Announcements

- Submit your NSF access account information via canvas
- If you checked one out on Tuesday 09/07:
  - Return the kit by Tuesday, 09/15
  - Please bring the kits to class or email me to make alternate arrangements
- Lab 2 will be assigned on Tuesday

Carnegie Mellon University

L6: Applied Machine Learning

18-848 RadarML

Tianshu Huang

In these two lectures:

- Everything you need to know about ML
- How to develop and evaluate ML models & methodology
- ML details are left as an exercise for other CMU faculty

Crash Course: Applied Machine Learning

- Introduction to Machine Learning
  - Empirical Risk Minimization
  - Deep Learning & Transformers
- Training & Evaluation
  - The ML Life Cycle
  - Variance of Evaluation
- Learning at Scale
  - Foundation Models
  - Scaling Laws

Crash Course: Applied Machine Learning

- **Introduction to Machine Learning**
  - **Empirical Risk Minimization** + probability theory review
  - Deep Learning & Transformers
- Training & Evaluation
  - The ML Life Cycle
  - Variance of Evaluation
- Learning at Scale
  - Foundation Models
  - Scaling Laws

Review: **What is a distribution?**

1. **All possible outcomes** of a random variable, and the **probabilities** associated with them
2. **PDF** (Probability Density Function) for continuous RVs or a **PFM** (Probability Mass Function) for discrete RVs
3. **A push-forward measure** of the probability measure onto the measure space induced by a random variable

Review: **What are some distributions?**

- Gaussian (normal)
- Bernoulli (0-1)

} real-values

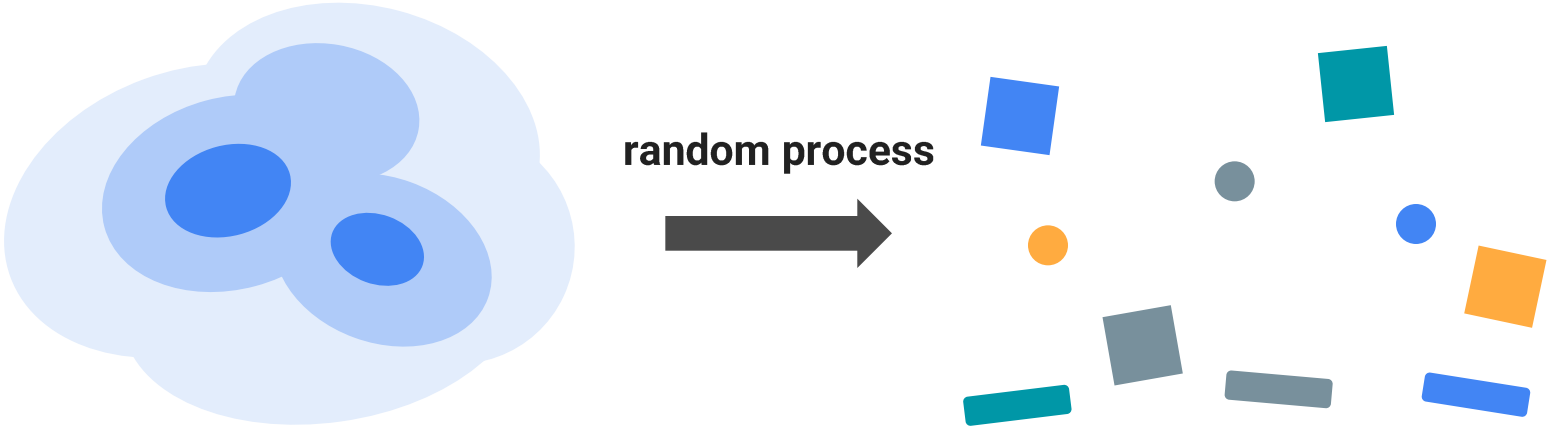
uniform

poisson

...

- all possible images from a camera
- all possible books

Review: **Data comes from distributions**



a distribution

observed data

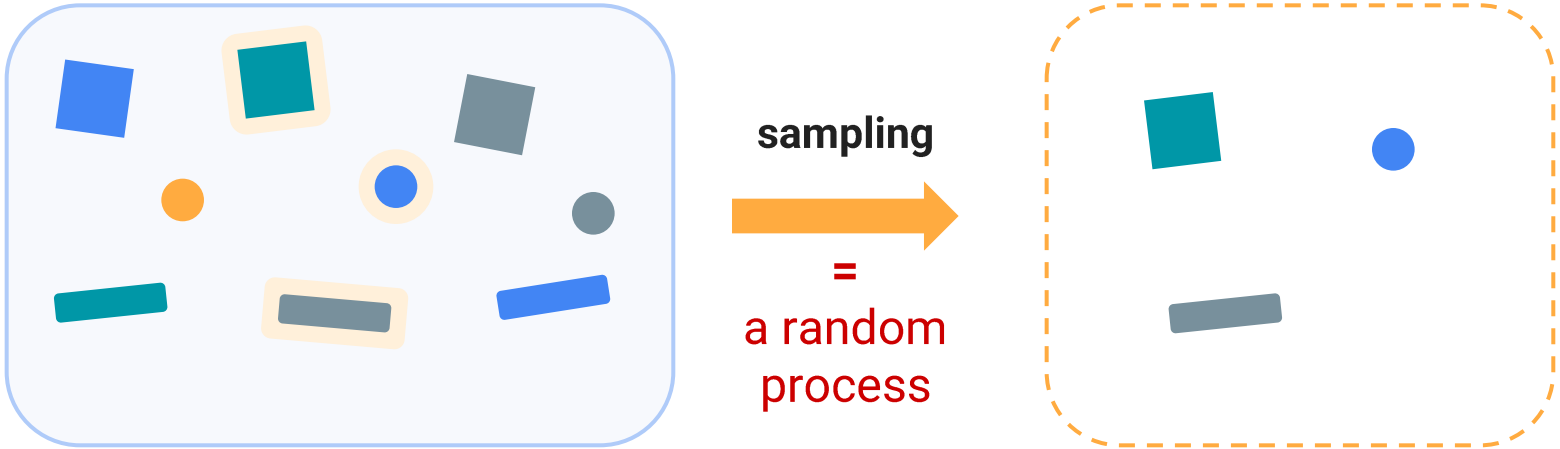
Review: **IID** (Independent and Identically Distributed)

What does it mean for data to be:

- Not independent?
  - time series
  - correlations in general
- Not identically distributed?
  - different data sources

Review: **Datasets are themselves distributions**

"empirical distribution"



a dataset

= a distribution

a sample

= observed data

Empirical

1. Computed using an “**empirical**” distribution, i.e., dataset

Empirical Risk

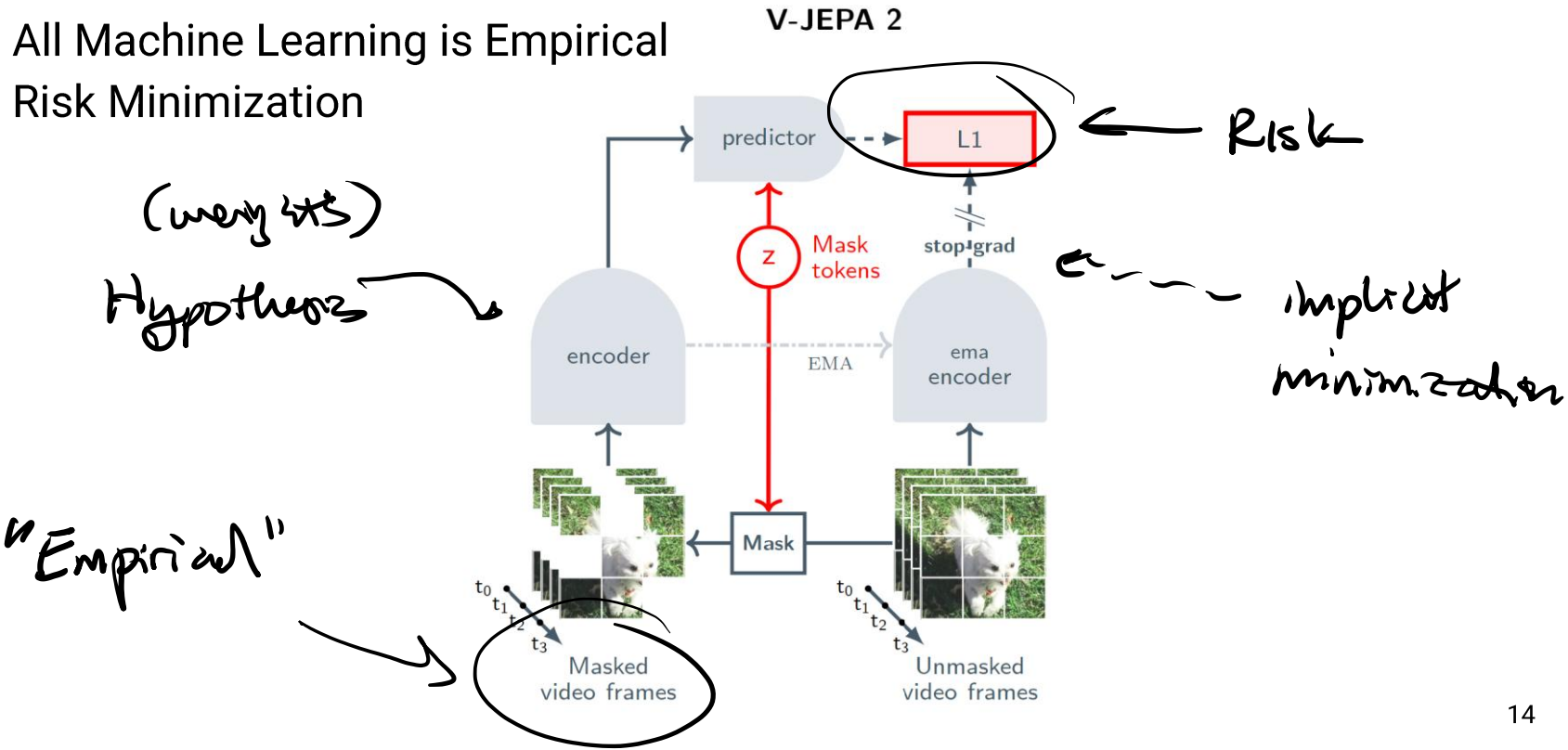
1. **The risk:** a measure of how different a hypothesis is from the true outcome (a.k.a., a loss function), when
2. Computed using an “**empirical**” distribution, i.e., dataset

Empirical Risk Minimization

1. Find the **hypothesis** (e.g., model weights) which best **minimizes**
2. **The risk:** a measure of how different a hypothesis is from the true outcome (a.k.a., a loss function), when
3. Computed using an “**empirical**” distribution, i.e., dataset

Empirical Risk Minimization

All Machine Learning is Empirical Risk Minimization



What can we “hill climb”?

Empirical

- more data, better data...

Risk

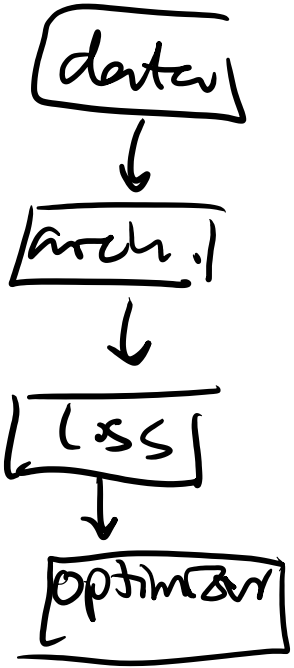
- loss better aligned with our goal

Minimization

- optimizer, more complete...

(Hypothesis)

- loss better for optimization
- arch., params, etc.



Crash Course: Applied Machine Learning

- **Introduction to Machine Learning**
  - Empirical Risk Minimization
  - **Deep Learning & Transformers**
- Training & Evaluation
  - The ML Life Cycle
  - Variance of Evaluation
- Learning at Scale
  - Foundation Models
  - Scaling Laws

A History of Deep Learning

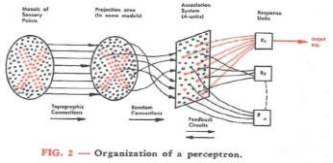
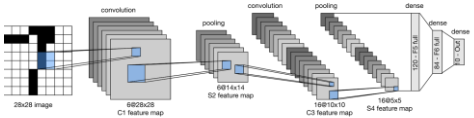
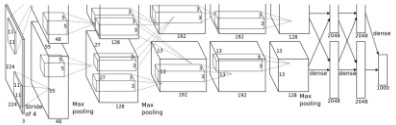


FIG. 2 — Organization of a perceptron.

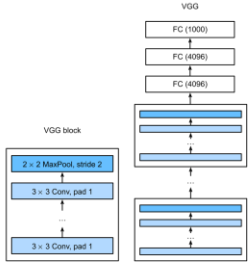
1957 Perceptrons



1998 LeNet



2012 AlexNet



2014 VGG

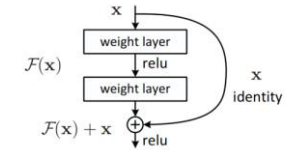
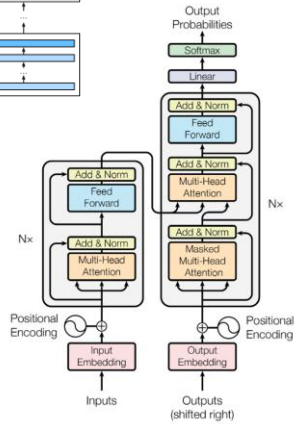


Figure 2. Residual learning: a building block.

2015 ResNet



2017 Transformers

Early Perceptrons (1957)

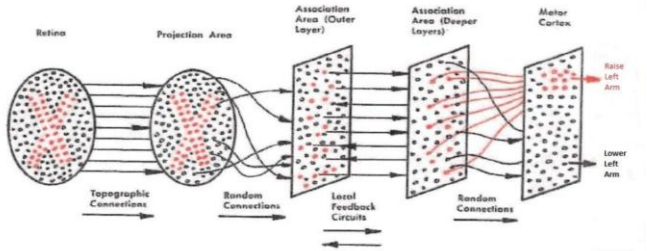


FIG. 1 — Organization of a biological brain. (Red areas indicate active cells, responding to the letter X.)

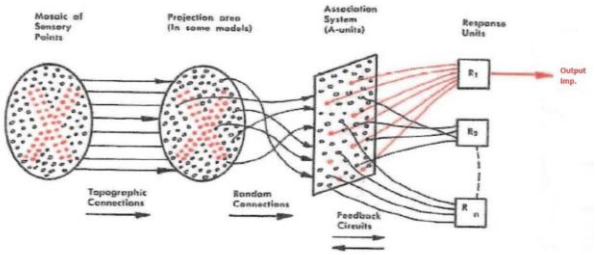
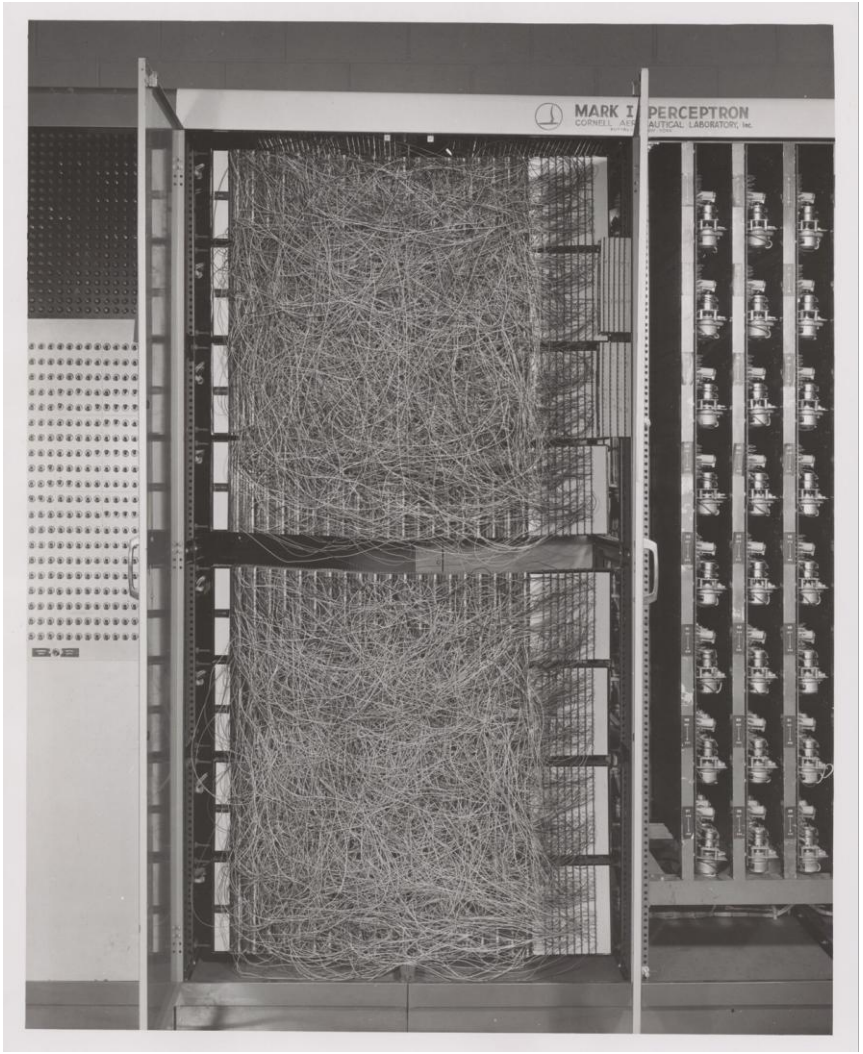
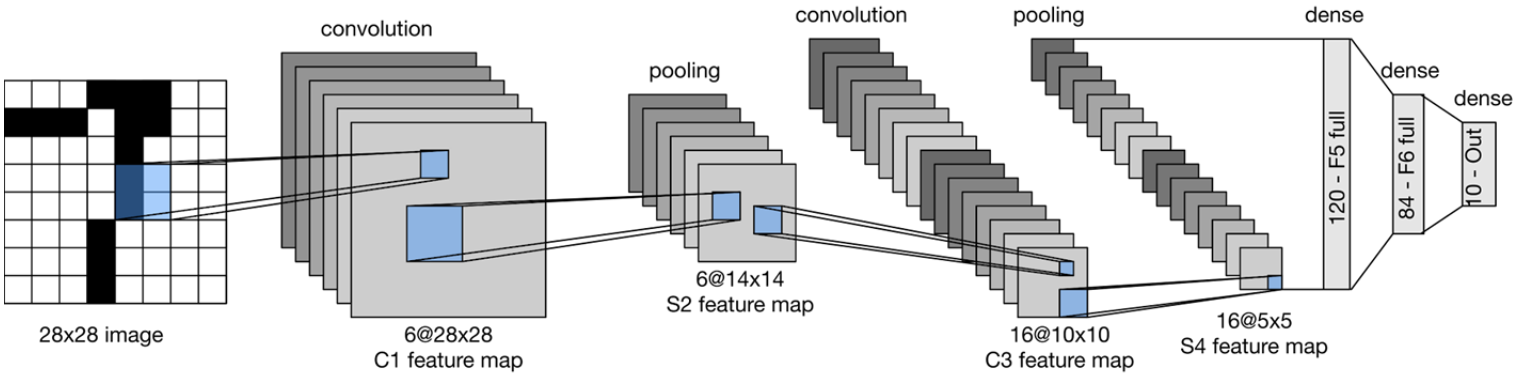


FIG. 2 — Organization of a perceptron.



LeNet (1998)


convolution
pooling
convolution
pooling
dense
dense
dense
120 - F5 full
84 - F6 full
10 - Out
28x28 image
6@28x28
C1 feature map
6@14x14
S2 feature map
16@10x10
C3 feature map
16@5x5
S4 feature map

AlexNet (2012)

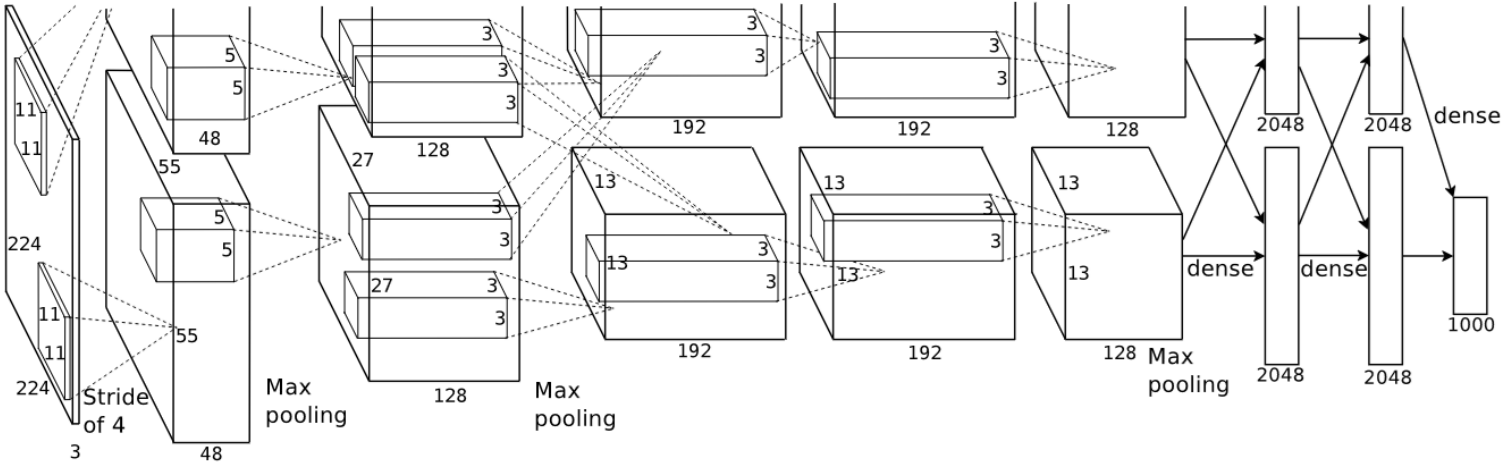
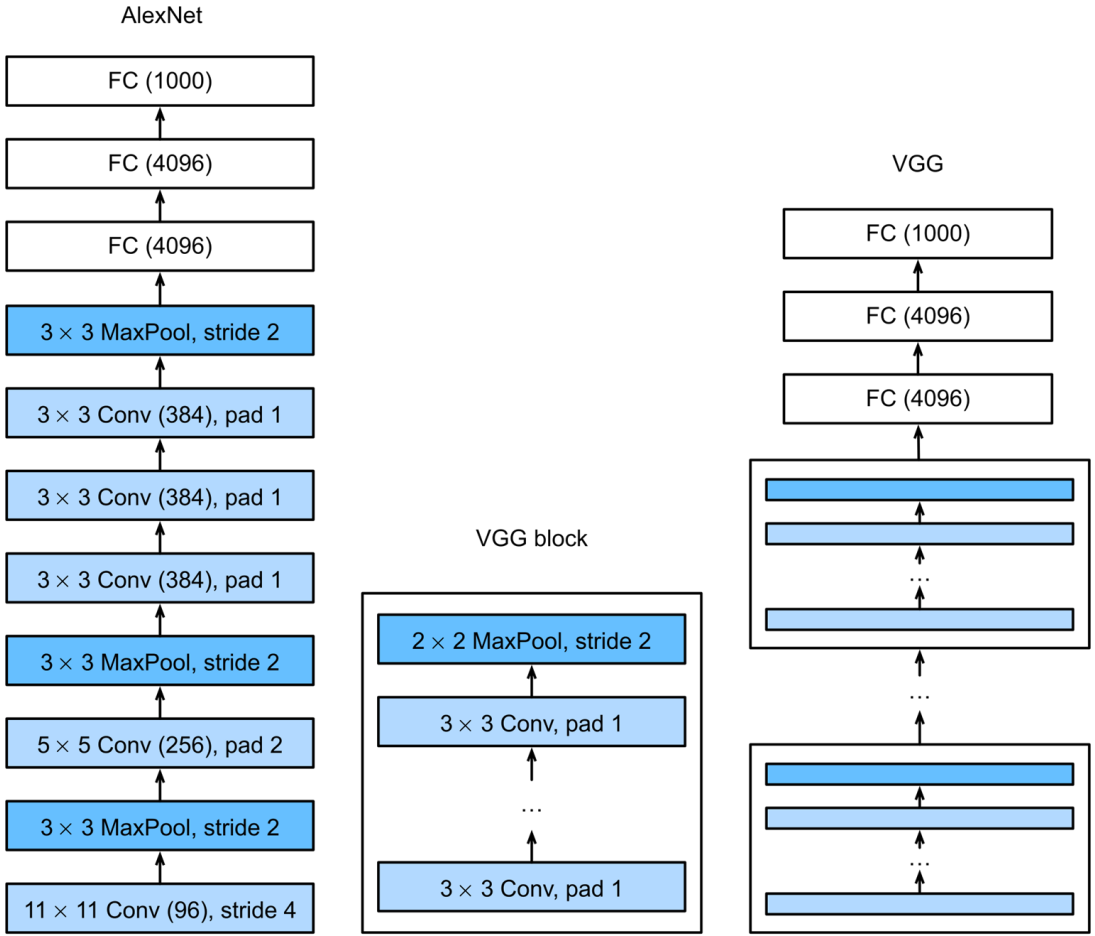


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

VGG (2014)


AlexNet
FC (1000)
FC (4096)
FC (4096)
3 × 3 MaxPool, stride 2
3 × 3 Conv (384), pad 1
3 × 3 Conv (384), pad 1
3 × 3 Conv (384), pad 1
3 × 3 MaxPool, stride 2
5 × 5 Conv (256), pad 2
3 × 3 MaxPool, stride 2
11 × 11 Conv (96), stride 4
VGG block
2 × 2 MaxPool, stride 2
3 × 3 Conv, pad 1
...
3 × 3 Conv, pad 1
VGG
FC (1000)
FC (4096)
FC (4096)
...
...
...

ResNet (2015)

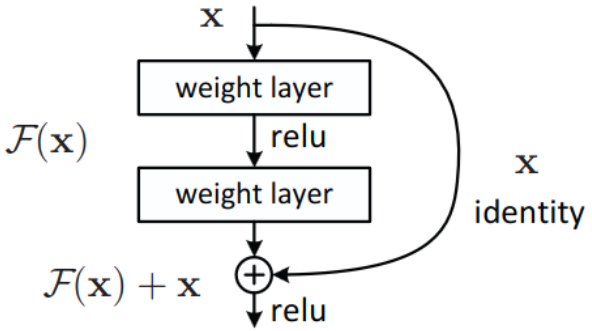


Figure 2. Residual learning: a building block.

Transformers (2017–)

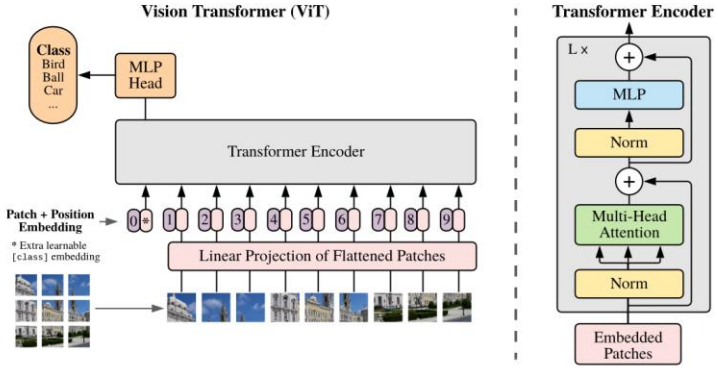


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification token" to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

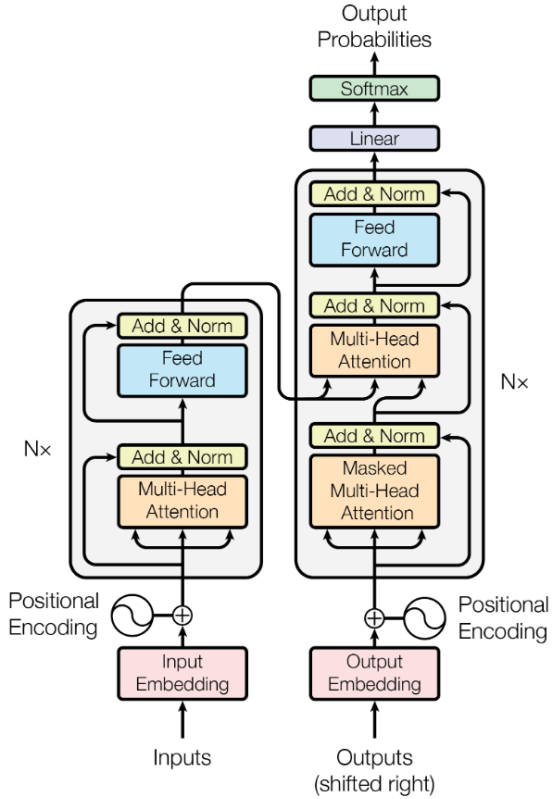
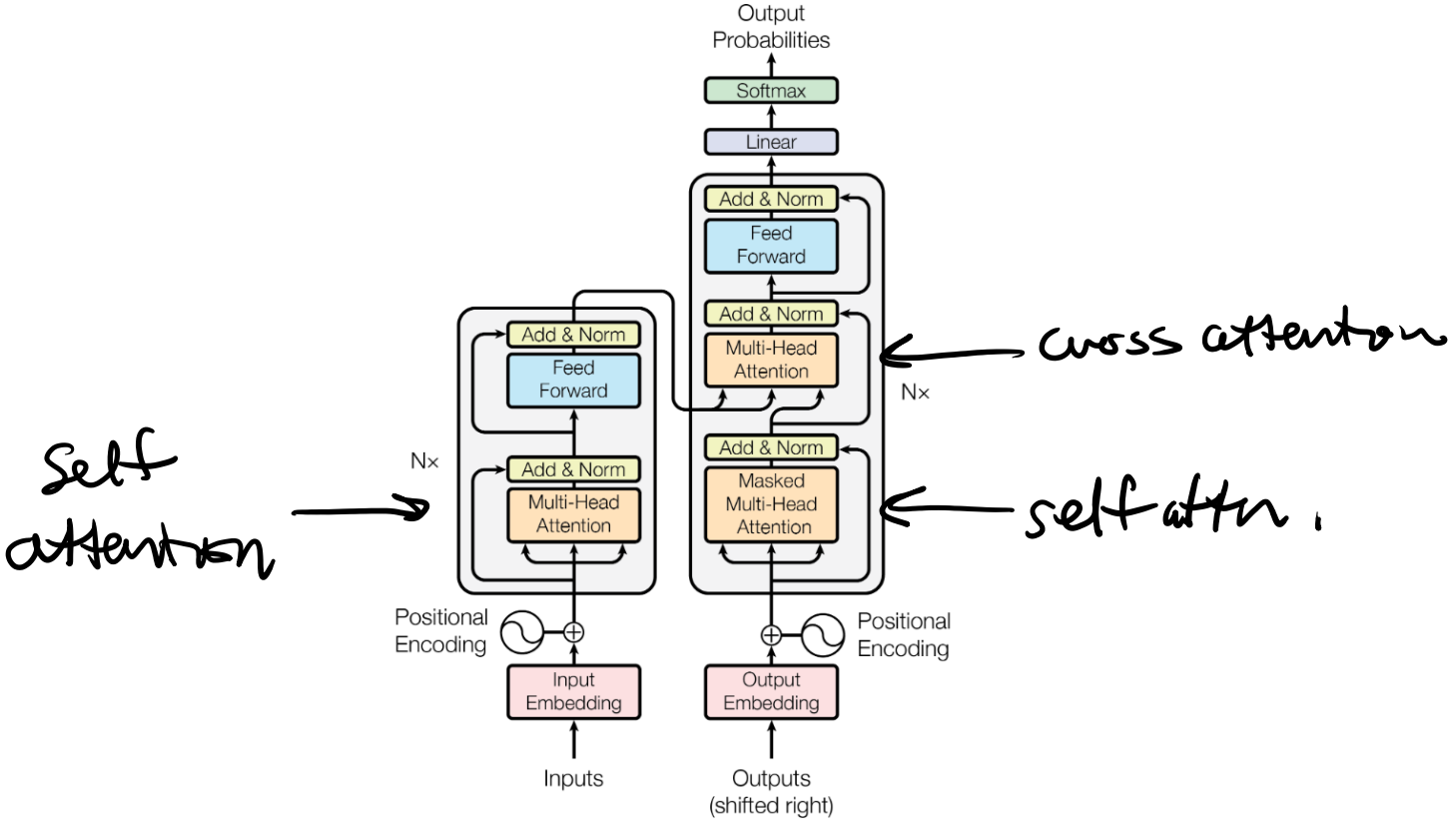


Figure 1: The Transformer - model architecture.

Transformers: **Cross vs Self Attention**



Transformers: **Feedforward vs Autoregressive**

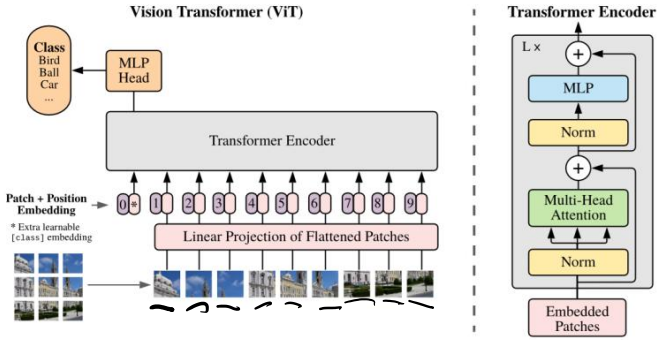
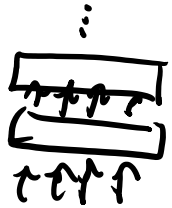
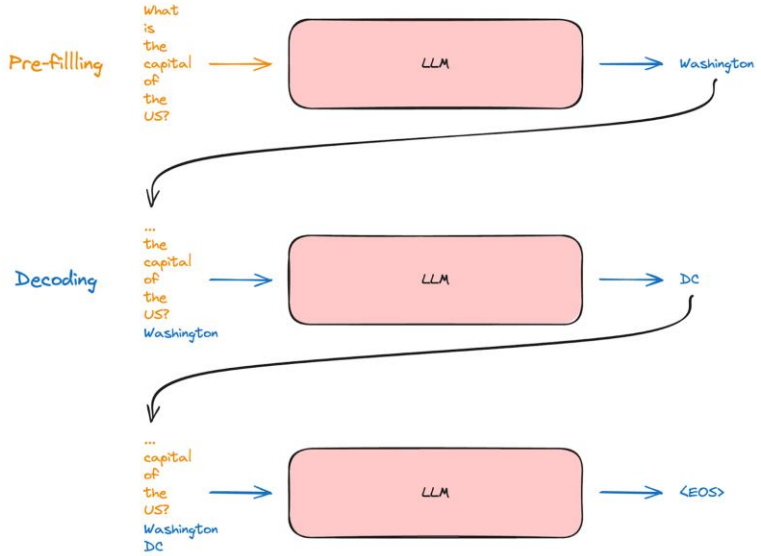


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification token" to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).





Transformers: **Encoder vs Decoder**

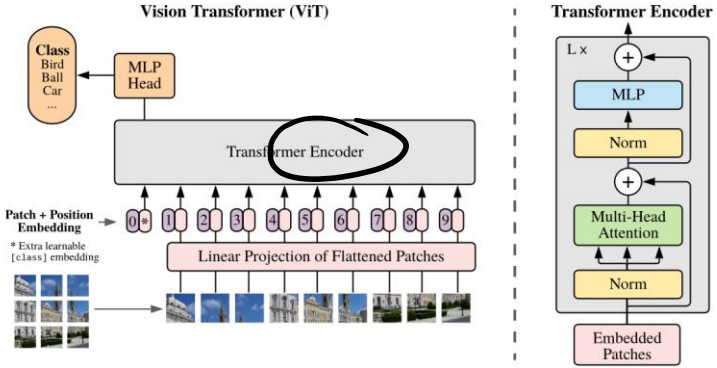


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification token" to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

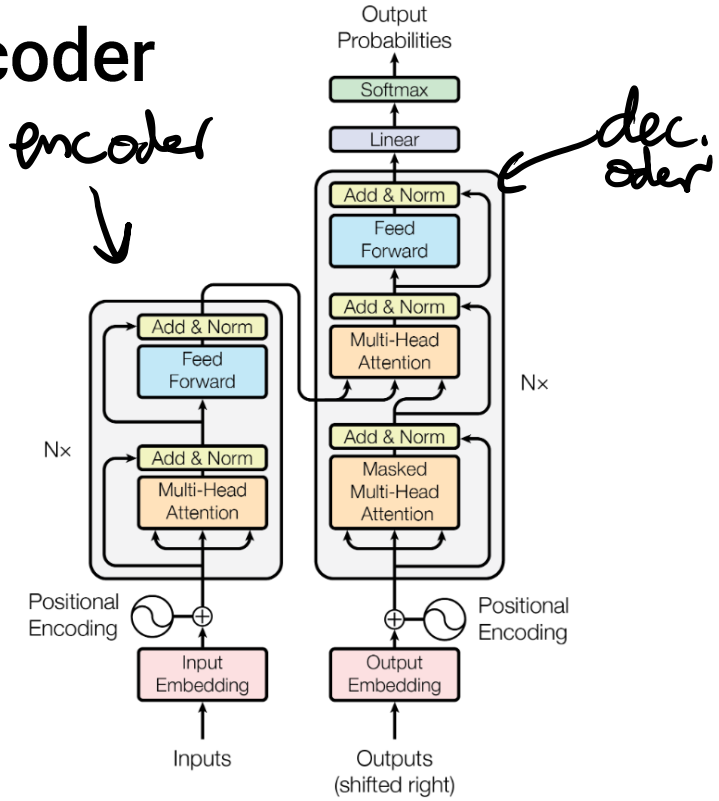


Figure 1: The Transformer - model architecture.

Transformers: **Encoder vs Decoder**

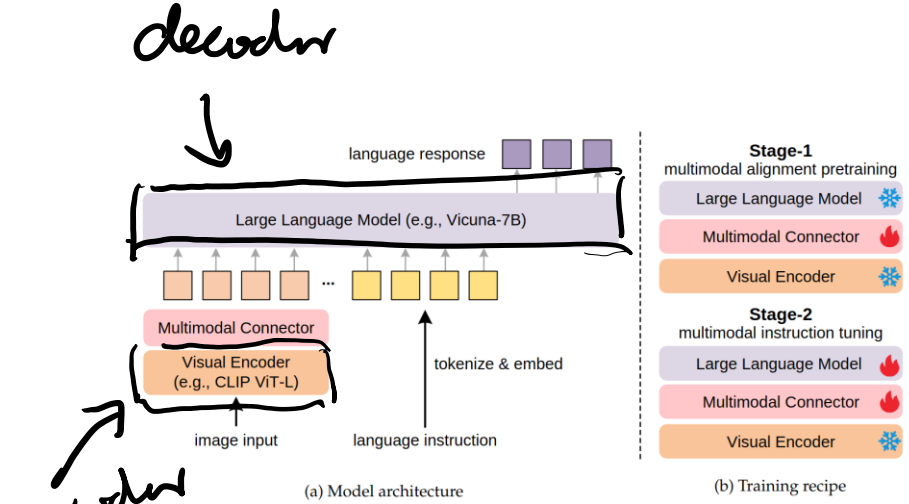
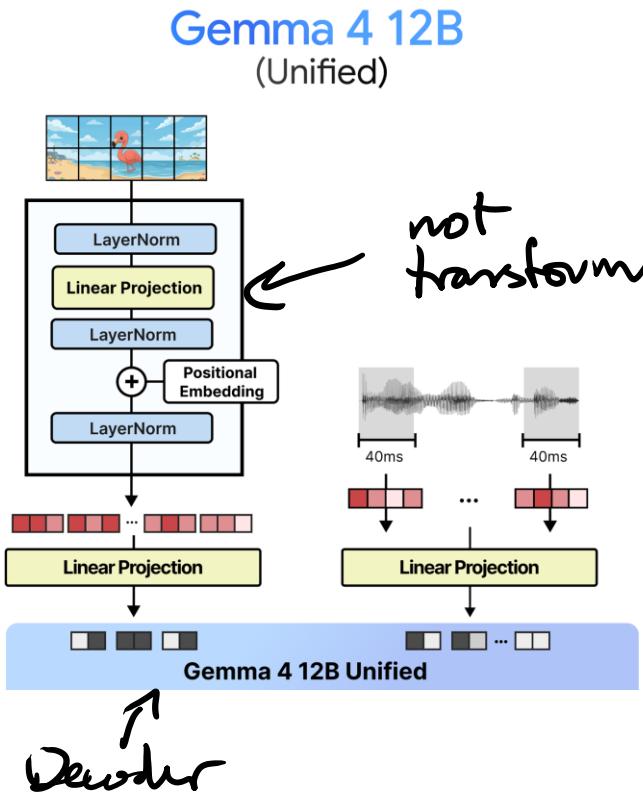


Figure 1 | LLaVA-1.5's model architecture and its two-stage training recipe.



Transformers: **Encoder vs Decoder**

1. Does the transformer operate primarily in the input or output space?
   - Input only ⇒ **Encoder**
   - Output only ⇒ **Decoder**
   - The input and the output space are the same ⇒ **continue to (2)**
2. Is the transformer autoregressive?
   - Yes ⇒ **Decoder**
   - No ⇒ **continue to (3)**
3. Does the transformer use cross-attention?
   - Yes ⇒ **Decoder**
   - No ⇒ Probably a **Encoder** (but no guarantees)

Transformers: **Context**

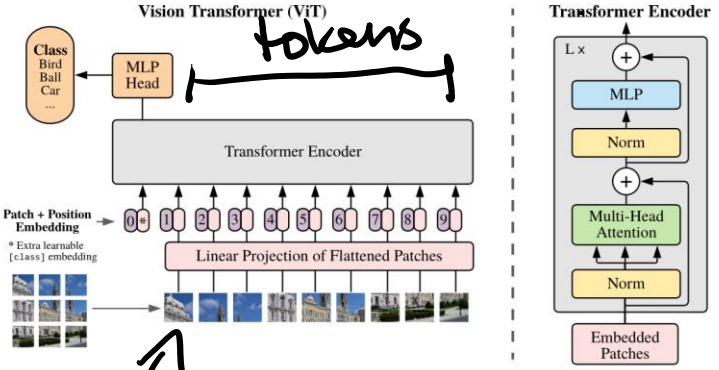


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification token" to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

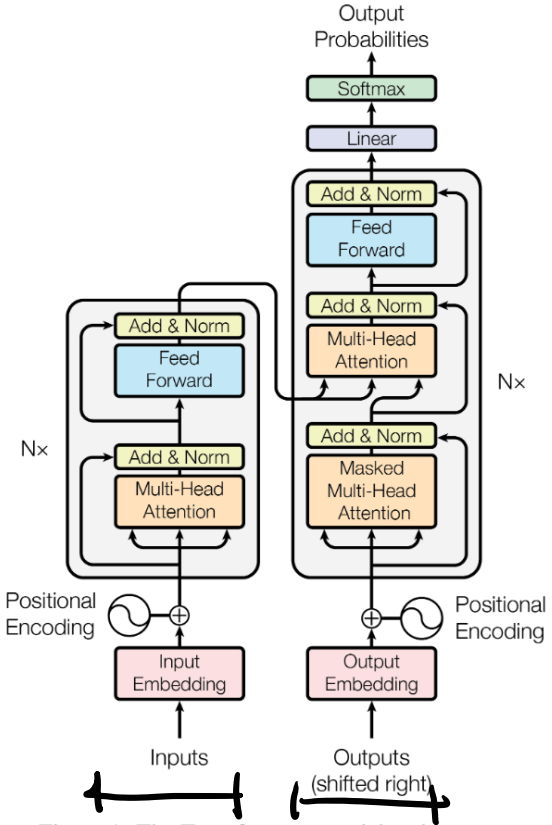


Figure 1: The Transformer - model architecture.

A more complicated transformer

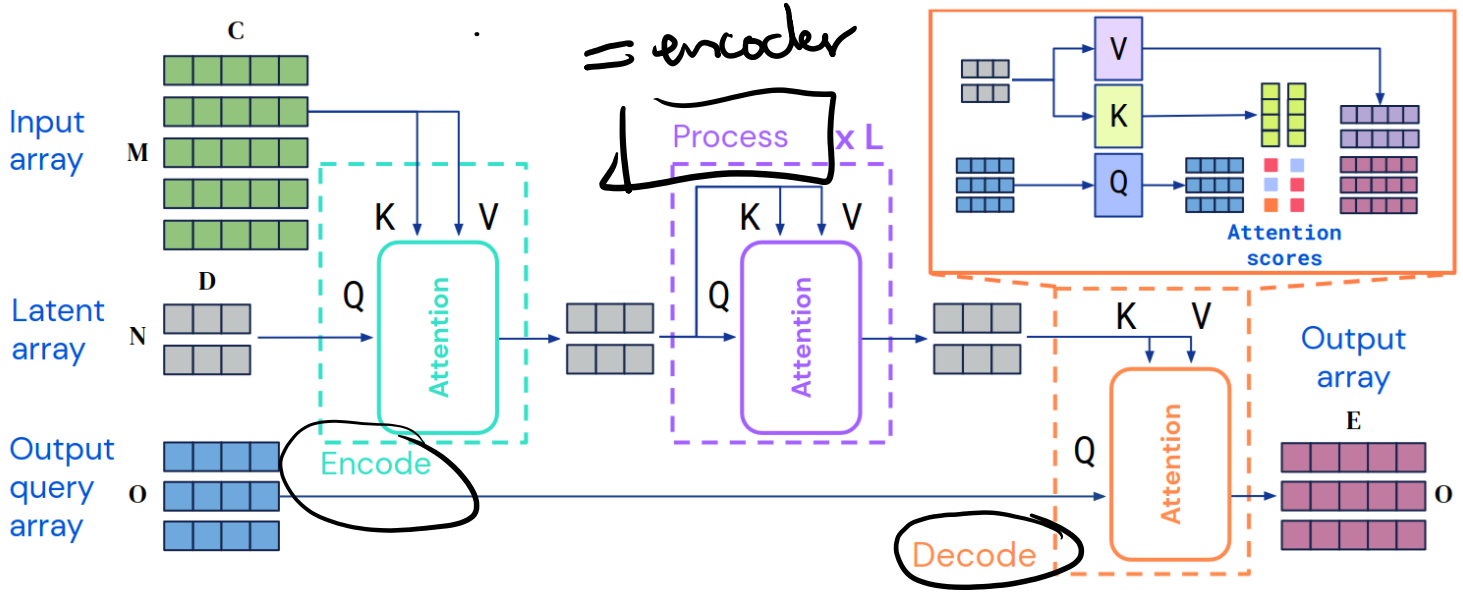
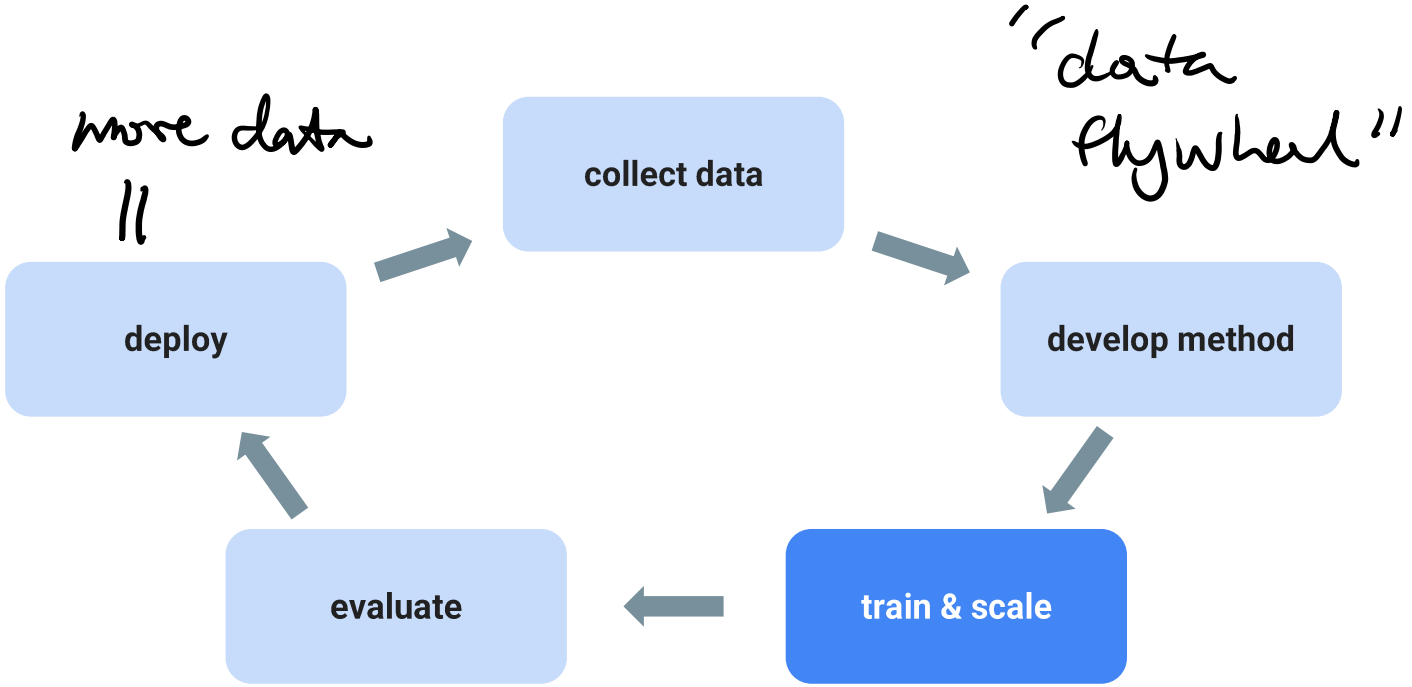


Figure 2: The Perceiver IO architecture. Perceiver IO maps arbitrary input arrays to arbitrary output arrays in a domain agnostic process. The bulk of the computation happens in a latent space whose size is typically smaller than the inputs and outputs, which makes the process computationally tractable even for very large inputs & outputs. See Fig. 5 for a more detailed look at encode, process, and decode attention.

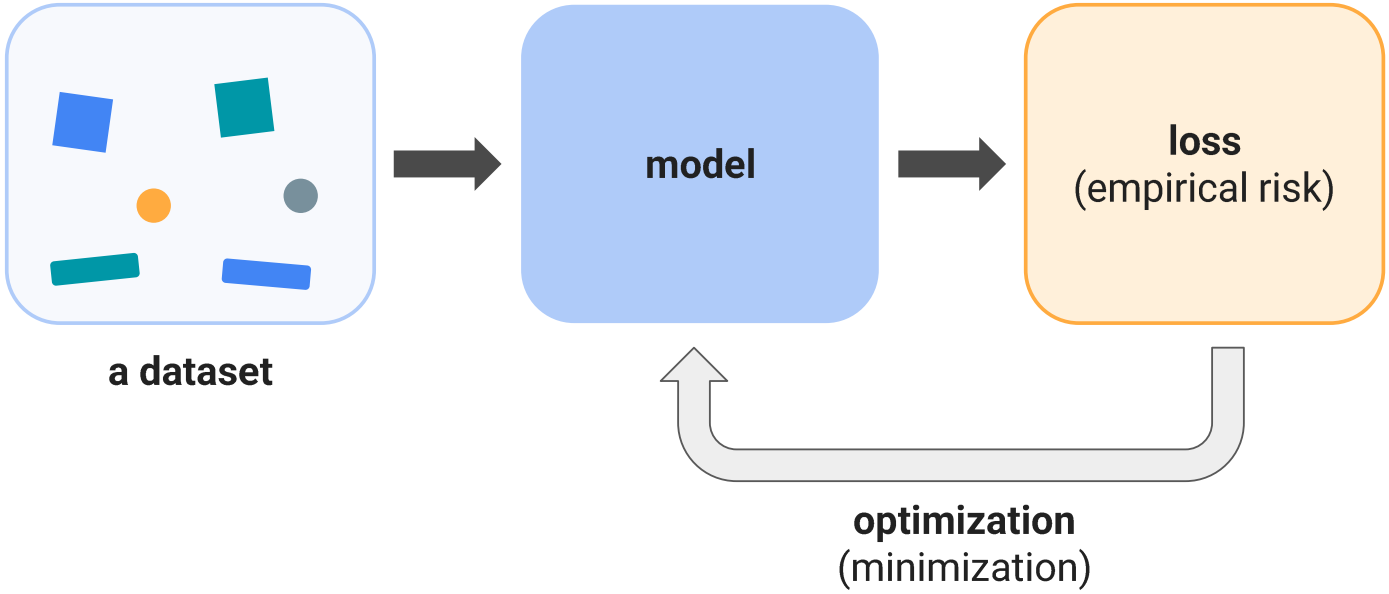
Crash Course: Applied Machine Learning

- Introduction to Machine Learning
  - Empirical Risk Minimization
  - Deep Learning & Transformers
- **The ML Life Cycle**
  - **Training & Evaluation**
  - Variance of Evaluation
- Learning at Scale
  - Foundation Models
  - Scaling Laws

The Machine Learning Life Cycle



Model Training



Train/Test

Why do we need a test set?

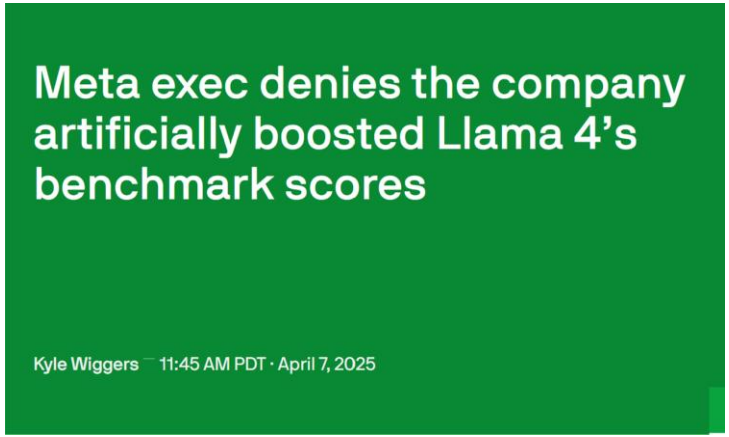
- representative of deployment
  <=> evaluate generalizability
- run experiments
  - compare methods

Train/Test

What properties should a test set have?

- similar distribution to deployment
- consistent, not noisy
  (low variance)
- cheap to run

Train/Test



LocalLLaMA Join
Posted by rrryougi · 1 yr. ago

reddit

"Serious issues in Llama 4 training. I Have Submitted My Resignation to GenAI"

Original post is in Chinese that can be found here. Please take the following with a grain of salt.

Content:

Despite repeated training efforts, the internal model's performance still falls short of open-source SOTA benchmarks, lagging significantly behind. Company leadership suggested blending test sets from various benchmarks during the post-training process, aiming to meet the targets across various metrics and produce a "presentable" result. Failure to achieve this goal by the end-of-April deadline would lead to dire consequences. Following yesterday's release of Llama 4, many users on X and Reddit have already reported extremely poor real-world test results.

As someone currently in academia, I find this approach utterly unacceptable. Consequently, I have submitted my resignation and explicitly requested that my name be excluded from the technical report of Llama 4. Notably, the VP of AI at Meta also resigned for similar reasons.

1.1K upvotes Comment Copy link

View 231 comments

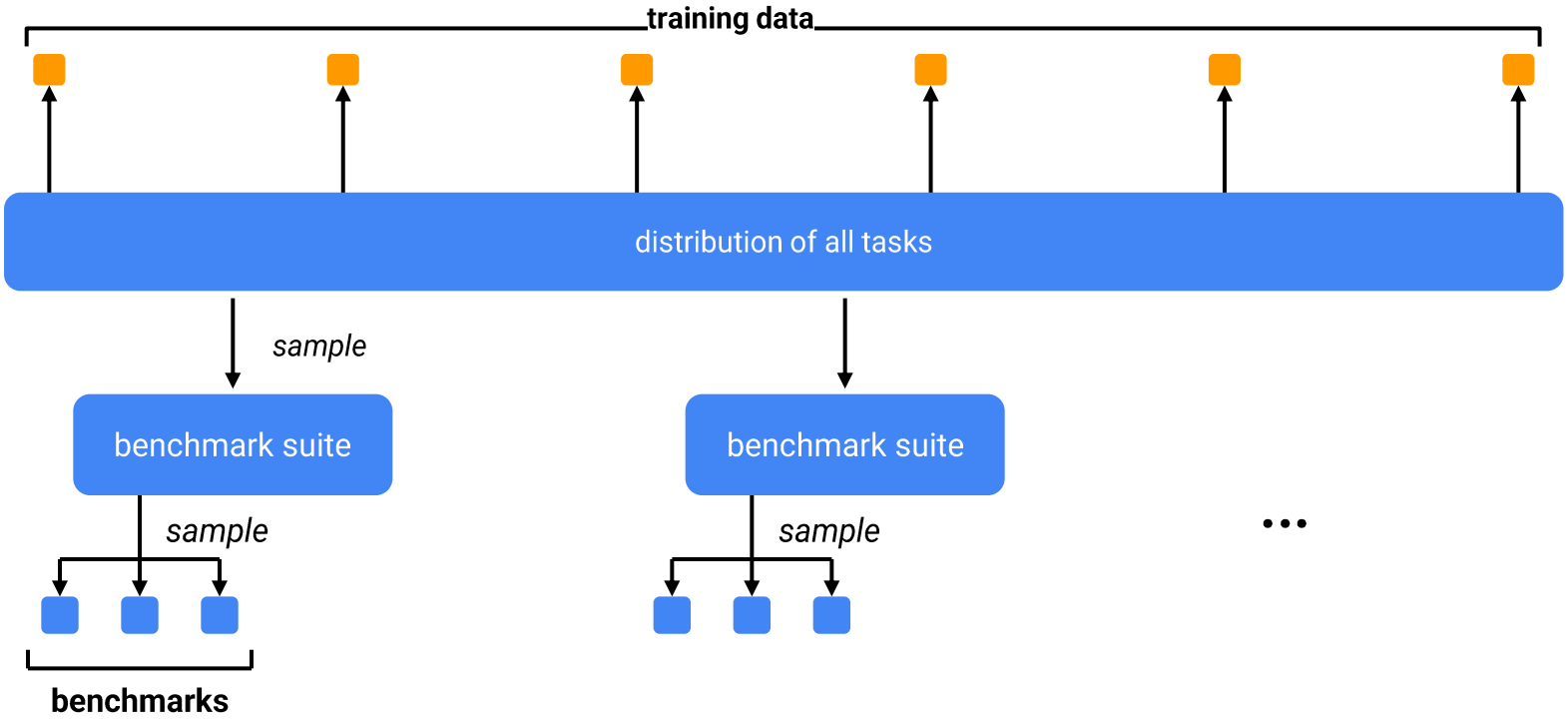
01-06-2026 | TECH

Yann LeCun: Meta 'fudged a little bit' when benchmark-testing Llama 4 model

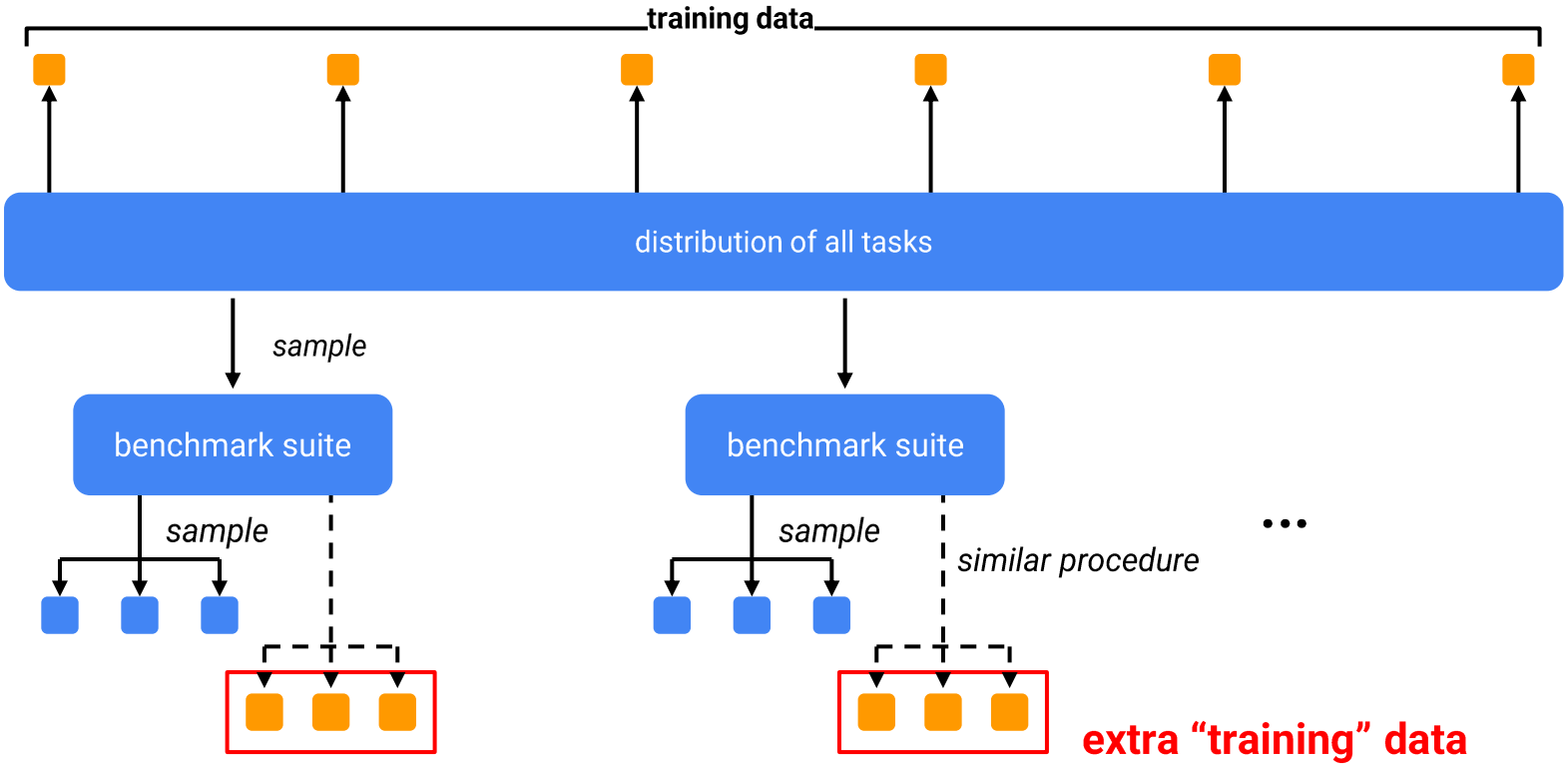
The testing sparked internal frustration about the progress of the Llama models.

SHARE ADD ON GOOGLE

Benchmaxxing: cheating without cheating



Benchmaxxing: cheating without cheating

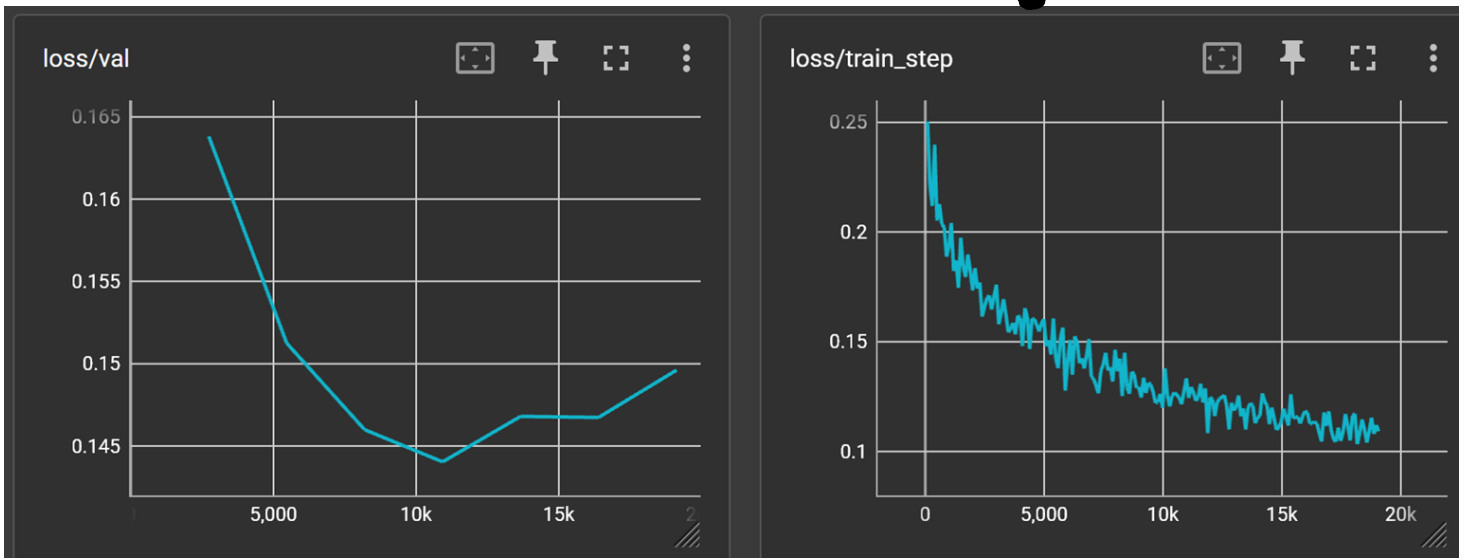


Train/Val/Test

Why do we need a val set?

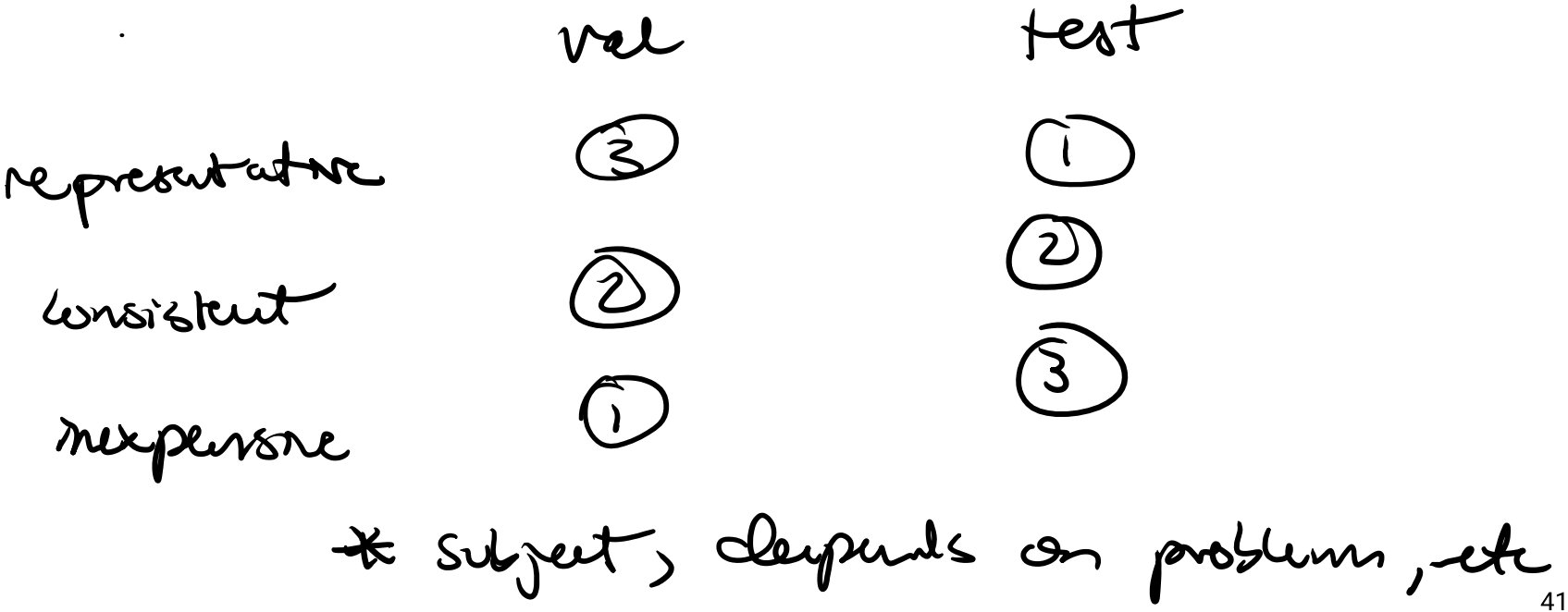
- monitor overfitting
- prevent information leakage from test set

Early Stopping



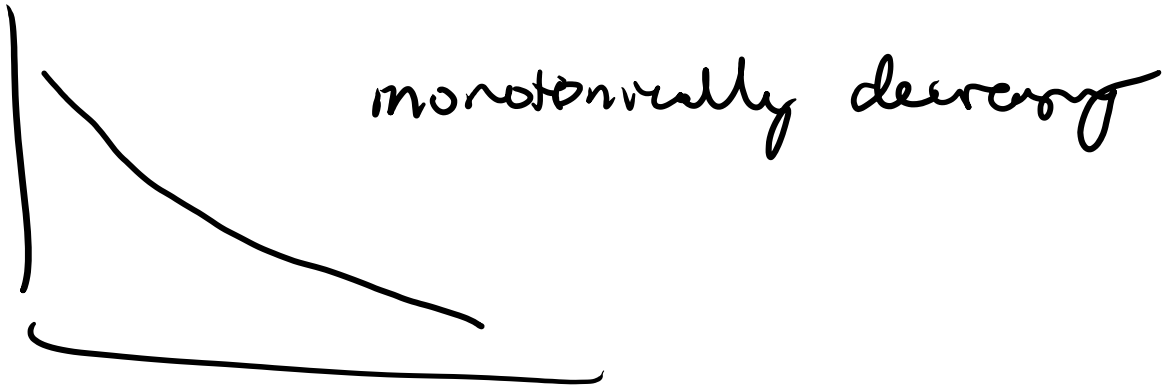
Train/Val/Test

What properties should a validation set have?



When can we validate on the test set?

- testing is not noisy / curve is very smooth
- not making decisions



Crash Course: Applied Machine Learning

- Introduction to Machine Learning
  - Empirical Risk Minimization
  - Deep Learning & Transformers
- **Training & Evaluation**
  - The ML Life Cycle
  - **Variance of Evaluation**
- Learning at Scale
  - Foundation Models
  - Scaling Laws

Review: **Variance**

Random var. $X \in \mathbb{R}$

$\mathbb{E}[X]$ = expected value

$\text{Var}(X) = \mathbb{E}[(X - \mathbb{E}[X])^2]$ ← prefer

$= \mathbb{E}[X^2] - \mathbb{E}[X]^2$

$\text{stddev} = \sqrt{\text{Var}(X)}$

Sample stddev

$$\frac{\sum_i (x_i - \bar{x}_i)^2}{n-1}$$

Review: **Standard Error (SE)**

$$\frac{\sum_i x_i}{n} \longrightarrow \mathcal{N}\left(E[X], \frac{Var(X)}{n}\right)$$

$$\text{stderr (SE)} = \sqrt{\frac{Var(X)}{n}} = \frac{\text{stddev}(x)}{\sqrt{n}}$$

(n-1)

Key Takeaways

- Data means distributions
- Transformers are all you need
  - There are different types; pay attention to encoder vs decoder, autoregressive vs feedforward
- You should probably have a val & test set
  - More data ⇒ higher # of test samples, but smaller %

Next Class

- Applied ML Continued
  - Variance of Evaluation
  - Method Comparisons
  - Scaling & Scaling Laws
- Lab 2 Overview