Understanding the Workloads: Part 2

Lecture 4 for Advanced Deep Learning Systems

Aaron Zhao, Imperial College London, a.zhao@imperial.ac.uk

Table of contents

Introduction

Introduction - Outline

Using the basic networks and building blocks we learned in the previous lecture, we will look at how researchers have constructed

- LLAMA2
- CLIP
- SAM
- Whisper

GPT's training detail is not open-sourced, and it is (or should be) somehow very similar to the OPT and LLaMA models that we have looked at.

LLaMA2

LLaMA2

LLaMA2 normally refers to the LLaMA2 model, LLaMA2-Chat refers to the chatbot.

What is the difference?

- LLaMA2 is fully trained with only the pre-training data.
- LLaMA2-Chat requires additional treatment such as an iterative refinement using Reinforcement Learning with Human Feedback (RLHF).

This is almost the same for all state-of-the-art LLMs! There are additional steps required to make them a good chatbot!

LLaMA2 pre-training details

Architecture details

- Standard Transformer architecture
- Pre-normalization with RMSNorm
- Rotary positional embedding (RoPE)
- Grouped-query attention (GQA), will cover in more detail in the next lecture

Training details

- 2 trillion tokens training data!
- AdamW optimizer
- Cosine learning rate scheduler with warmup

Normalization

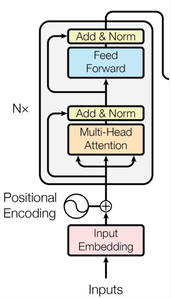
Pre- and Post-Normalization

Post-Normalization

$$x = Norm(x + f(x)) \tag{1}$$

Pre-Normalization

$$x = x + f(Norm(x)) \tag{2}$$



LayerNorm and RMSNorm

LayerNorm: $y = \frac{x-\mu}{\sigma}\alpha$

μ is the mean and σ is the std, α is a learnable gain parameter.

RMSNorm: $y = \frac{x}{RMS(x)}\alpha$

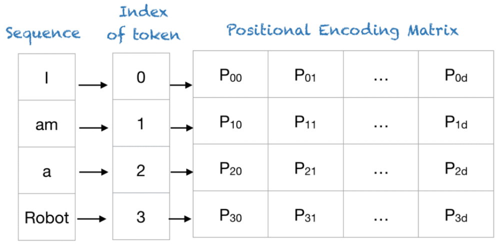
$RMS(x) = \sqrt{\frac{1}{n}\sum_{i=0}^{n-1} x_i^2}$

RMSNorm can be seen as a simplified LayerNorm, and also more efficient.

Positional Embedding

The original input is a sequence of word embedding $X_{embed} \in \mathcal{R}^{N \times D}$. Positional embedding $X_{pos} \in \mathcal{R}^{N}$, that is added to the word embedding to provide $X = X_{embed} + X_{pos}$.

Without positional embedding, the model cannot tell the difference between "I am a robot" and "am I a robot", because we take these inputs in parallel.

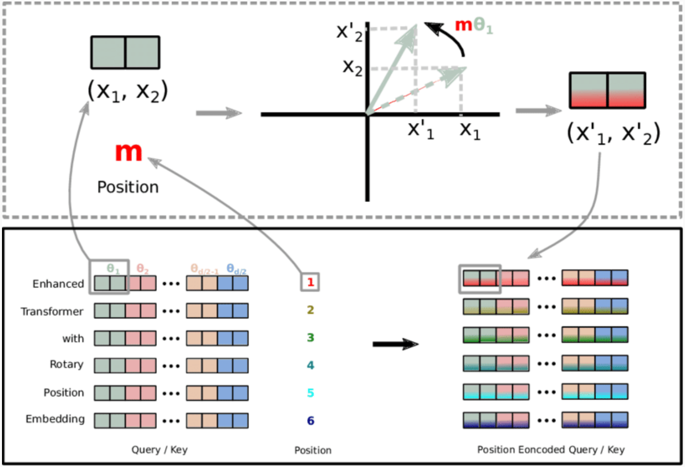


Positional Encoding Matrix for the sequence 'I am a robot'

Rotary Positional Embedding (RoPE)

- Absolute positional embedding: sinusoidal functions (no length constraints)
- Relative positional embedding: additional components added to the Q, K components, such as T5, slow at large sequences, no KV caching, not commonly used in large models.
- Rotary positional embedding (RoPE): think about angular changes

Rotary Positional Embedding (RoPE)



Contrastive Language–Image Pre-training (CLIP)

CLIP

Classic computer vision systems (such as ResNets) have the following disadvantages:

- Trained on labelled data.
- Normally predict a fixed set of predetermined object categories
- New labelled data is needed if you want to specify any other visual concept

Can we learn directly from raw text about images and the images?

CLIP

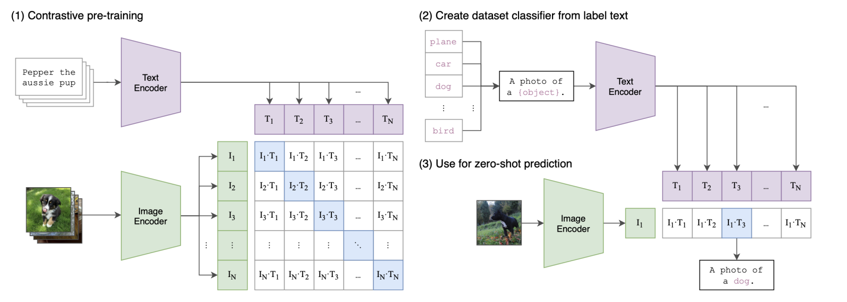
The core idea is to use natural language as supervision.

Contrastive representation learning works by predicting which of the $N \times N$ possible (image, text) pairings across a batch actually occurred.

This forces learning in a multi-modal embedding space by jointly training an image encoder and text encoder to

- maximize the cosine similarity of the image and text embeddings of the N real pairs in the batch
- minimize the cosine similarity of the embeddings of the $N^2 - N$ incorrect pairings.

CLIP


(1) Contrastive pre-training
Pepper the aussie pup
Text Encoder
Image Encoder
T_1 T_2 T_3 ... T_N
I_1 I_2 I_3 ⋮ I_N
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
$I_2 \cdot T_1$ $I_2 \cdot T_2$ $I_2 \cdot T_3$ ... $I_2 \cdot T_N$
$I_3 \cdot T_1$ $I_3 \cdot T_2$ $I_3 \cdot T_3$ ... $I_3 \cdot T_N$
$I_N \cdot T_1$ $I_N \cdot T_2$ $I_N \cdot T_3$ ... $I_N \cdot T_N$
(2) Create dataset classifier from label text
plane
car
dog
bird
A photo of a {object}.
Text Encoder
(3) Use for zero-shot prediction
Image Encoder
I_1
T_1 T_2 T_3 ... T_N
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
A photo of a dog.

CLIP - The Actual Code

```
# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]
# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)
# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)
# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss   = (loss_i + loss_t)/2
```

CLIP - Finetuning

Although in the original CLIP paper, they have presented a zero-shot prediction method for image classification.

We normally use the CLIP image encoder as a pre-trained model, connect it with a classifier to perform classification.

CLIP (contrastive learning) opened the door for building vision foundation models.

Segment Anything Model (SAM)

Segment Anything

A foundation model for image segmentation

- A large-scale dataset on the task: 1+ billon masks and 11 million images
- An Image Encoder: ViT model
- A Prompt Encoder
  - Sparse prompts (points, text): points are positional encoded using random spatial frequencies, text using CLIP.
  - Dense prompts (masks): feed into convolution to extract an embedding.
- A Mask Decoder: maps the image embedding, prompt embeddings, and an output token to a mask. We focus on this part.

Segment Anything

Panoptic segmentation

Image segmentation a computer vision and image processing technique that involves grouping or labeling similar regions or segments in an image on a pixel level.

- Semantic segmentation: Segments amorphous regions (or repeating patterns) of similar material, which is uncountable (e.g., road, sky, and grass).
- Instance segmentation: Segments countable objects in an image (e.g., people, flowers, birds, animals, etc.).
- Panoptic segmentation: Combines both.

Panoptic segmentation

Figure 1: Semantic Segmentation

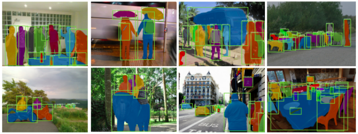


Figure 2: Instance Segmentation

Figure 3: Panoptic Segmentation

SAM Model Architecture

Image Encoder

Prompt Encoder

Mask Decoder

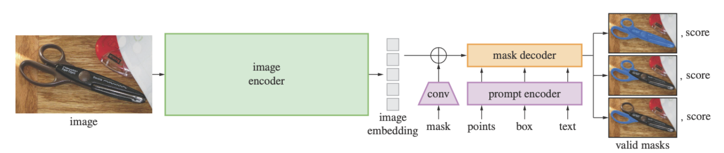


Image Encoder in SAM

Original Image in shape of 1024×1024

Pretrained Masked AutoEncoder

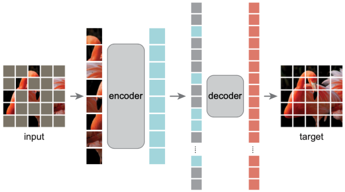


Figure 1. **Our MAE architecture**. During pre-training, a large random subset of image patches (*e.g.*, 75%) is masked out. The encoder is applied to the small subset of *visible patches*. Mask tokens are introduced *after* the encoder, and the full set of encoded patches and mask tokens is processed by a small decoder that reconstructs the original image in pixels. After pre-training, the decoder is discarded and the encoder is applied to uncorrupted images (full sets of patches) for recognition tasks.

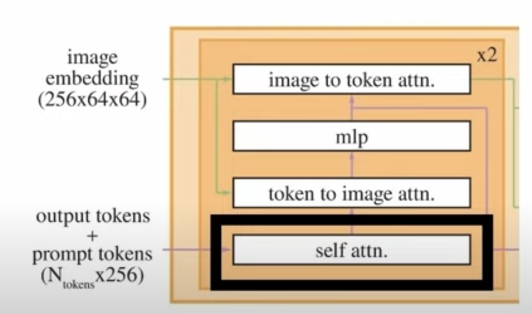
Transform to embedding $64 \times 64 \times 256$

Prompt Encoder in SAM

- Points and Bounding Box: positional encoding with trained embeddings
- Language inputs: CLIP model

Mask Decoder

- Self-attention (prompt tokens and output tokens)
- Cross-attention (prompt with image query): update prompt using contextual information from images.
- Cross-attention (image with prompt query): update image embedding using contextual information from prompts.



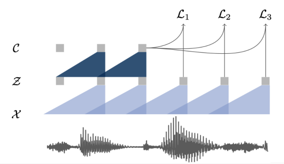
Whisper

Speech recognition

Speech recognition is a task of converting spoken language to text. It involves recognizing the words spoken in an audio recording and transcribing them into a written format (text).

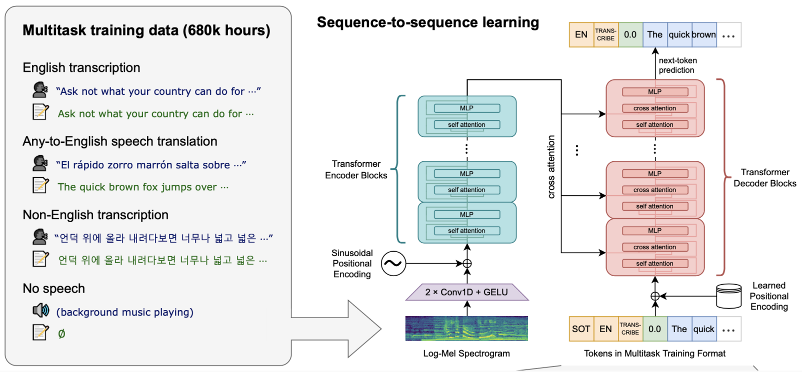
WAV2Vec

The ‘classic’ speech recognition network



- An encoder network: takes audio signal as inputs and project them to an embedding space using Convolutions
- A context network: combines multiple time-steps of the encoder to obtain contextualized representations also using Convolutions.
- Limited context length.

Whisper Architecture



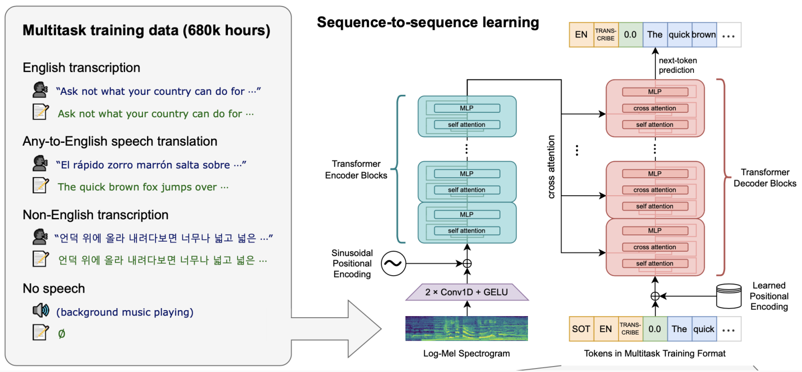
log-MelSepectrum

- Transform signal from the time domain to frequency domain
- Studies have shown that humans do not perceive frequencies on a linear scale. We are better at detecting differences in lower frequencies than higher frequencies.
  - we can easily tell the difference between 500 and 1000 Hz
  - but we will hardly be able to tell a difference between 10,000 and 10,500 Hz
- In 1937, Stevens, Volkmann, and Newmann proposed a unit of pitch such that equal distances in pitch sounded equally distant to the listener. This is called the mel scale.
- Perform a unit conversion to the log-mel scale
- All audio is re-sampled to 16,000 Hz, and an 80-channel log-magnitude Mel spectrogram representation is computed on 25-millisecond windows with a stride of 10 milliseconds.

Whisper Architecture

- Conv1D and GELU to transform very long spectrogram input into valid embedding.
- Classic Transformer encoder-decoder architecture with self- and cross-attentions.
- Sinusoidal positional embedding.
- Indicate the beginning of prediction with a startoftranscript token

Whisper Architecture



log-MelSepectrum

- Transform signal from the time domain to frequency domain
- Studies have shown that humans do not perceive frequencies on a linear scale. We are better at detecting differences in lower frequencies than higher frequencies.
  - we can easily tell the difference between 500 and 1000 Hz
  - but we will hardly be able to tell a difference between 10,000 and 10,500 Hz
- In 1937, Stevens, Volkmann, and Newmann proposed a unit of pitch such that equal distances in pitch sounded equally distant to the listener. This is called the mel scale.
- Perform a unit conversion to the log-mel scale
- All audio is re-sampled to 16,000 Hz, and an 80-channel log-magnitude Mel spectrogram representation is computed on 25-millisecond windows with a stride of 10 milliseconds.

Summary

- Introduced three new tasks:
  - Unsupervised learning (contrastive learning)
  - Image segmentation
  - Speech Recognition
- A few model architectures (CLIP, LLaMA2, SAM, Whisper)
- Reuse classic and powerful building blocks (eg. self-attention, image encoder)