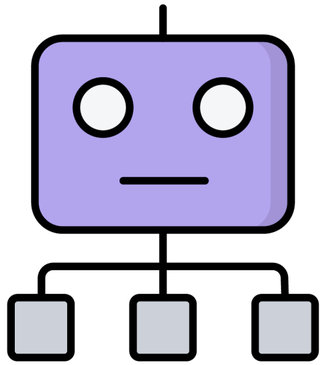
一部字典

包含了世界上所有可能的单词。



一个神奇的机器

会告诉你字典里的单词组成的句子的概率。

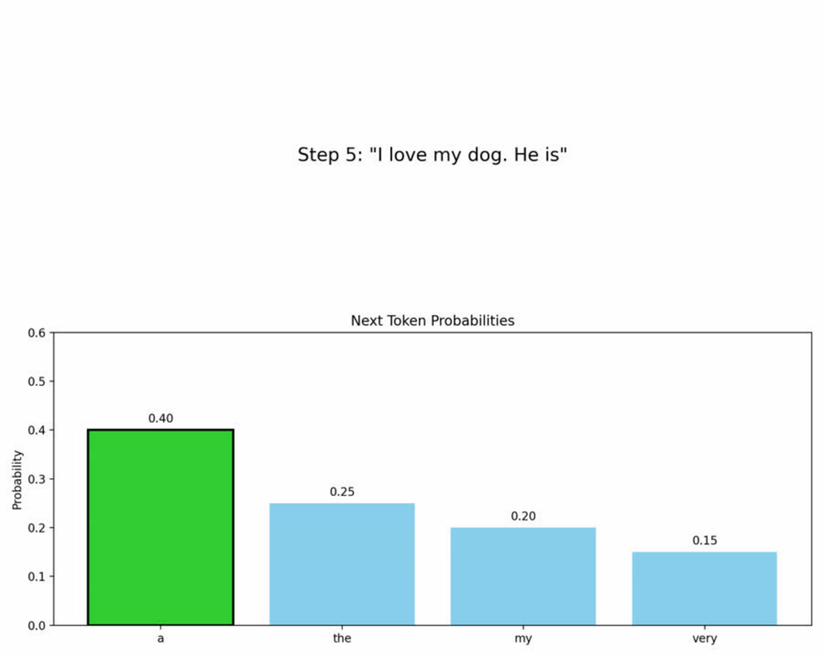


我有个故事的开头：I love my ___

大语言模型的解码策略

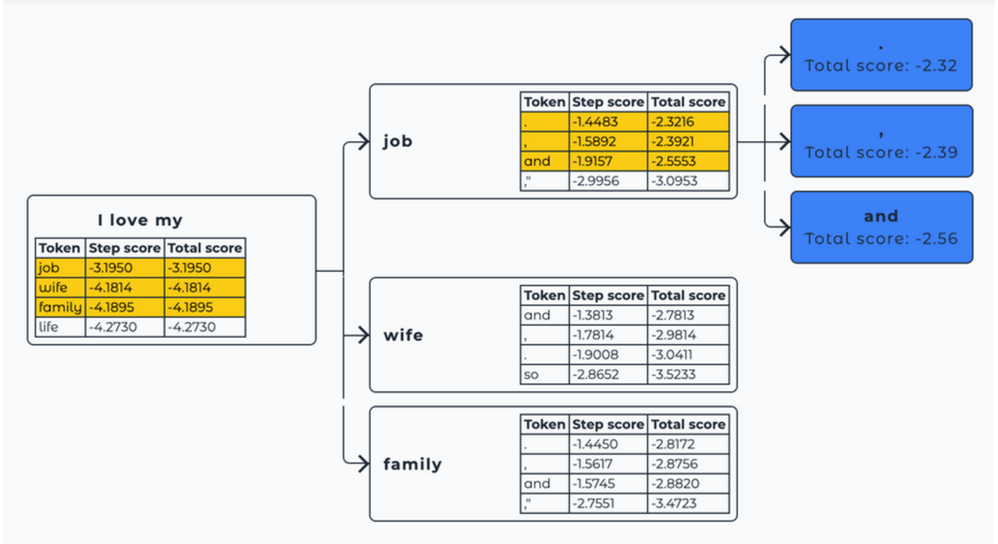
Greedy Search

只看眼前，选择最优：每次都挑选概率最大的词。



Beam Search

多条路一起走：同时追踪多个(beam size)最有希望的候选句子。

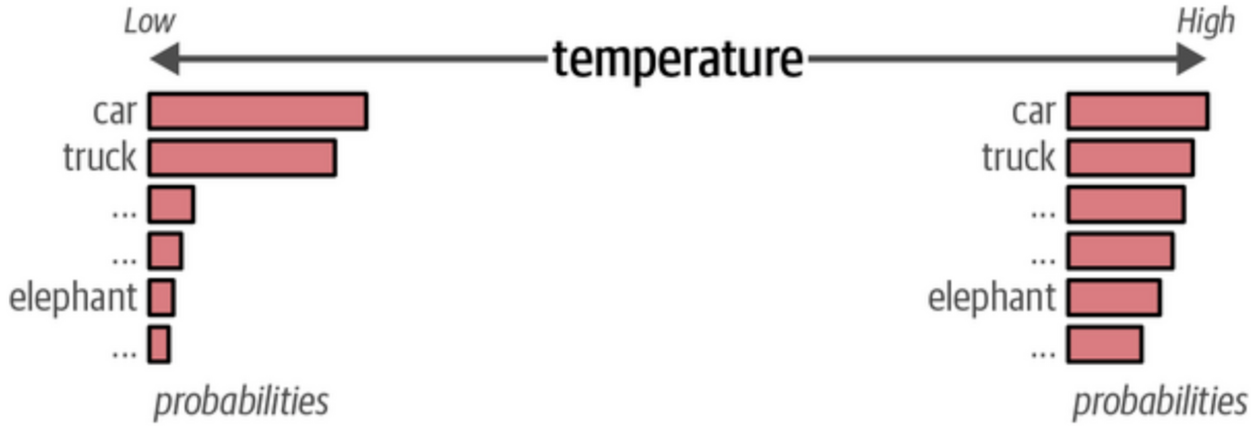


https://huggingface.co/spaces/m-ric/beam_search_visualizer

大语言模型的解码策略

Sampling Strategy

从当前步骤的候选词列表里随机选出一个词。让模型更有多样性和创造性。



temperature: 调整概率分布，越低越尖锐，越高越平缓。

Top-p (nucleus) sampling：从概率总和超过 p 的子集里随机挑选下一个词。

Top-k sampling：从概率值排名前 k 的子集里随机挑选下一个词。

大语言模型的解码策略

Quick question

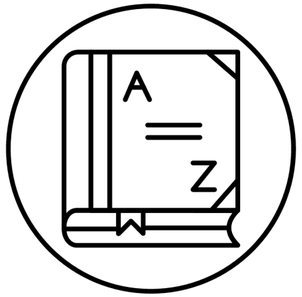
Greedy Search =
Beam search, when beam is ? =
Sampling strategy, when temperature is ?

https://huggingface.co/blog/how-to-generate

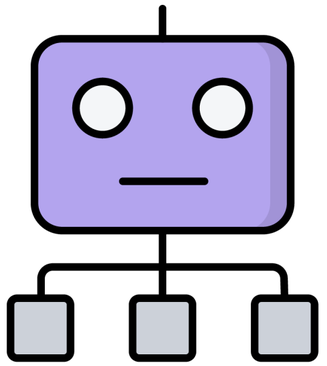
如何得到大语言模型?

I love my cat. That’s probably the most consistent truth in my life, a sturdy anchor in a sea of ever-shifting schedules and responsibilities. Her name is Luna, and she's a sleek, midnight-black creature with eyes like polished emeralds. Right now, she was curled on my lap, a warm, purring weight that chased away the chill of the evening. Outside, the rain lashed against the window, a relentless drumbeat that usually amplified my anxieties. But with Luna there, a soft, vibrating engine of contentment, the storm seemed distant, almost cozy. I ran a hand over her silky fur, feeling the rhythmic rise and fall of her tiny chest. It was a simple moment, one of hundreds just like it, yet each one felt precious, a quiet testament to the profound bond we shared.

一部字典（vocab）



一个神奇的机器（model）



从无标注的海量文本中训练得到大语言模型

Statistical language model

Counting words!

- unigram: $P(w_1)$ = w1 出现的次数 / 所有词出现的次数。
- bigram: $P(w_2 \mid w_1)$ = <w1, w2> 出现的次数 / w1 出现的次数。
- trigram: $P(w_3 \mid w_1,w_2)$ = <w1, w2, w3> 出现的次数 / <w1, w2> 出现的次数。
- n-gram: $P(w_n \mid w_1,w_2,...,w_{n-1})$ = <w1, w2, ..., wn> 出现的次数 / <w1, w2, ..., wn-1> 出现的次数。

Limitations

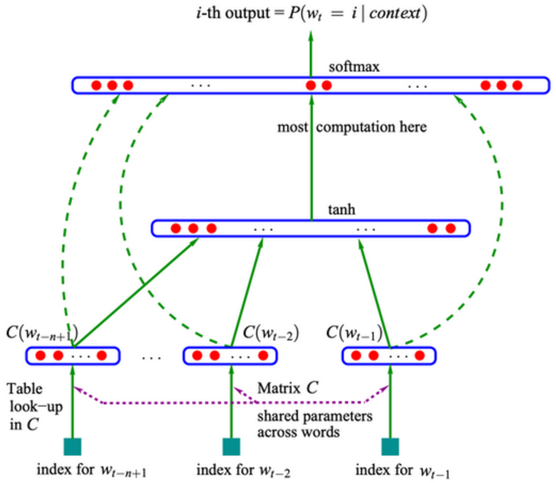
- **Limited memory：** n-gram 中的 n 在现实里一般到 3。
- **Sparsity：**不知道怎么处理训练数据中没出现过的词的组合，或者即便见过，出现的次数很少。

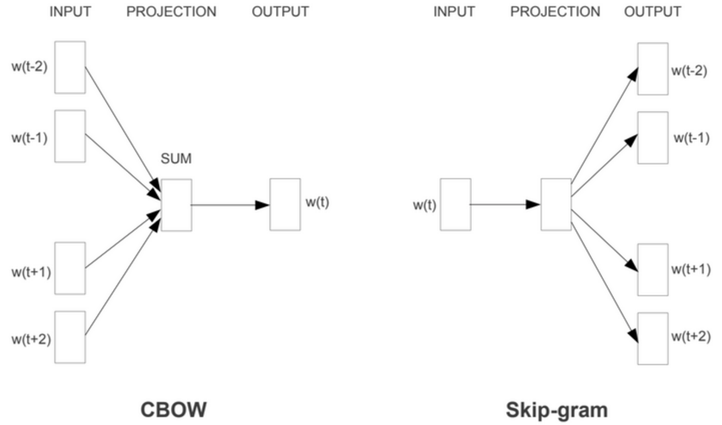
NNLM and word2vec

Embedding comes to rescue!

把一个词转换为 embedding (vector)，从而可以喂给神经网络。

- cat → [0.2, -0.5, 0.8, ...]
- dog → [0.3, -0.6, 0.7, ...]





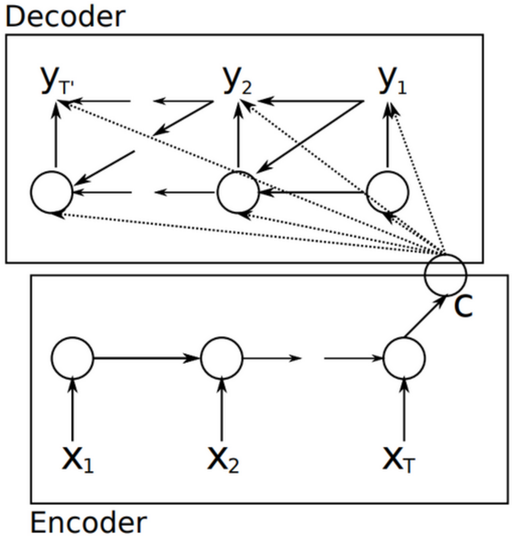
NNLM: https://www.jmlr.org/papers/volume3/tmp/bengio03a.pdf word2vec: https://arxiv.org/pdf/1301.3781

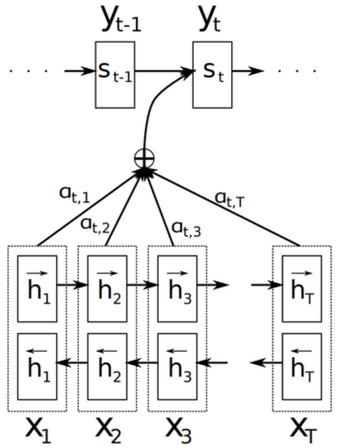
- embedding 为用神经网络处理文本数据架设起了桥梁，可以理解为词的语义。
- 训练数据中未出现的词的组合，依然可以处理。
- **Limitations：**
  - BOW（Bag of words）会丢失词的顺序信息；NNLM 中的 n 不会太大，且记录的是绝对位置。

RNN with attention

Sequential nature of language!

RNN(Recurrent Neural Network): 带着上一个词的信息走向下一个词。
Attention: 注意力放在更相关的词上。





RNN encoder decoder: https://arxiv.org/pdf/1406.1078

Learn to align and translate: https://arxiv.org/pdf/1409.0473

机器翻译变得真正可用！

Limitations：

- **Vanishing gradient problem：**比较难保留很早之前的词的信息（尽管LSTM可以缓解）。
- **Sequential processing：**一次处理一个词，无法并行化。

Attention is all you need!

YES! Attention is all you need.

- 不再需要 RNN 结构，仅使用 attention 机制
  - An attention function can be described as **mapping a query and a set of key-value pairs to an output**, where the query, keys, values, and output are all vectors. The **output is computed as a weighted sum of the values**, where the **weight assigned to each value is computed by a compatibility function of the query with the corresponding key.**

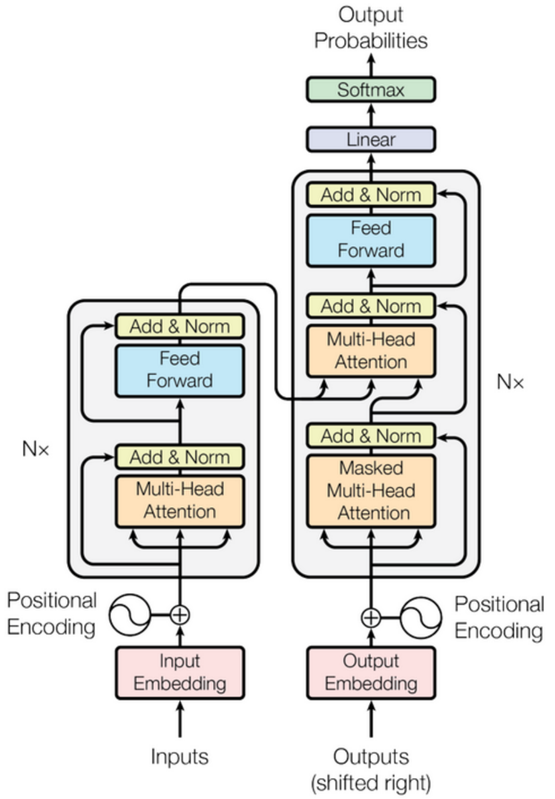


Figure 1: The Transformer - model architecture.

这是所有现代大语言模型的基石！
计算过程可以并行化！
不会忘掉前面的词的信息！

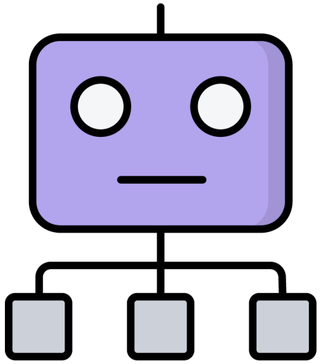
https://arxiv.org/pdf/1706.03762

Today’s LLM

- **Really Large: ERNIE4.5 的总参数最小 0.3B，最大 424B**
- **Multi-Modal: ERNIE4.5 可以同时处理文本和视觉**
- **Decoder-only: 只使用 transformer block**
- 结构改进
  - **MHA(multi head attention) → GQA(group query attention), etc.**
  - **FFN(feed forward network) → MoE(mixture of attention)**
- 硬件改进
  - 低精度混合训练和推理
  - 硬件内置计算引擎
- 软件框架
  - 训练：PaddlePaddle, PyTorch, DeepSpeed, ...
  - 推理：FastDeploy, vLLM, sglang, ollama, ...

Recap

一个神奇的机器（model）



怎么用大模型生成文本

greedy search, beam search, sampling strategy.

怎么建造大模型

Use Transformer block, with large amounts of unlabeled text.

但目前为止还是只能做文本补全，不能做问答！
It’s a pretrained based model, we need post-training!

- P("I love my job")
- P("I love your product. This sentiment is: Positive")
- P("We won the soccer game. The topic is: Sports")
- P("Translate 'I love my cat.' to Chinese: 我爱我的猫")
- P("User: who are you? Assistant: I am a robot.")
- P("Query: best restaurants. Document: Top 10 Restaurants in Beijing. Are they relevant? Yes")

Post training in LLM

Follow instruction and chat

SFT(supervised fine-tuning): 在<query, response> 构成的数据上后训练模型，让模型学会回答问题，并遵守指令。

Align with human value

DPO(Direct Preference Optimization) : 在<query, chosen response, rejected response> 构成的数据上后训练给模型打上思想钢印，来实现：Helpful, Honest, and Harmless.

Improve performance

RL(Reinforcement learning) : 模型先生成一个输出，然后根据人类反馈（Human feedback）或自动反馈（Reward model）获得奖励分数，然后更新模型以提升性能，通常会提升模型在推理、数学等领域的效果。

https://www.deeplearning.ai/short-courses/post-training-of-llms/

ERNIE 4.5 as an example

Model	Multimodal	MoE	Post-Trained	Thinking/Non-Thinking Mode
ERNIE-4.5-300B-A47B-Base	✗	✓	✗	-
ERNIE-4.5-300B-A47B	✗	✓	✓	non-thinking
ERNIE-4.5-21B-A3B-Base	✗	✓	✗	-
ERNIE-4.5-21B-A3B	✗	✓	✓	non-thinking
ERNIE-4.5-0.3B-Base	✗	✗	✗	-
ERNIE-4.5-0.3B	✗	✗	✓	non-thinking
ERNIE-4.5-VL-424B-A47B-Base	✓	✓	✗	-
ERNIE-4.5-VL-424B-A47B	✓	✓	✓	both
ERNIE-4.5-VL-28B-A3B-Base	✓	✓	✗	-
ERNIE-4.5-VL-28B-A3B	✓	✓	✓	both

Announcing the Open Source Release of the ERNIE 4.5 Model Family

Assignment

- 给定一句话，如：“hello, world”，用 ernie4.5-0.3b 模型计算这句话的概率。
- ernie4.5-0.3b 的 base model 和 instruct model 分别计算这句话的概率，哪个概率高？为什么？
- 进阶：自己实现一个 greedy search 解码器，自己的解码器的输出结果，跟调用 API 输出的结果一致。

Have a nice day!