Data 188: Introduction to Deep Learning

Multimodal models

Speaker: Eric Kim
Lecture 23 (Week 15)
2026-04-28, Spring 2026. UC Berkeley.

Announcements

- HW4 released: "Transformers for NLP (machine-translation)"
  - Groups of 4!
    - Ed post: "(HW4) Group finder thread"
  - Start early!

- HW5 released: "Visual Transformer, Masked Autoencoder"
  - Groups of 4
  - Start early!

Today's lecture

Multimodal models

Text-image alignment-based fusion: CLIP

Vision Language Models (VLM): LLaVA

Audio modeling: WaveNet, VALL-E

Modalities

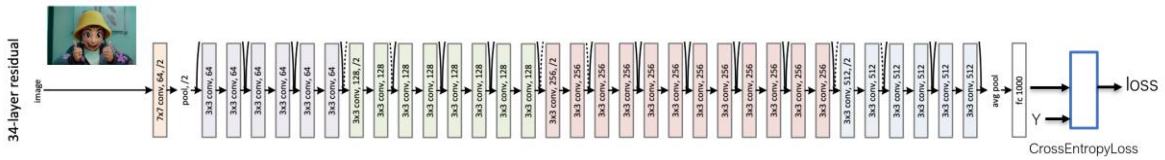
So far, we have focused on models that work with a **single input modality**.

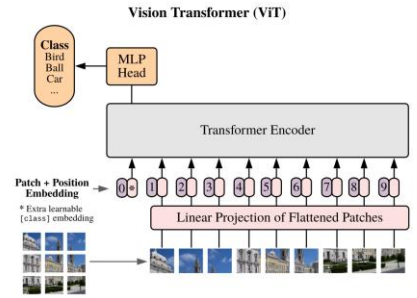
Convnets, ViT: image input modality

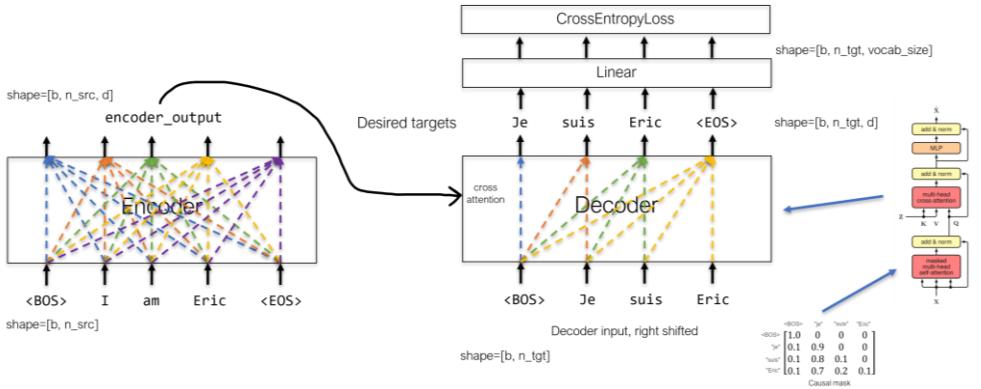
Transformers (NLP): text input modality

There are other modalities:

- Audio
- Videos (image+sound+time!)
- Other imaging data (eg fMRI data)
- ...





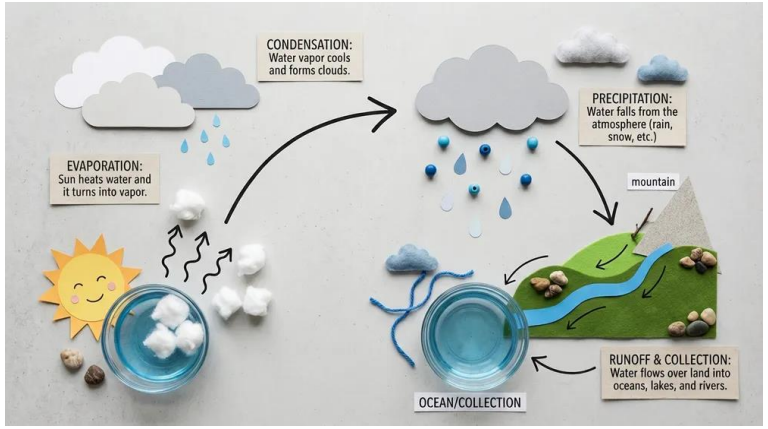


Multimodal modeling

Multimodal models operate on multiple input modalities.

As of 2026, text+image multimodal models are particularly popular.

Ex: image generation from text prompt (Stable Diffusion, Dalle, Nano Banana, Image-GPT)

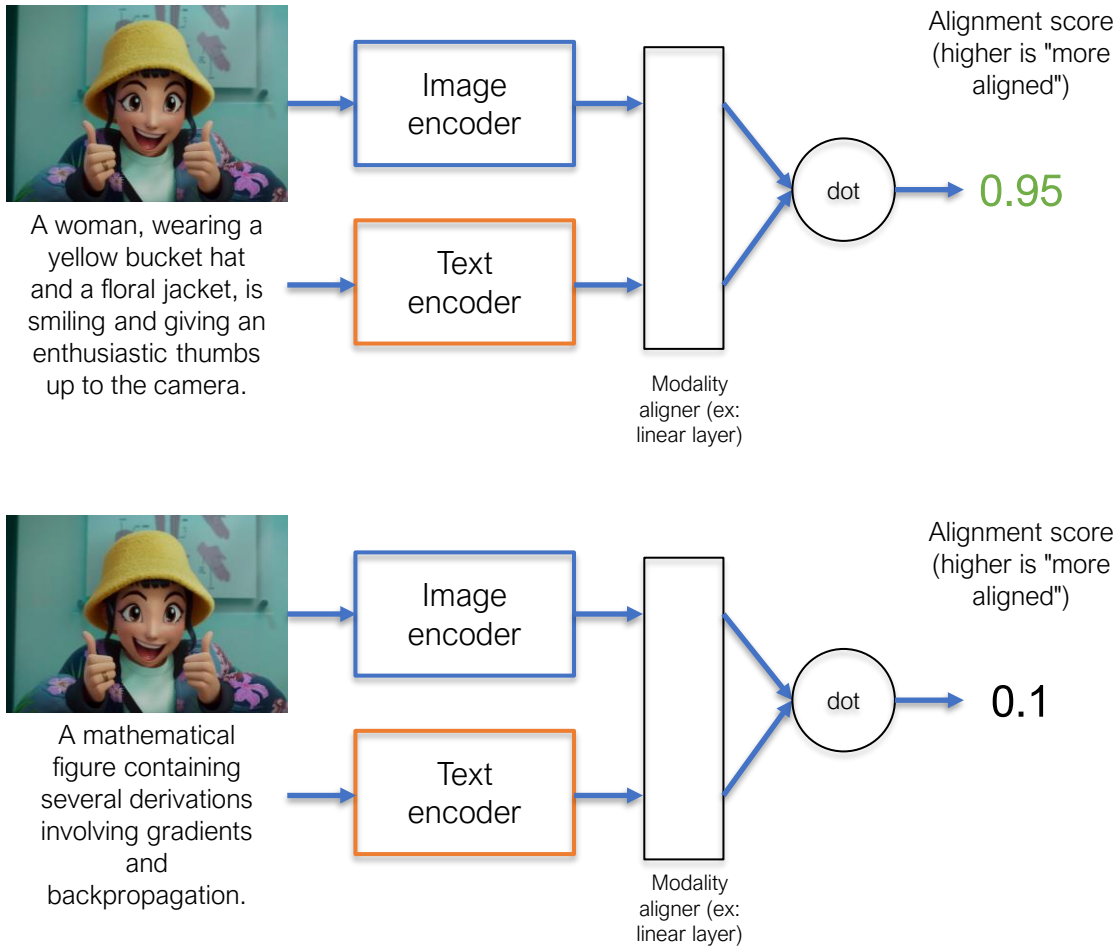


Prompt: High-quality flat lay photography creating a DIY infographic that simply explains how the water cycle works, arranged on a clean, light gray textured background. The visual story flows from left to right in clear steps. Simple, clean black arrows are hand-drawn onto the background to guide the viewer's eye. The overall mood is educational, modern, and easy to understand. The image is shot from a top-down, bird's-eye view with soft, even lighting that minimizes shadows and keeps the focus on the process.
From: Nano Banana 2 release press article (Feb 26, 2026)

Image-text alignment

Suppose we have a dataset of (image, text) pairs.

Idea: take a standard pretrained image encoder and text encoder (ex: ViT, BERT), and learn a transformation that "aligns" the image and text embeddings into a common embedding space.

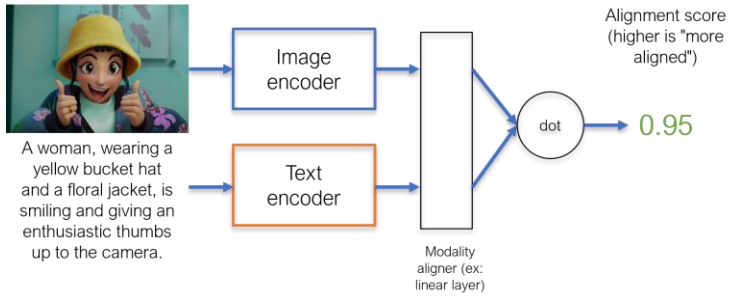


CLIP (2021)

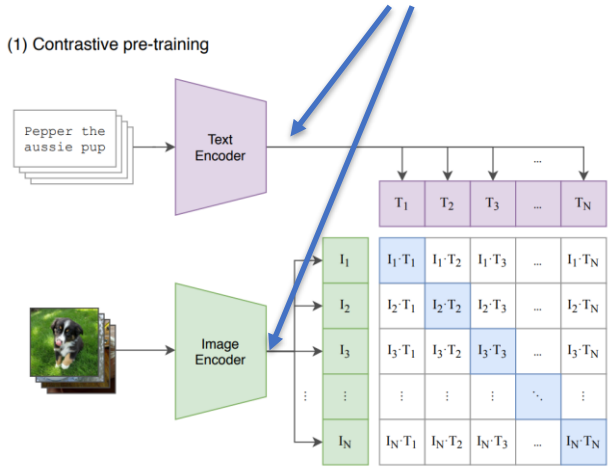
Contrastive Language-Image Pretraining ("CLIP").

Alignment score: dot product between CLIP-image and CLIP-text embedding.

Training dataset: Proprietary dataset of 400M image-text pairs crawled from the internet ("WebImageText").



Not shown: linear layer after the text and image encoder to project each modality to a shared embedding space




```
# image_encoder - ResNet or Vision Transformer
# text_encoder  - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t             - learned temperature parameter

# 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
```

Figure 3. Numpy-like pseudocode for the core of an implementation of CLIP.

During training: given batch of (image, text) pairs, calculate the pairwise dot product similarity matrix between CLIP-image and CLIP-text embeddings.
Positive pairs: along the main diagonal.
Negative pairs: all other pairs ("in-batch negatives").

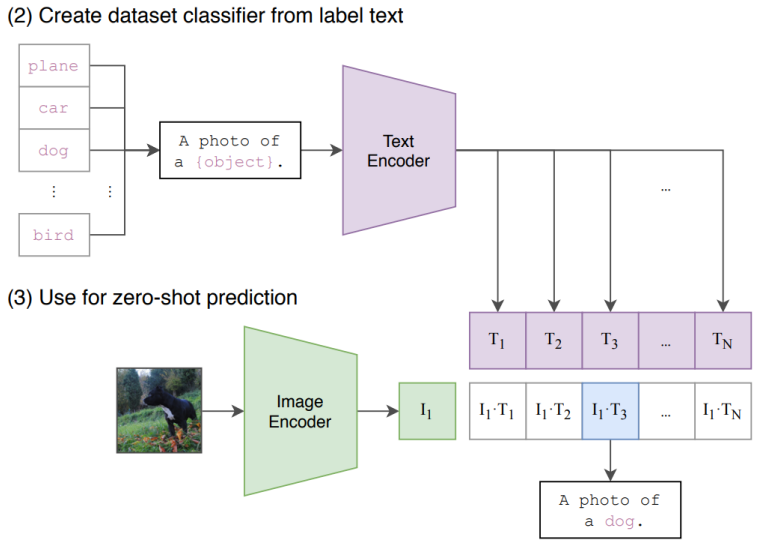
Important: must have a suitably large batchsize so that you have diverse enough in-batch negatives! Ex: CLIP paper used batchsize=32768.

CLIP: Zero-shot classification

Zero-shot image classification setup: given a query image, and a category taxonomy (ex: ImageNet-1k):

1. Calculate CLIP-image embedding.
2. Calculate CLIP-text embedding for all N categories (text="A photo of {category}")
3. Compute logits via dot-product between CLIP-image and all N CLIP-text embeddings.

No fine-tuning required ("zero shot")



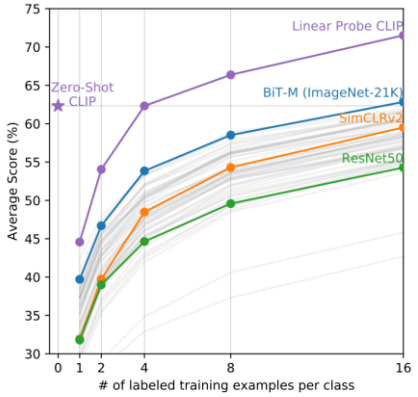


Figure 6. **Zero-shot CLIP outperforms few-shot linear probes.** Zero-shot CLIP matches the average performance of a 4-shot linear classifier trained on the same feature space and nearly matches the best results of a 16-shot linear classifier across publicly available models. For both BiT-M and SimCLRv2, the best performing model is highlighted. Light gray lines are other models in the eval suite. The 20 datasets with at least 16 examples per class were used in this analysis.

CLIP: cross-modal retrieval

Since CLIP-image and CLIP-text embeddings are in a shared embedding space, we can do **cross-modal** retrieval:

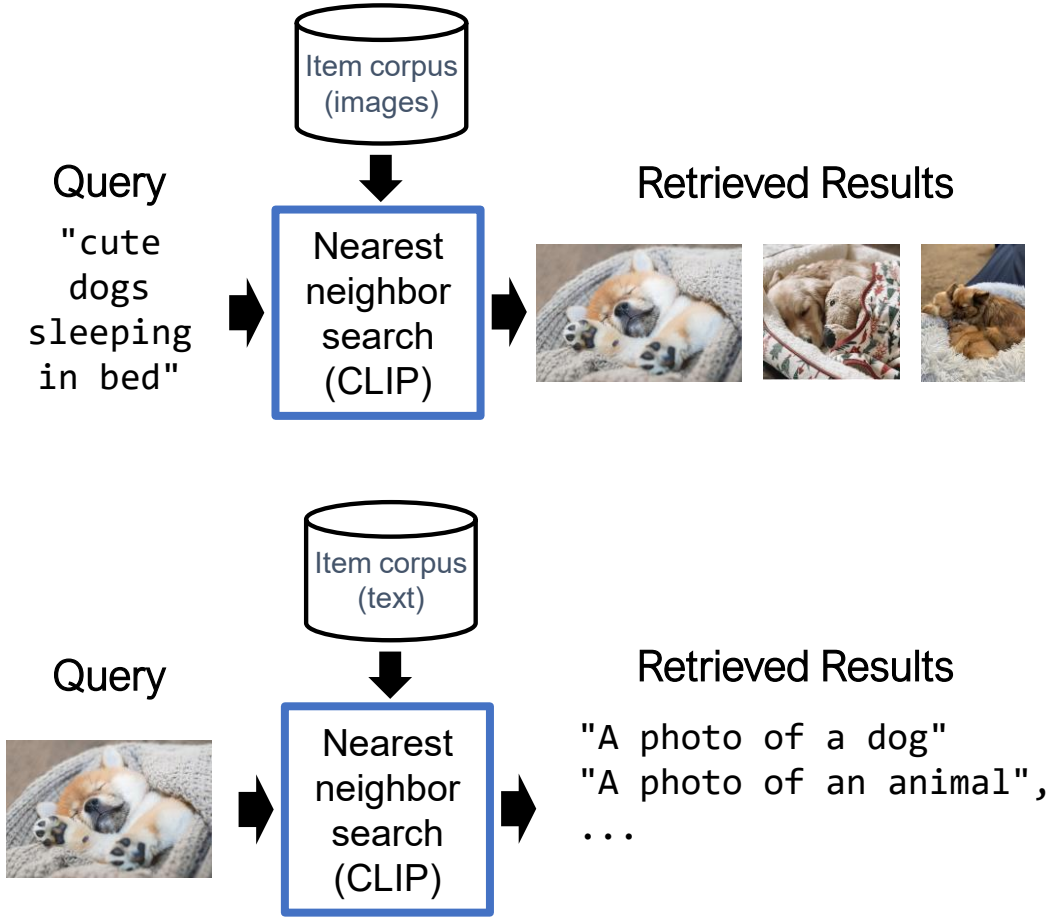
- Text to Image retrieval

- Image to Text retrieval

As well as unimodal search:

- Text to text

- Image to Image



https://living.greatpetcare.com/entertainment/sleeping-dog-images/
@_olives_odyssey_
@asistercalledwinston

LAION-400M (2021)

LAION-400M: dataset of 400M (image, text) pairs.

Image-text pairs crawled from Common Crawl, with various filtering.

CLIP filtering: use a CLIP model to filter out poor quality examples: image-text pairs with poor alignment score.

Great for tasks like: text-to-image generation, image captioning.

Open-source: while the original CLIP model was trained on a proprietary dataset, OpenCLIP is trained on open-source datasets such as LAION.

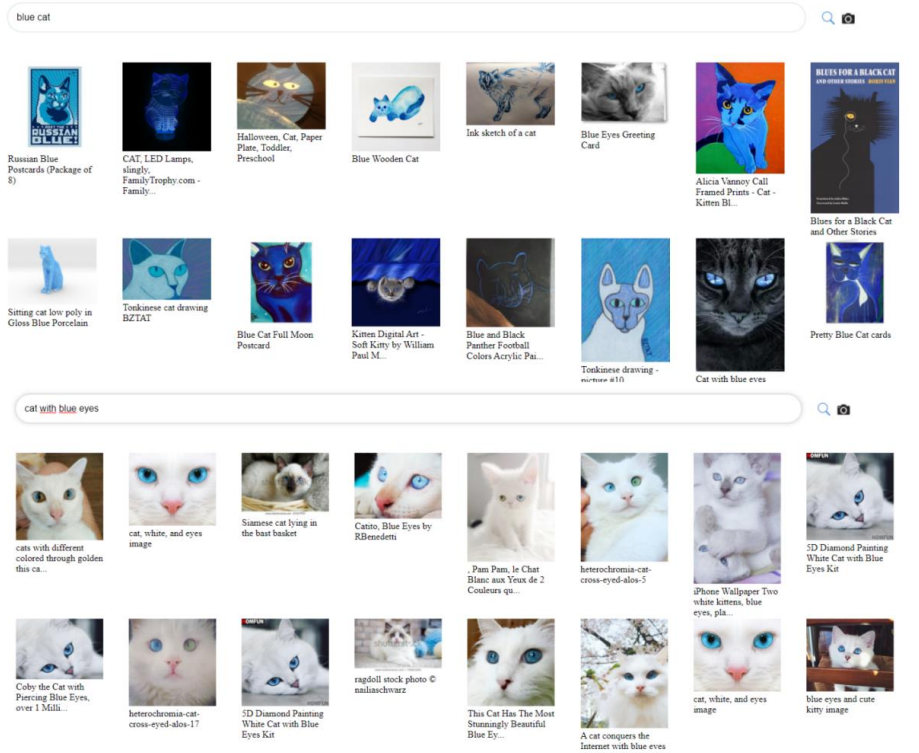


Figure 1: Sample images retrieved from the queries "blue cat" or "cat with blue eyes" in the web demo

Re-LAION-5B (2024)

Re-LAION-5B (2024) (orig paper): 5B (image, text) pairs.

Improved scale, improved filtering.

Improved Multilingual support (LAION-400M is primarily English only)

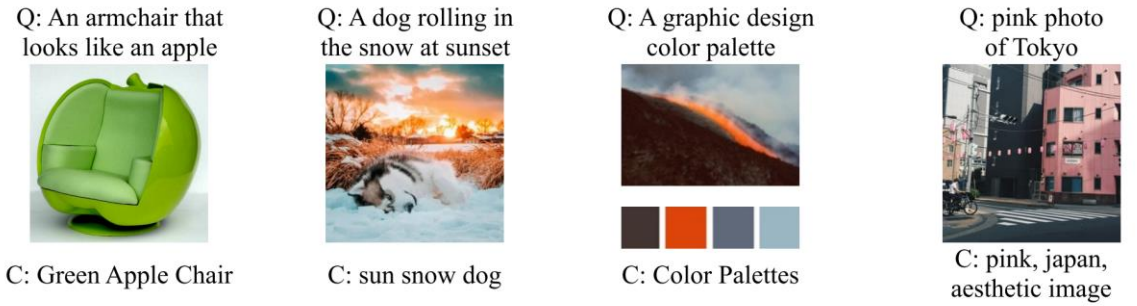


Figure 3: **LAION-5B examples.** Sample images from a nearest neighbor search in LAION-5B using CLIP embeddings. The image and caption (C) are the first results for the query (Q).

Dataset	# English Img-Txt Pairs
Public Datasets	
MS-COCO	330K
CC3M	3M
Visual Genome	5.4M
WIT	5.5M
CC12M	12M
RedCaps	12M
YFCC100M	100M[2]
LAION-5B (Ours)	**2.3B**
Private Datasets	
CLIP WIT (OpenAI)	400M
ALIGN	1.8B
BASIC	6.6B

Table 1: **Dataset Size.** LAION-5B is more than 20 times larger than other public English image-text datasets. We extend the analysis from Desai et al. [14] and compare the sizes of public and private image-text datasets.

LLaVA (2023)

LLaVA (2023): Vision Language Model (VLM).

Approach: take an existing pretrained LLM model that only operates on text (ex: Vicuna).

Modify it to accept images by adding image tokens (ie from a pretrained image encoder like ViT) to the input sequence.

Training methodology: use language-only GPT-4 to generate vision-language instruction-following data.

Autoregressive transformer decoder, pretrained on text tasks (ex: Vicuna)

Projection (ie linear layer) maps vision tokens to text token space

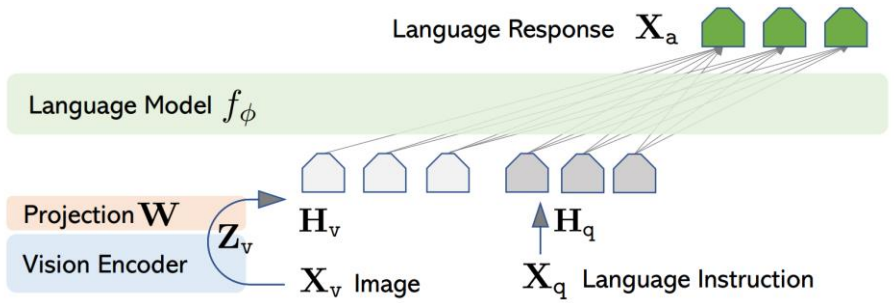
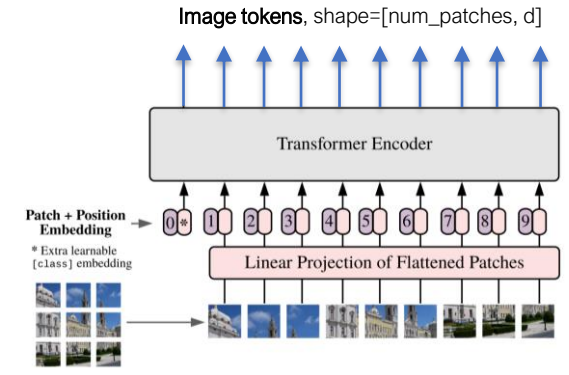


Figure 1: LLaVA network architecture.

Rather than pass the "summary" image embedding to LLaVA (ex: a single 1024d CLS embedding), we pass all encoder output token embeddings to LLaVA. Ex: for 197 tokens, shape=[197, 1024].

Intuition: Image tokens contain much more detailed semantics than the single CLS token aggregate embedding.



LLaVA: instruction-following dataset

Goal: want to create a high-quality text-image conversational dataset.

Issue: collecting this dataset via human labelers is extremely challenging.

Idea: use extremely powerful VLM's like ChatGPT/GPT-4 to help create this dataset!

"Teacher-student distillation": utilize ChatGPT's impressive world knowledge (and impressive scale!) to create a strong dataset to train other (likely smaller!) models on.

Context type 1: Captions
A group of people standing outside of a black vehicle with various luggage.
Luggage surrounds a vehicle in an underground parking area
People try to fit all of their luggage in an SUV.
The sport utility vehicle is parked in the public garage, being packed for a trip
Some people with luggage near a van that is transporting it.

Context type 2: Boxes
person: [0.681, 0.242, 0.774, 0.694], backpack: [0.384, 0.696, 0.485, 0.914], suitcase: ...<omitted>

Response type 1: conversation
Question: What type of vehicle is featured in the image?
Answer: The image features a black sport utility vehicle (SUV) ...<omitted>
Response type 2: detailed description
The image is an underground parking area with a black sport utility vehicle (SUV) parked. There are three people in the scene, with one person standing closer to the left side of the vehicle, another person in the middle, and the third person on the right side. They are all working together to pack their luggage into the SUV for a trip. ...<omitted>
Response type 3: complex reasoning
Question: What challenges do these people face?
Answer: In the image, a group of people is standing outside a black SUV in a parking area, surrounded by various pieces of luggage, including suitcases and backpacks. They are facing the challenge of fitting all their luggage into the black SUV. There are multiple suitcases and backpacks to be packed, which suggests that the group has a significant amount of belongings ...<omitted>

Table 1: One example to illustrate the instruction-following data. The top block shows the contexts such as captions and boxes used to prompt GPT, and the bottom block shows the three types of responses. Note that the visual image is not used to prompt GPT, we only show it here as a reference.

LLaVA: example 1

Visual input example, Chicken Nugget Map:	
	
User	Can you explain this meme in detail?
LLaVA	The meme in the image is a creative and humorous take on food, with a focus on chicken nuggets, as the center of the universe. The meme begins with the phrase "Sometimes I just look at pictures of the Earth from space and I marvel at how beautiful it all is..." The main image features a tray of cooked chicken nuggets, with the chicken pieces positioned to resemble the continents and islands on a world map. The meme playfully suggests that the chicken nuggets represent the Earth, and the various locations depicted in the photo are actually chicken nugget versions of different places. The punchline of the meme appears in smaller text at the bottom, reading "I mean, it's not the real Earth, but how beautiful it is all is." This lighthearted and entertaining meme combines food and humor, using the image of chicken nuggets to create a funny and imaginative representation of the world.
User	Can you explain this meme?
GPT-4 [36]	This meme is a joke that combines two unrelated things: pictures of the earth from space and chicken nuggets. The text of the meme suggests that the image below is a beautiful picture of the earth from space. However, the image is actually of chicken nuggets arranged to vaguely resemble a map of the world. The humor in this meme comes from the unexpected juxtaposition of the text and the image. The text sets up an expectation of a majestic image of the earth, but the image is actually something mundane and silly.
User	Can you explain this meme in detail?
BLIP-2	sometimes i just look at pictures of the earth from space and marvel how beautiful it is
User	Can you explain this meme in detail?
OpenFlamingo	It's a picture of a chicken nugget on the International Space Station.

Table 9: Example prompt comparing LLaVA, GPT-4, BLIP-2, and OpenFlamingo's visual reasoning capabilities in understanding the humor. BLIP-2 and OpenFlamingo fail to follow the user's instructions. LLaVA and GPT-4 both explain the meme and its humor, while GPT-4 produces a more concise answer.

LLaVA: example 2

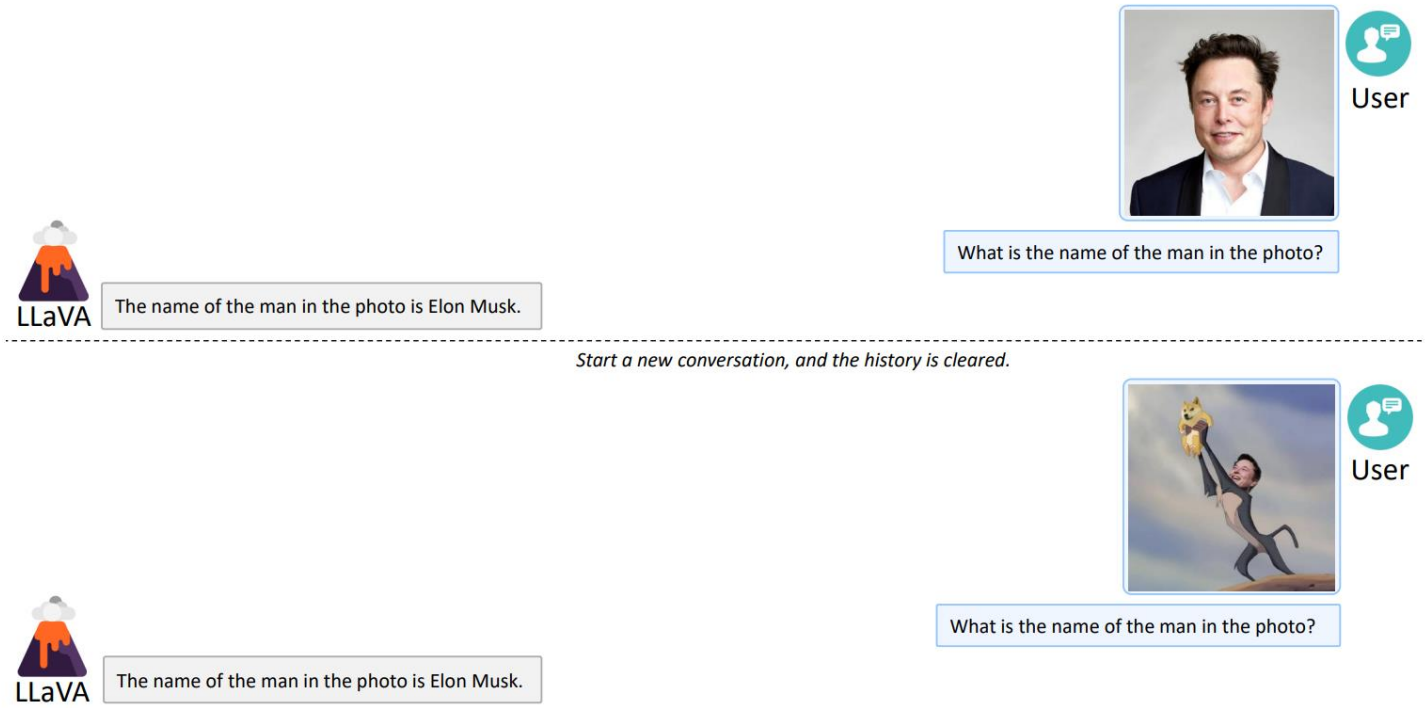


Figure 6: An interesting emergent behavior of LLaVA is its ability to recognize Elon Musk both in a headshot and in a humorous meme where he is dressed as a doge. This implies that the pre-trained CLIP vision encoder may have seen images of Elon Musk. However, it is still surprising because Elon Musk *never* appears in the training data for either the visual feature alignment or visual instruction tuning stages of LLaVA, which indicates that the base language model generalizes to unseen visual concepts.

Audio representations: waveform

Digital audio raw format is typically stored as a **waveform** (sound wave amplitude), sampled at a high frequency.

Ex: 44.1 kHz (44100 samples per second) is standard for music streaming.

How to pass audio into a model?

One (computationally infeasible) way: pass the waveform as a (very long!) sequence.

Ex: for a 2 minute song sampled at 44.1 kHz (standard for streaming), would lead to sequence length of 5.292M (!)

Figure 1: A second of generated speech.

WaveNet (2016)

WaveNet (2016, DEMO).

Tasks: text-to-speech generation, audio generation.

Operates on waveform, but with a model architecture that only looks at a medium-sized chunk of audio at a time (dilated causal 1D convolutions).

Chunk size is ~300ms (1024 samples), aka "receptive field".

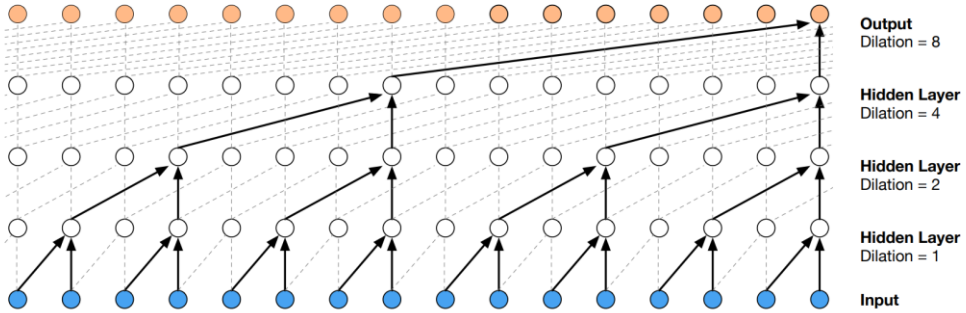


Figure 3: Visualization of a stack of *dilated* causal convolutional layers.

Making Music

Since WaveNets can be used to model any audio signal, we thought it would also be fun to try to generate music. Unlike the TTS experiments, we didn't condition the networks on an input sequence telling it what to play (such as a musical score); instead, we simply let it generate whatever it wanted to. When we trained it on a dataset of classical piano music, it produced fascinating samples like the ones below:

0:10 / 0:10

0:10 / 0:10

If you're curious, a more computationally-efficient followup is: Tacotron 2 (2017) (pytorch)

Audio representation: Mel spectrogram

Idea: rather than analyze audio at the waveform level (computationally challenging!), perhaps looking at its frequencies is a better idea.

Mel spectrogram (MFCC, torchaudio.transforms.MelSpectrogram): frequency representation of audio, where frequency bins are tailored to human audio perception.

Models can treat this spectrogram as an image.

Ex: Conv2d! Or tokenize it similar to ViT.

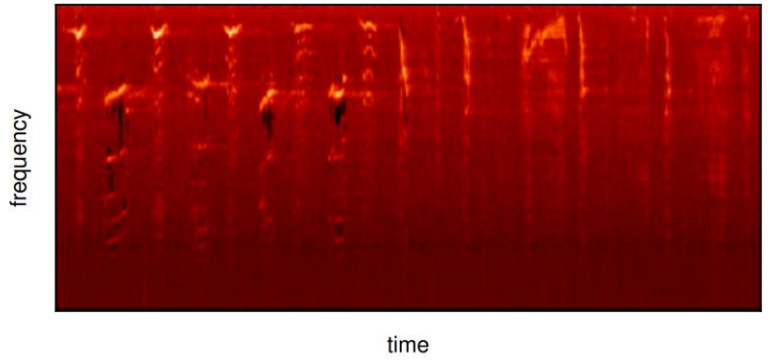


Figure 12.25 An example mel spectrogram of a humpback whale song. [Source data copyright ©2013–2023, librosa development team.]

Note: while it's possible to convert from spectrogram back to audio waveform, it's a lossy transformation (waveform -> spectrogram transformation discards phase information)

Text to speech: VALL-E (2023)

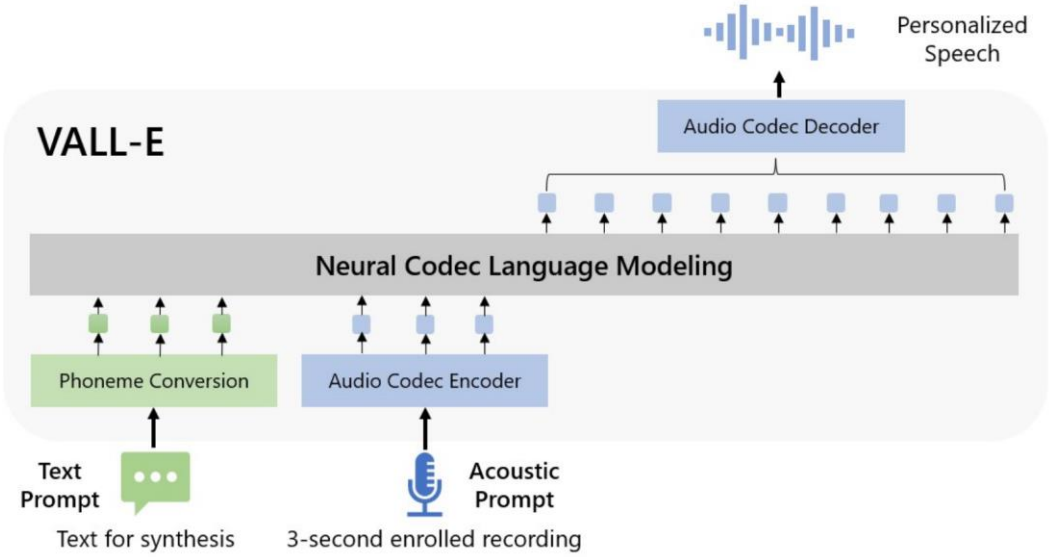
VALL-E (2023, DEMO).

Task: text to speech, with speaker personalization.

Speaker personalization: given a short clip of a target person speaking, use that as context to condition the model to generate speech in the target person's voice.

Model Architecture: transformer autoregressive decoder!

Takeaway: to add speaker personalization, we add the speaker's "enrolled recording" as additional input to the model (aka additional context).



VALL-E: codebook ids

They represent audio as discrete codebook ID's (see: EnCodec), and the autoregressive decoder accepts/predicts codebook ids.

Think of this codebook-approach as a way to tokenize audio inputs.

Notably: given predicted codebook IDs, we can produce the raw audio.

To learn more, look up: residual vector quantization ("RVQ").

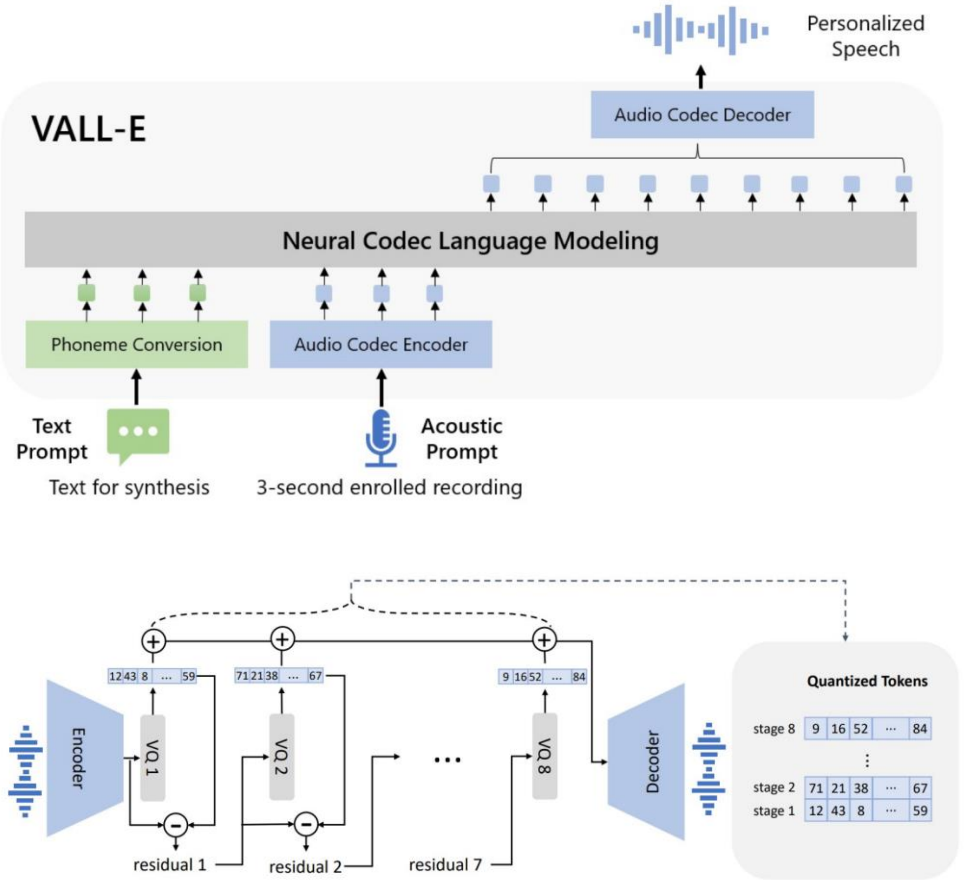


Figure 2: The neural audio codec model revisit. Because RVQ is employed, the first quantizer plays the most important role in reconstruction, and the impact from others gradually decreases.

Closing thoughts

Treating input data as a sequence of tokens has proven to be an effective (and flexible) approach that uniformly handles multimodal inputs and outputs.

Training multimodal models requires large datasets that contain all desired modalities.

A new trend: utilize powerful LLM's/VLM's (ex: ChatGPT) to help curate high-quality training datasets in a cost-effective manner.