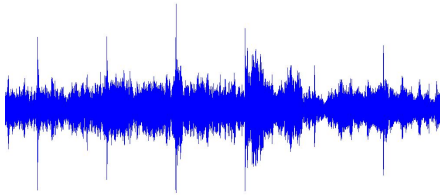
Introductory Seminar on Artificial Intelligence and Machine Learning

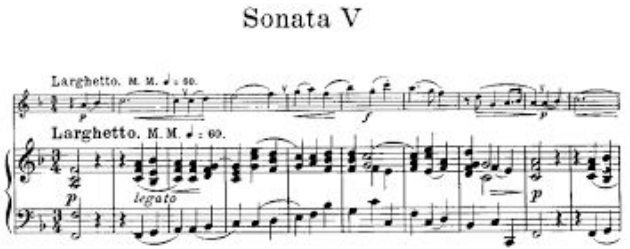
Emanuele Ledda, Cagliari Digital Lab 2024 - Day 4

Recurrent Neural Networks (RNN)

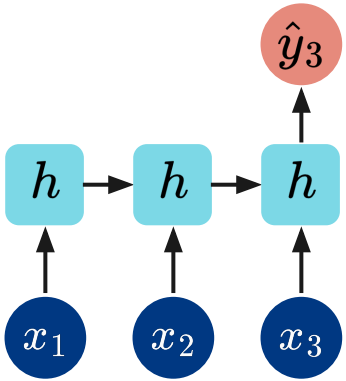
The Notion of Sequence

One Piece (stylized in all caps) is a Japanese manga series written and illustrated by Eiichiro Oda. It has been serialized in Shueisha's *shōnen* manga magazine *Weekly Shōnen Jump* since July 1997, with its individual chapters compiled in 108 *tankōbon* volumes as of March 2024.

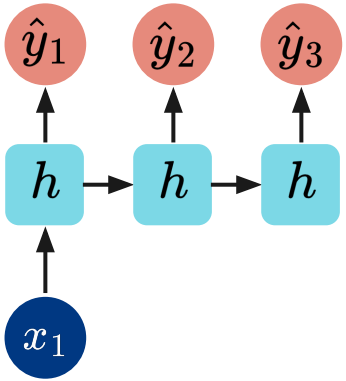




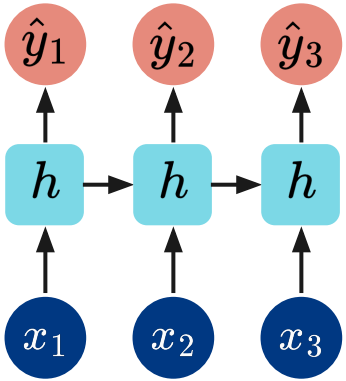
Sequence modelling



Many-to-One



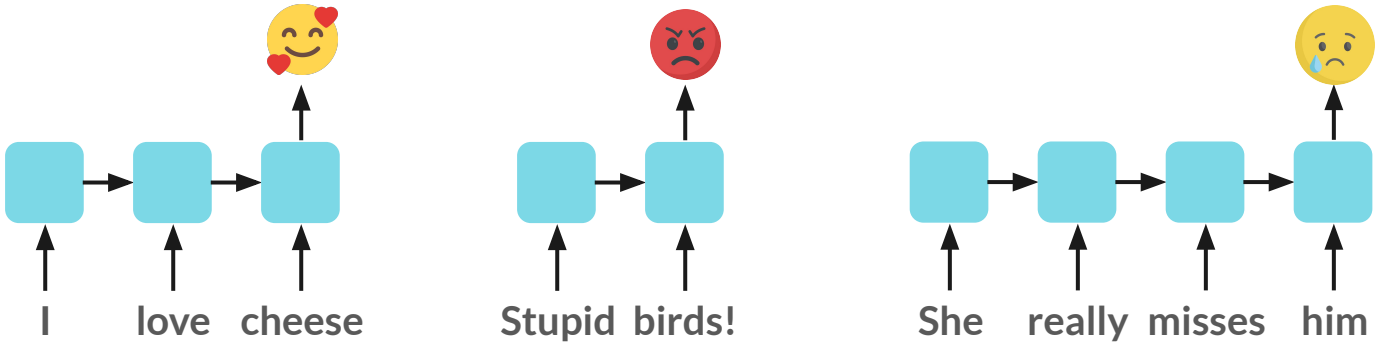
One-to-Many



Many-to-Many

Many-to-One

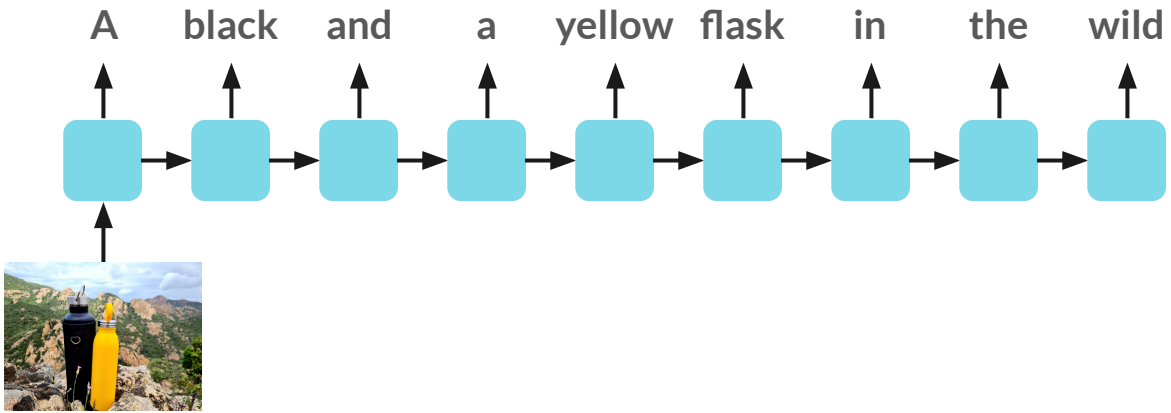
- In **many-to-one** sequence modelling the **input represents a Serie** and the **output represent a single value** (e.g., Sentiment Analysis)
- The **input** has an **arbitrary length** (not specified a priori)



One-to-Many

- In one-to-many sequence modelling the **input represents a single 'object'** and the **output represent a Serie** (e.g., Image Captioning)

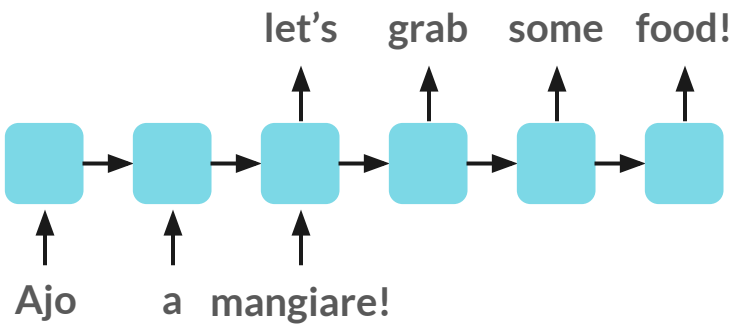
- The **output** has an **arbitrary length** (not specified a priori)



Many-to-Many

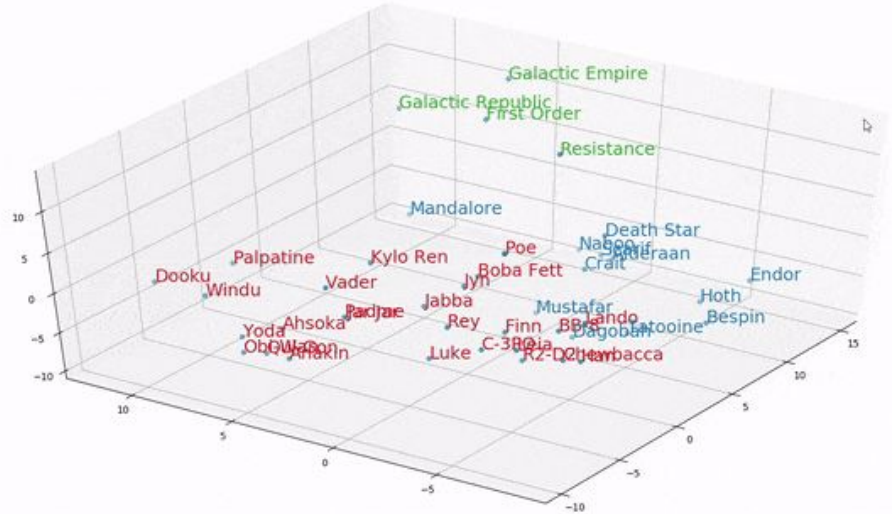
- In one-to-many sequence modelling the **both the input and the output represent Series** (e.g., Machine Translation)

- **Both the output and the input has arbitrary length**

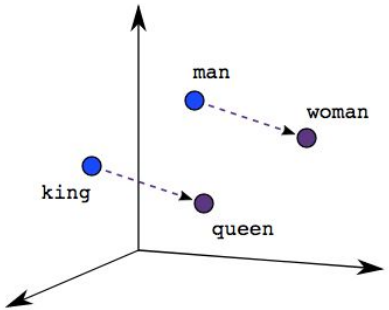


Natural Language Processing

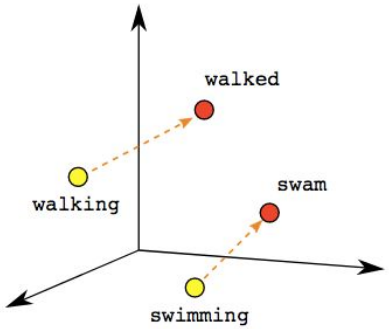
- If we want to encode natural language in deep learning models we need a strategy for encoding each word
- Generally, we use standard pre-computed embeddings which are dictionaries mapping words in multidimensional vectors



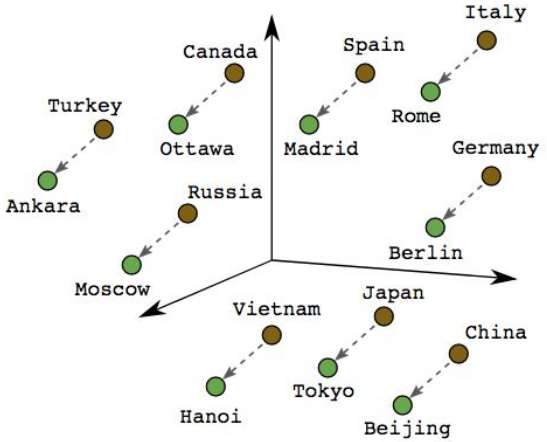
Word Embedding



Male-Female



Verb Tense

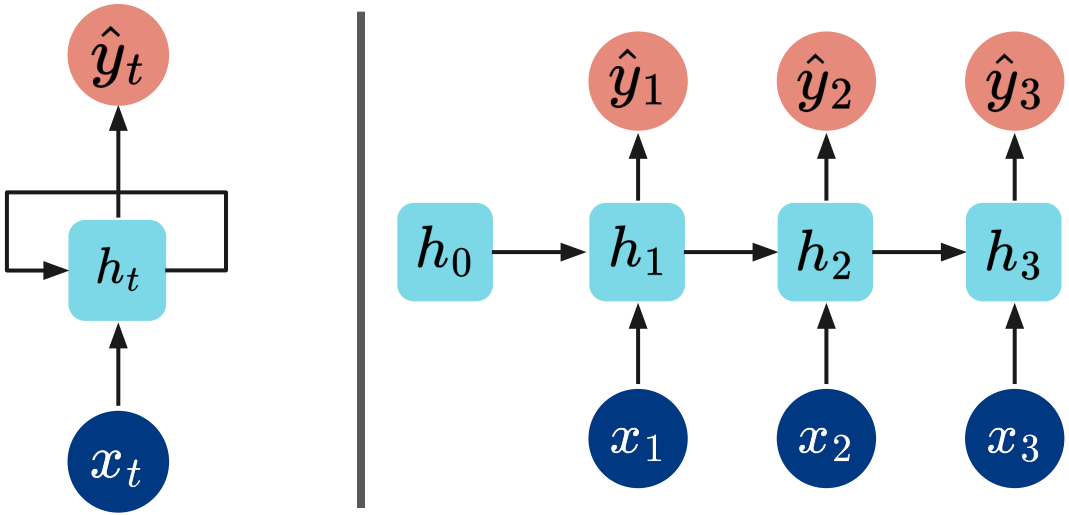


Country-Capital

Recursive Neural Network (RNN)

$$\hat{y}_t = f(x_t, h_{t-1})$$

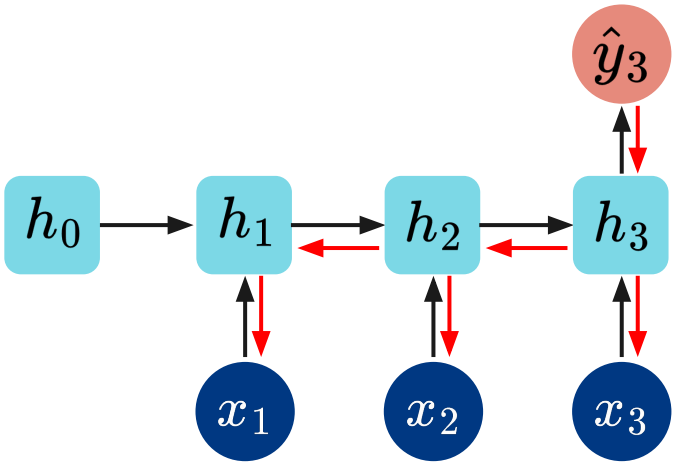
$$h_t = f^W(x_t, h_{t-1})$$



- The **hidden state encodes** in some sense the information about the **past tokens**
- At each timestep we update the hidden state

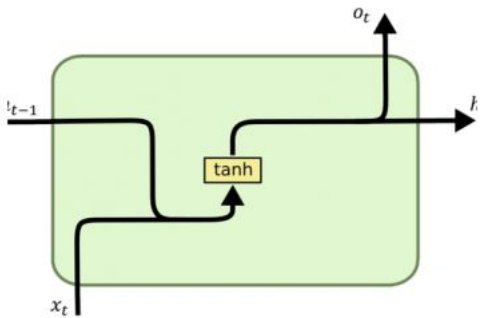
Backpropagation Through Time (BPTT)

$$\frac{\partial \mathcal{L}(\hat{y}_k, y)}{\partial h_k} \cdot \frac{\partial h_k}{\partial h_{k-1}} \cdot \frac{\partial h_{k-1}}{\partial h_{k-2}} \cdots \frac{\partial h_1}{\partial W_h}$$

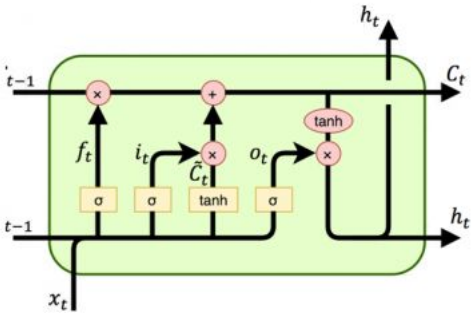


- The gradients propagate throughout all the time-steps
- High risk of vanishing or exploding gradients due to the serie of multiplication from the derivative chain rule
- Many solutions try to develop recurrent units which combine wisely the activation functions for avoiding exploding/vanishing gradients

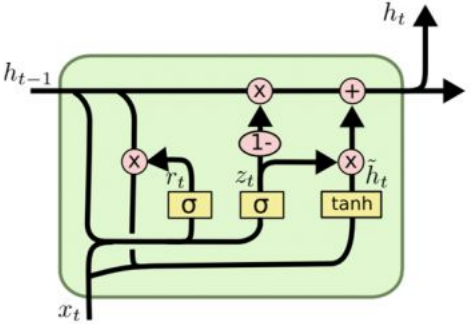
Different types of RNNs



Regular RNN



LSTM

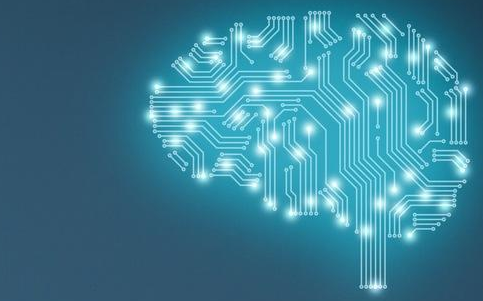


GRU

https://colah.github.io/posts/2015-08-Understanding-LSTMs/

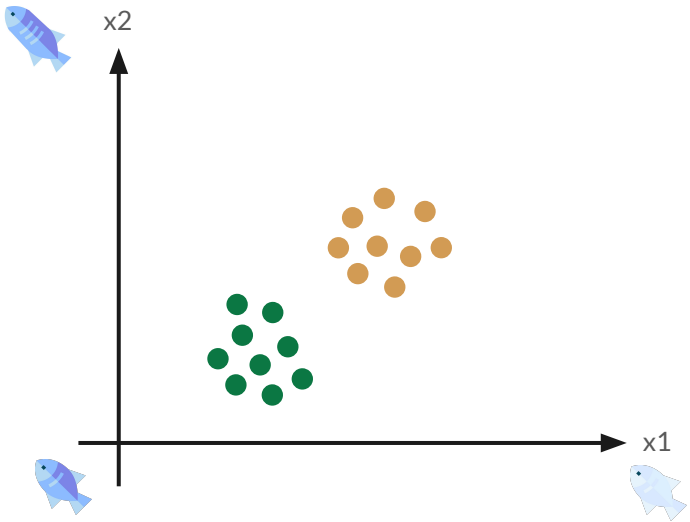
Modern AI Ides

- Generative Models
- Attention and Transformers
- Self-Supervision
- Multi Modality
- Foundation Models

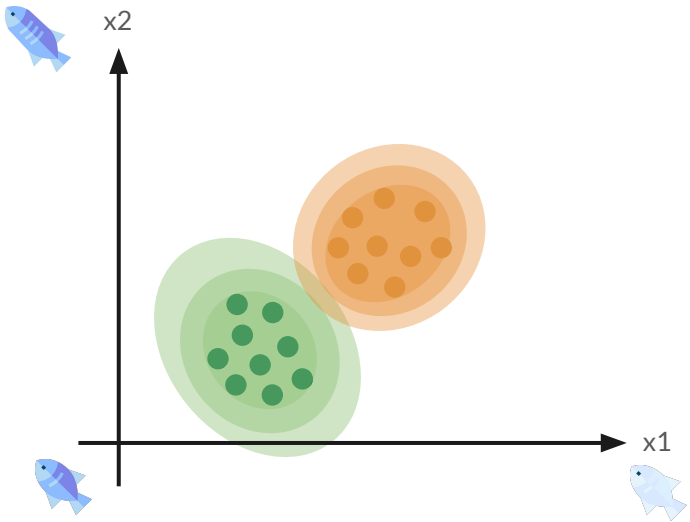


Generative Models

Generative Modelling

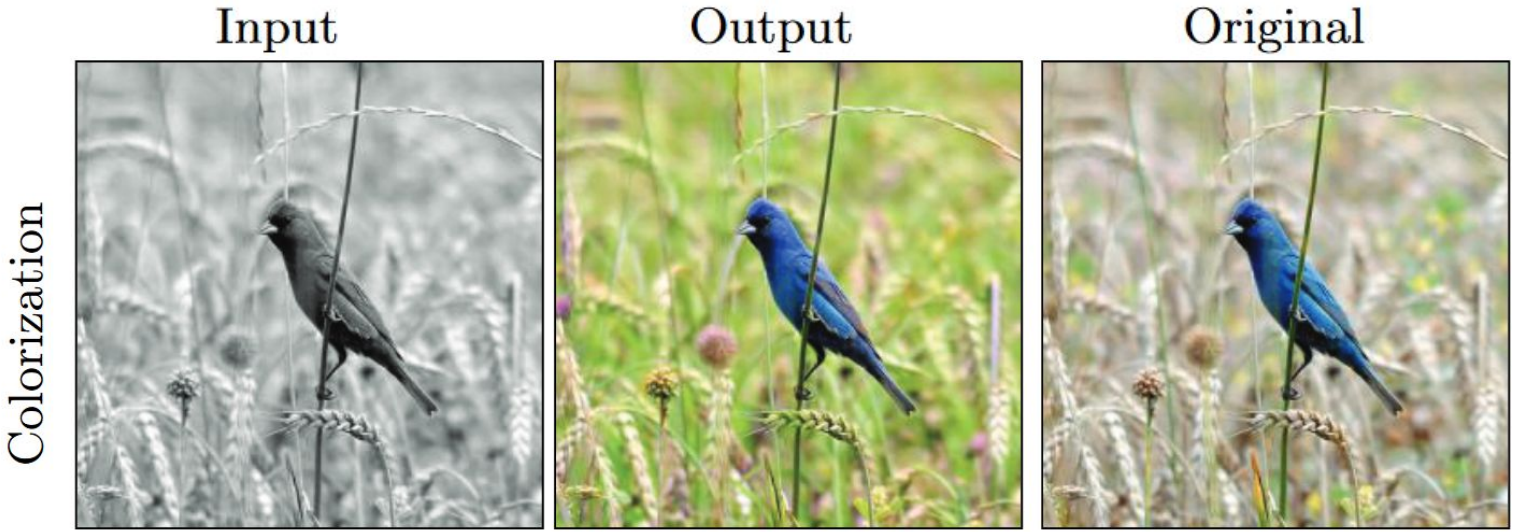


Generative Modelling

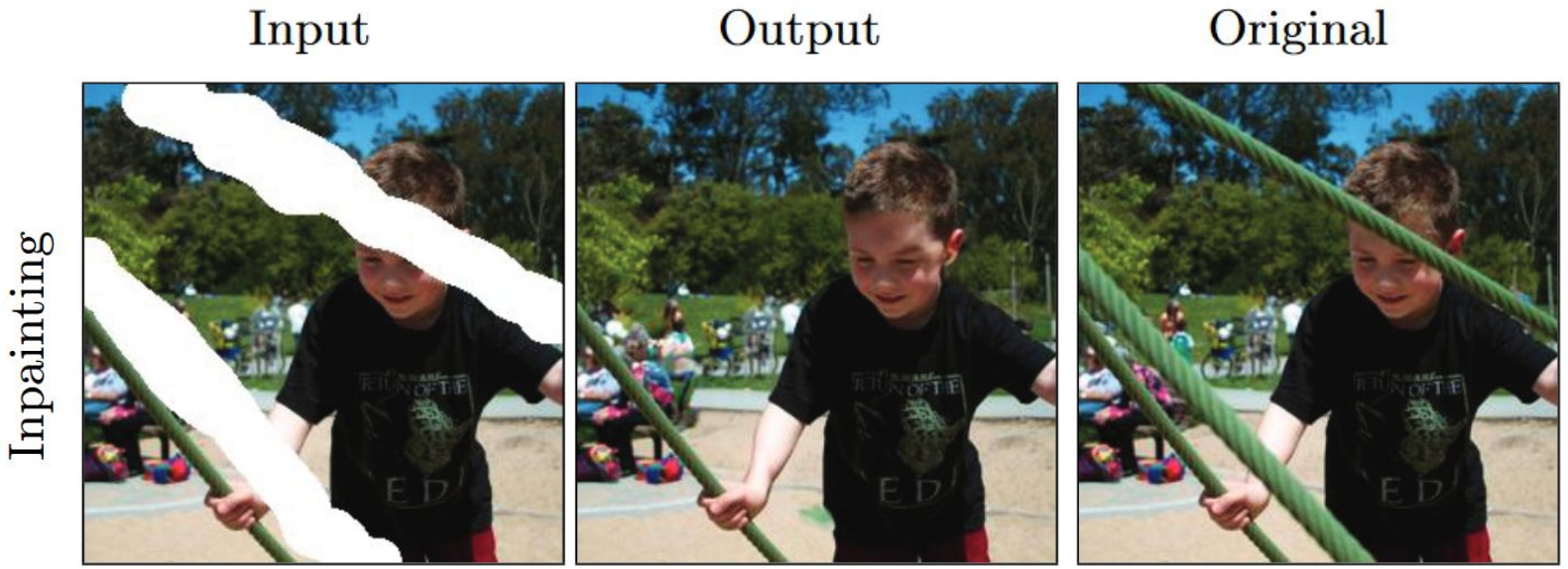


- The idea is to imagine the data as if they were sampled from a **generative distribution**
- Generative models just try to **fit this probability distribution**
- One fitted, one can **sample from this distribution** for **creating new samples**

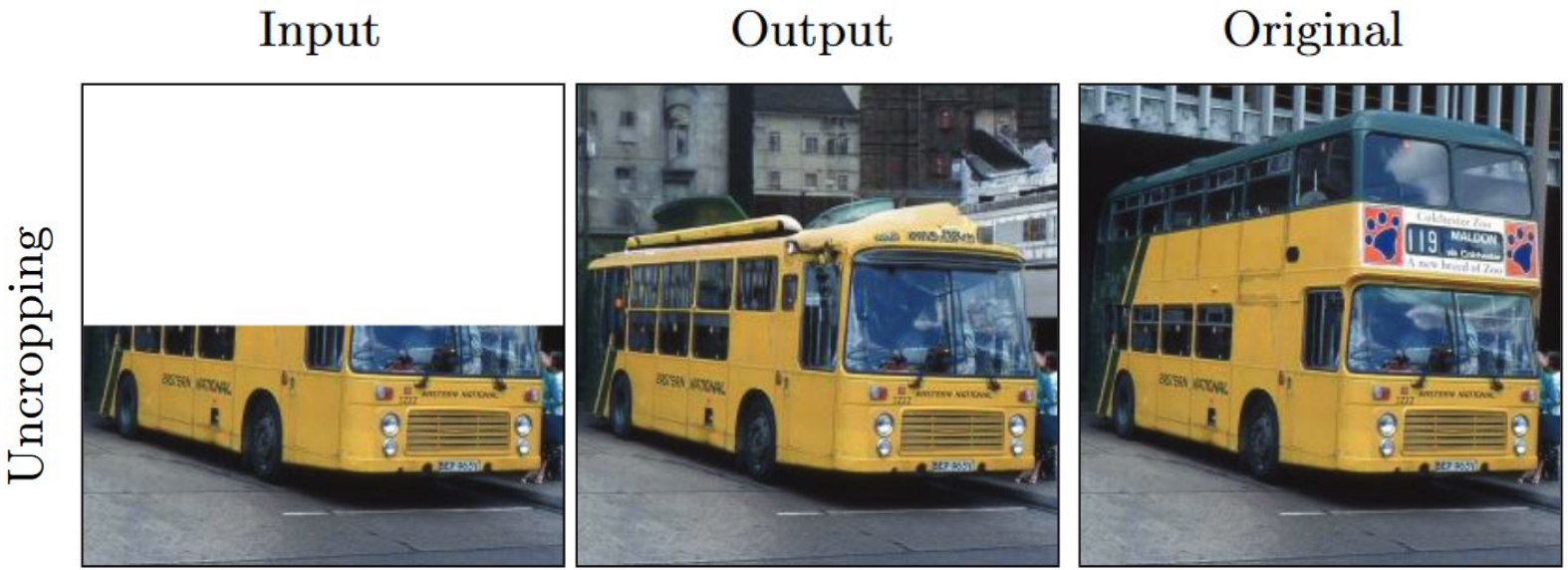
Applications of Generative Models



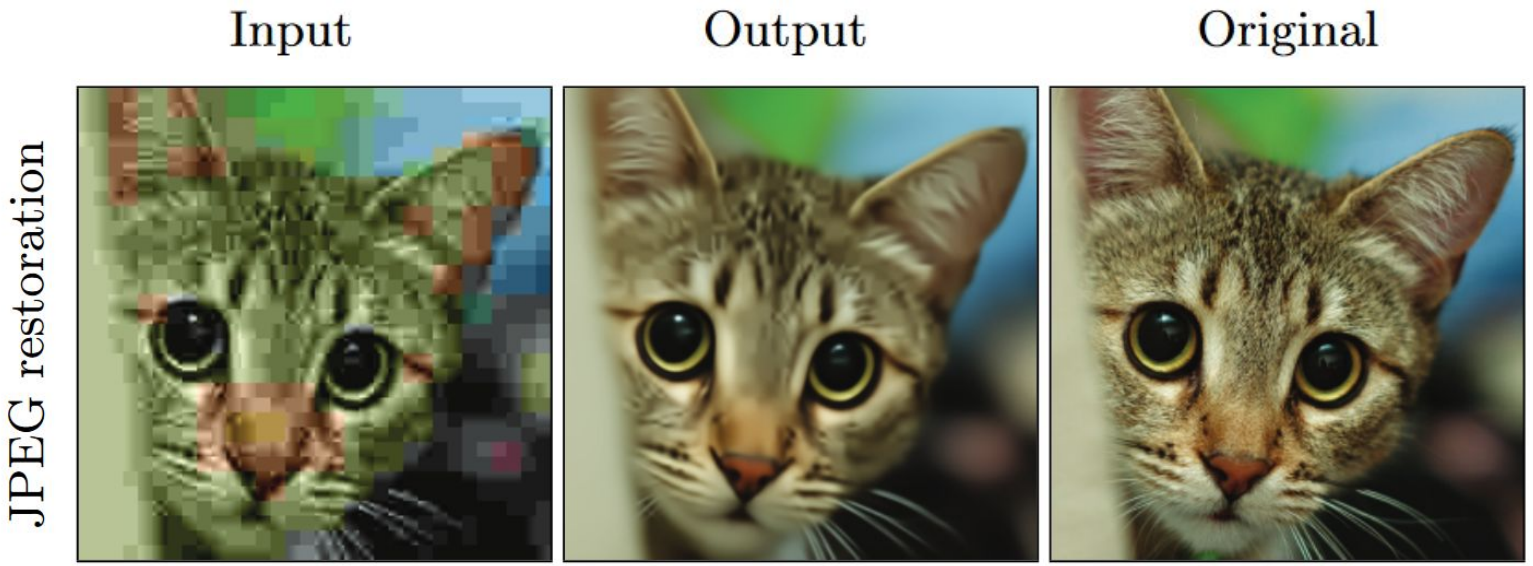
Applications of Generative Models



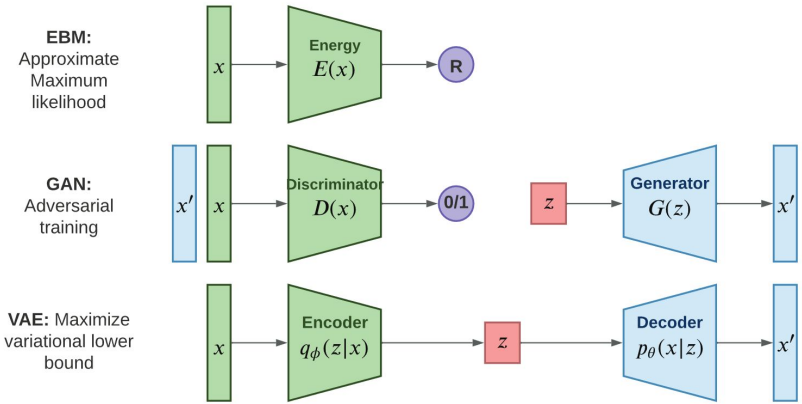
Applications of Generative Models

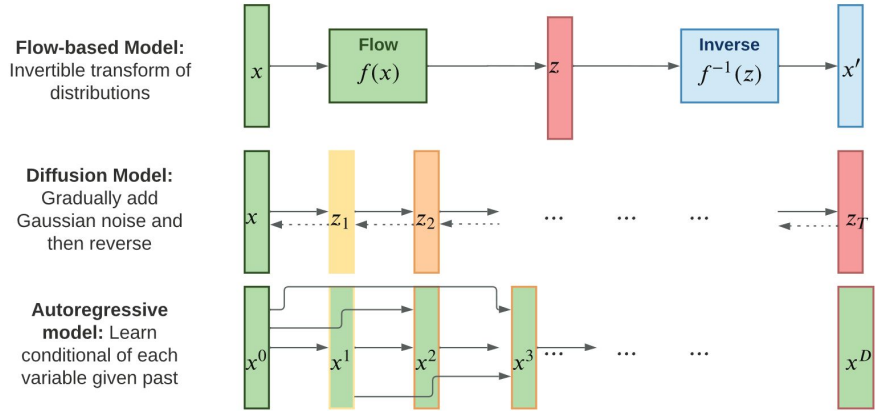


Applications of Generative Models

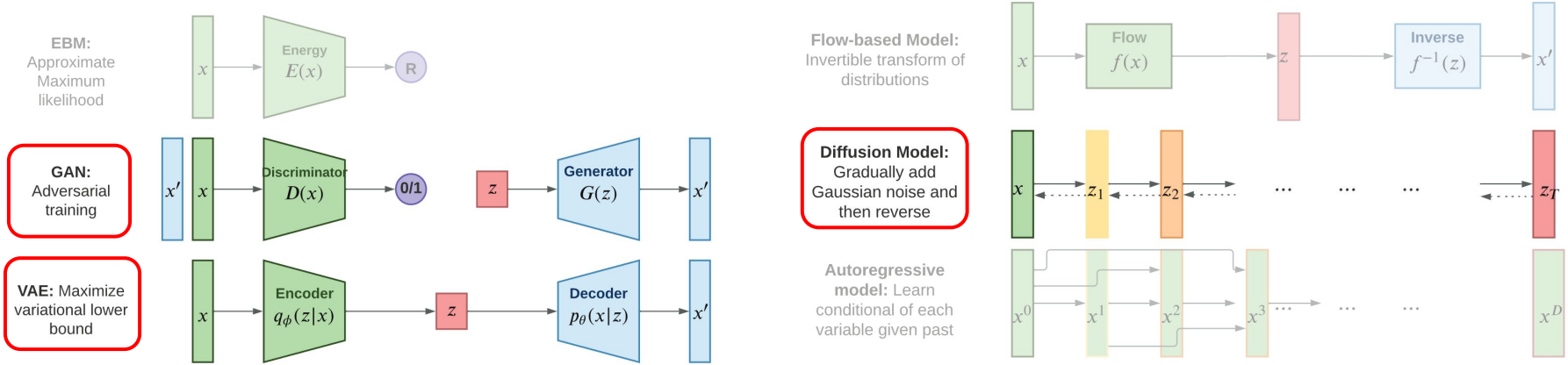


Types of Generative Models



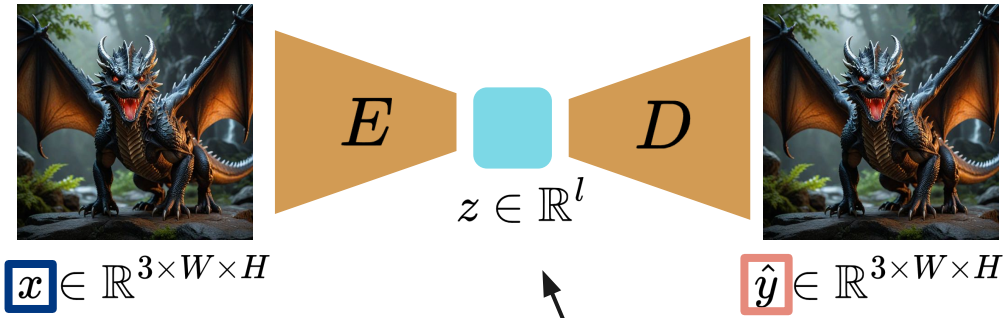


Types of Generative Models



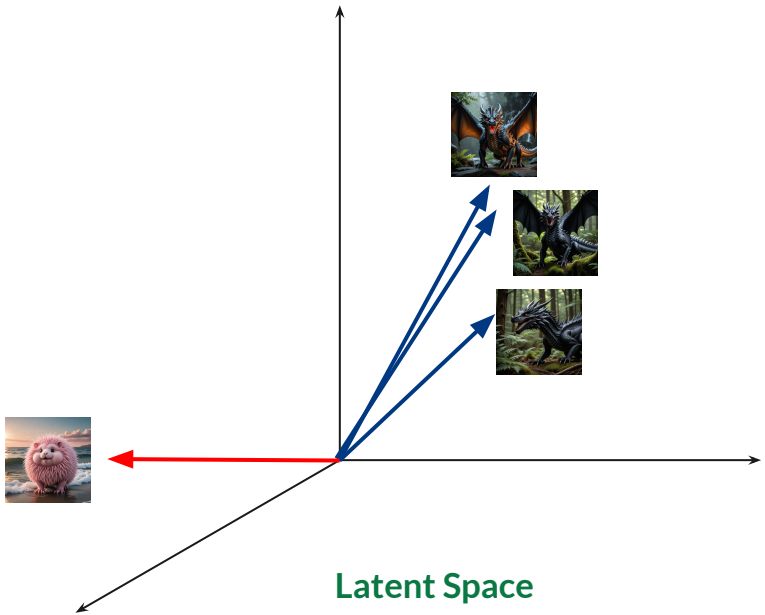
Auto-Encoders (AEs) - idea

- I want to “compress” all the information describing data points in a space with smaller dimensionality



- I can fit a model which at first **“compress”** and then tries to **“reconstruct” the input** by using an **Encoder-Decoder network**
- The only way the model can minimize the loss is to **find a compact form for describing the input**!

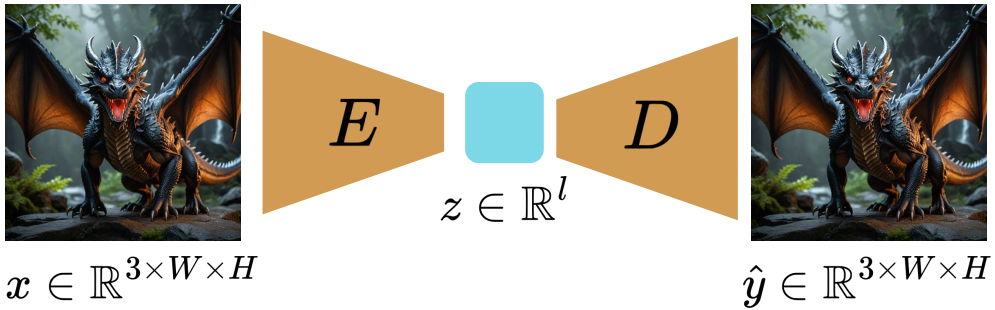
Auto-Encoders (AEs)



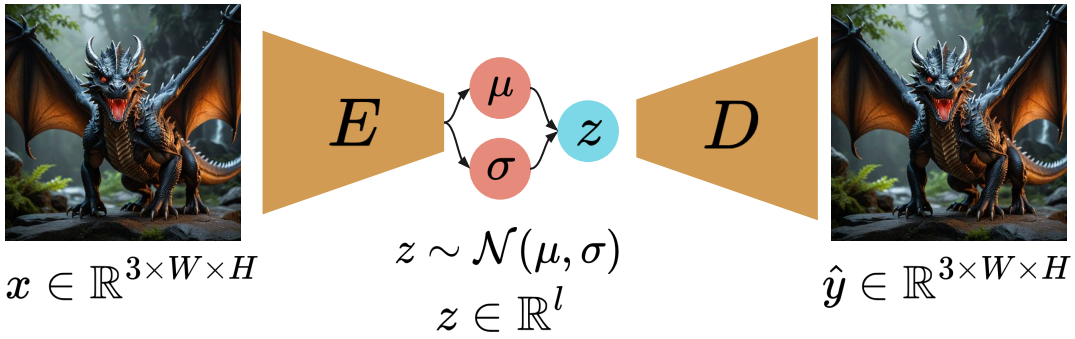
- In the latent space, similar vectors encode similar data…
- So, by exploring the latent space one can eventually find similar data, which when reconstructed will create new data
- But can we do better?

Variational Auto-Encoders (VAEs)

- Instead of just encoding…

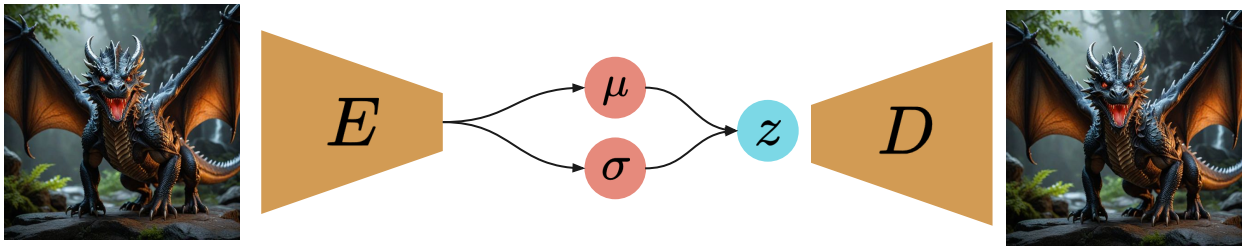


Variational Auto-Encoders (VAEs)



- Instead of just encoding…
- We can try to directly encode the **probability distribution of the latent space**!
- But we need sophisticated strategies for backpropagating

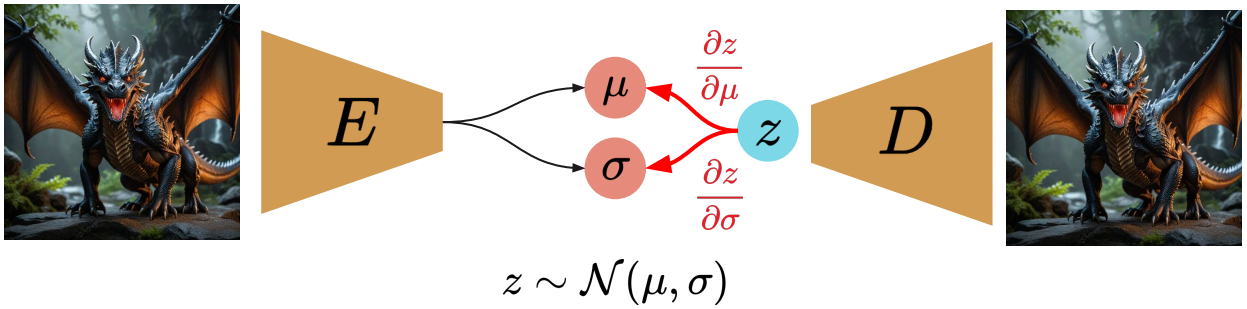
Reparametrization Trick



$$z \sim \mathcal{N}(\mu, \sigma)$$

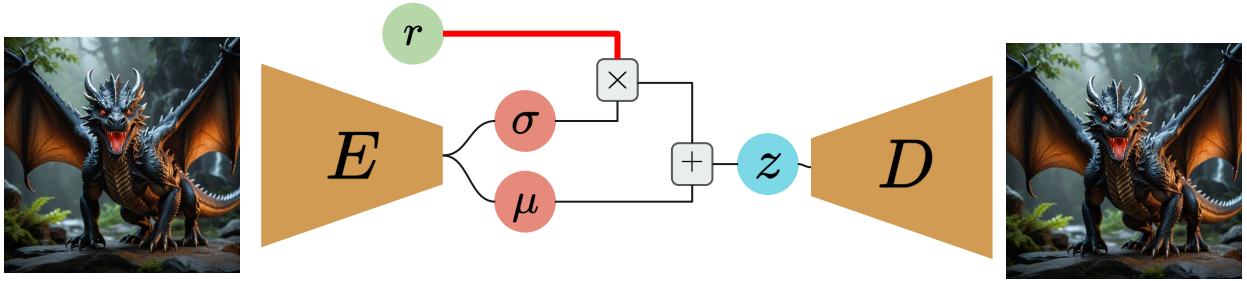
- I need to compute the derivative with respect to the distribution parameters….

Reparametrization Trick



- I need to compute the **derivative with respect to the distribution parameters….**
- **But we cannot compute the derivative**!

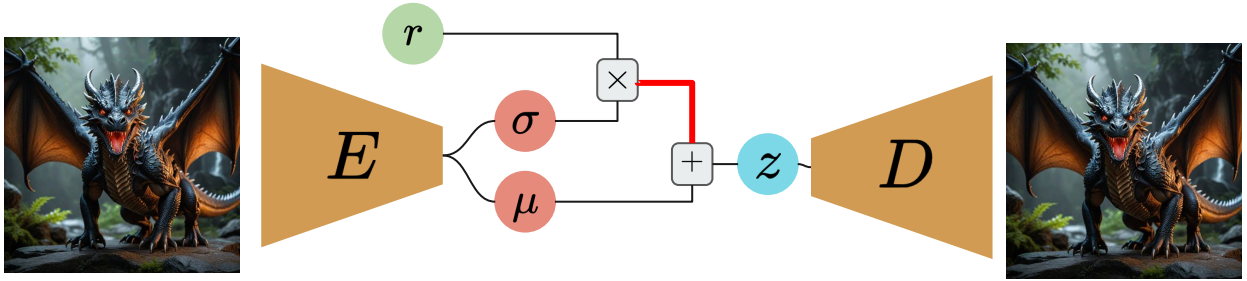
Reparametrization Trick



$$r \sim \mathcal{N}(0, 1)$$

- I need to compute the **derivative with respect to the distribution parameters**….
- But we **cannot compute the derivative**!
- However, we can indeed **separate the random part**

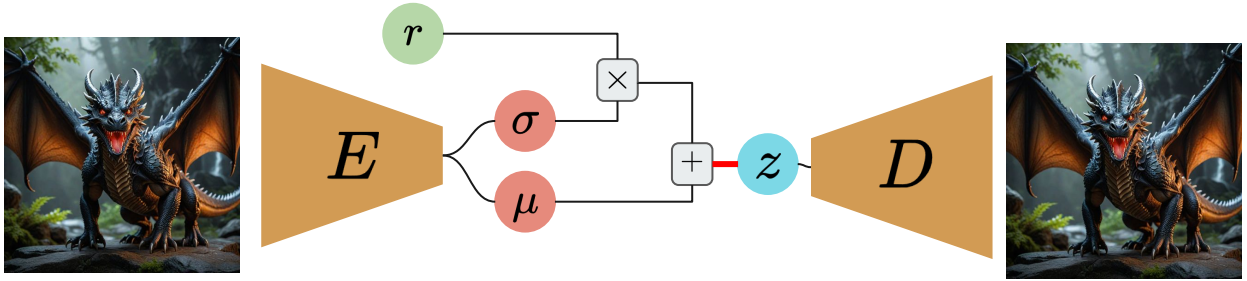
Reparametrization Trick



$$r \sim \mathcal{N}(0, 1)$$
$$r \cdot \sigma \sim \mathcal{N}(0, \sigma)$$

- I need to compute the **derivative with respect to the distribution parameters**….
- But we **cannot compute the derivative**!
- However, we can indeed **separate the random part**

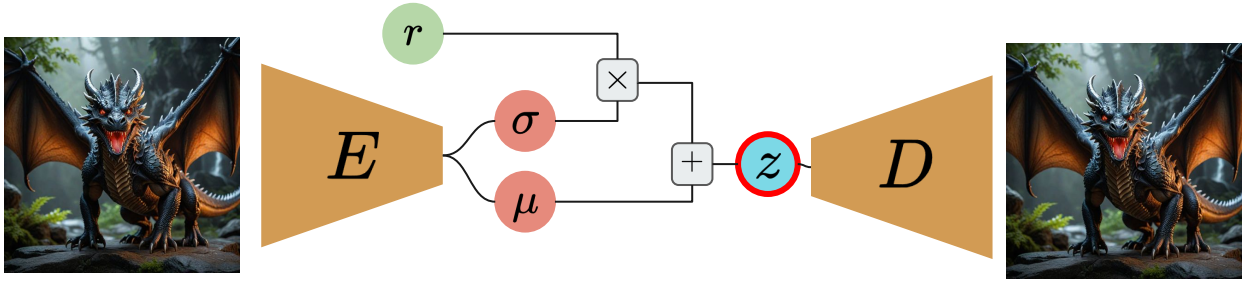
Reparametrization Trick



$$r \sim \mathcal{N}(0, 1)$$
$$r \cdot \sigma \sim \mathcal{N}(0, \sigma)$$
$$\mu + r \cdot \sigma \sim \mathcal{N}(\mu, \sigma)$$

- I need to compute the **derivative with respect to the distribution parameters**….
- But we **cannot compute the derivative**!
- However, we can indeed **separate the random part**

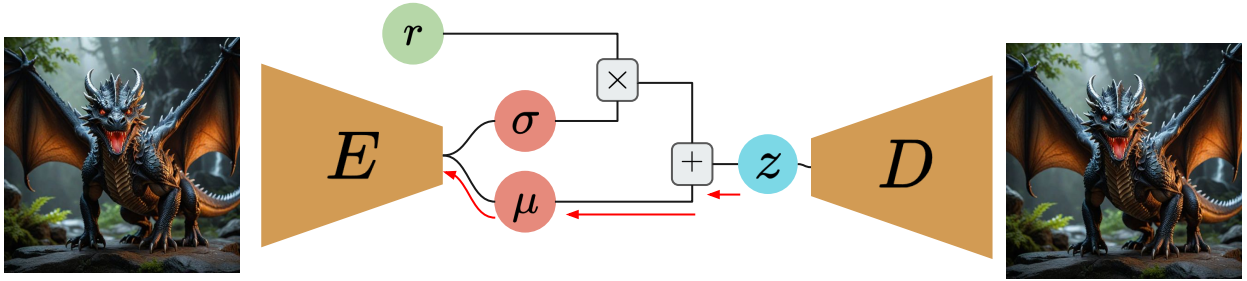
Reparametrization Trick



$$r \sim \mathcal{N}(0, 1)$$
$$r \cdot \sigma \sim \mathcal{N}(0, \sigma)$$
$$\mu + r \cdot \sigma \sim \mathcal{N}(\mu, \sigma)$$
$$z \sim \mathcal{N}(\mu, \sigma)$$

- I need to compute the **derivative with respect to the distribution parameters**….
- But we **cannot compute the derivative**!
- However, we can indeed **separate the random part**

Reparametrization Trick



$$r \sim \mathcal{N}(0,1)$$
$$r \cdot \sigma \sim \mathcal{N}(0,\sigma)$$
$$\mu + r \cdot \sigma \sim \mathcal{N}(\mu,\sigma)$$
$$z \sim \mathcal{N}(\mu,\sigma)$$

By doing so we can now compute the derivatives!

- I need to compute the **derivative with respect to the distribution parameters**….
- But we **cannot compute the derivative**!
- However, we can indeed **separate the random part**

VAEs Latent Space Exploration

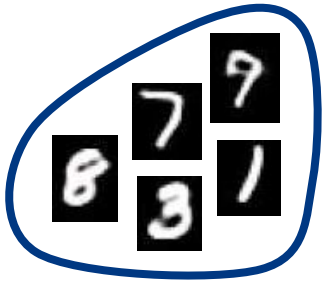
Generative Adversarial Networks (GANs) - idea

Legit Picasso

Fake Picasso

Generative Adversarial Networks (GANs) - idea

$(x, y) \sim \mathcal{G}(\mathcal{X}, \mathcal{Y}), \quad \forall (x, y) \in \mathcal{D}$



$\mathcal{D}_{train}$

Legit Data

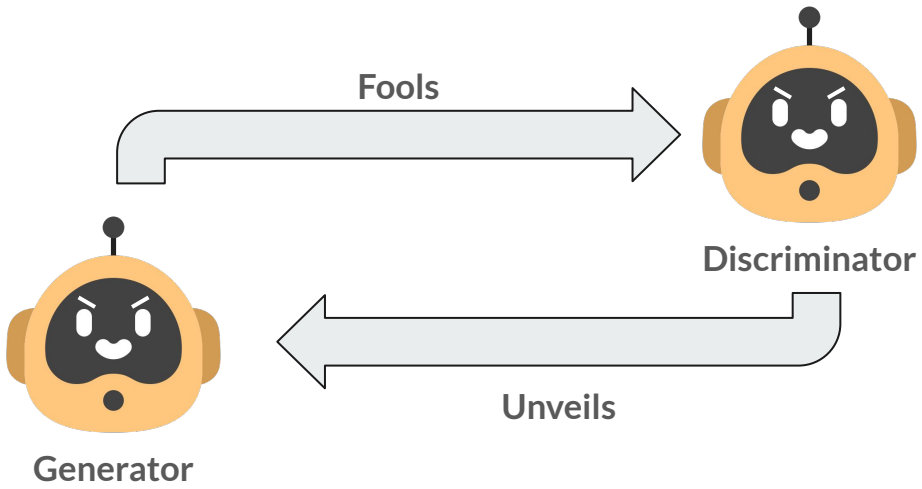
Generator

(x', y)

Fake Picasso

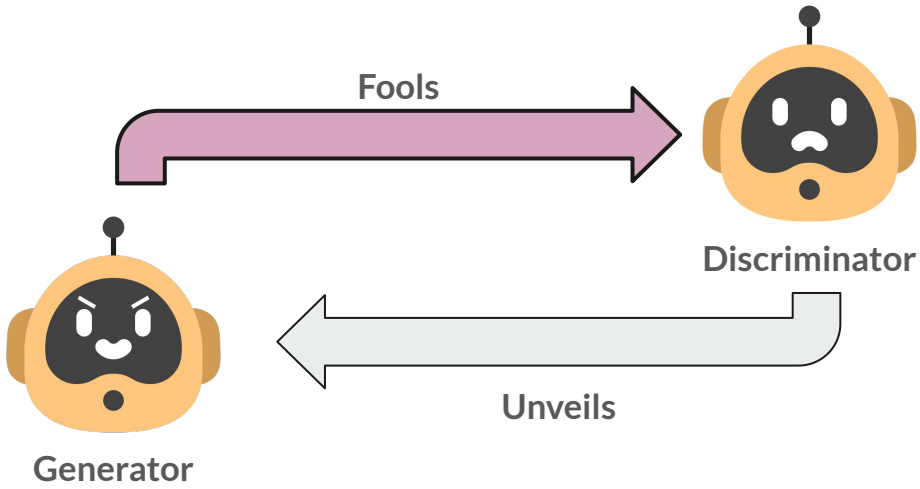
Discriminator

Generative Adversarial Networks (GANs) - idea



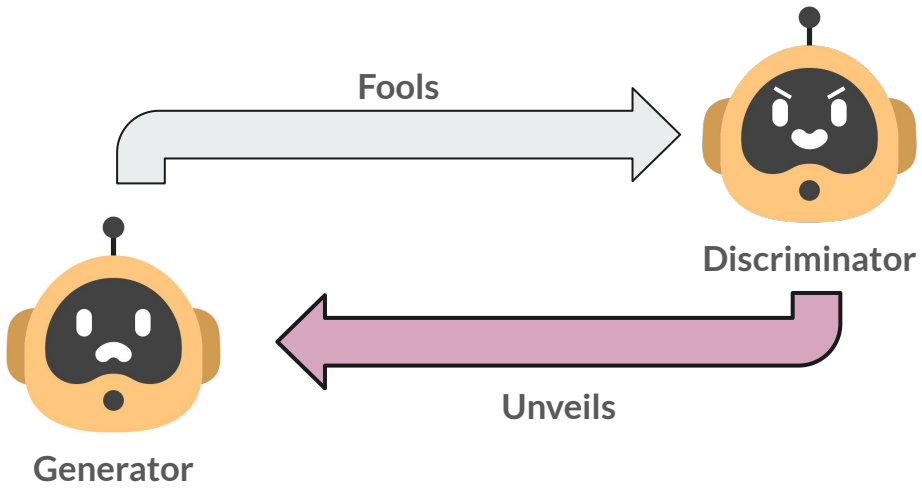
- The **Generator** and the **Discriminator** play a **game against each other**

Generative Adversarial Networks (GANs) - idea



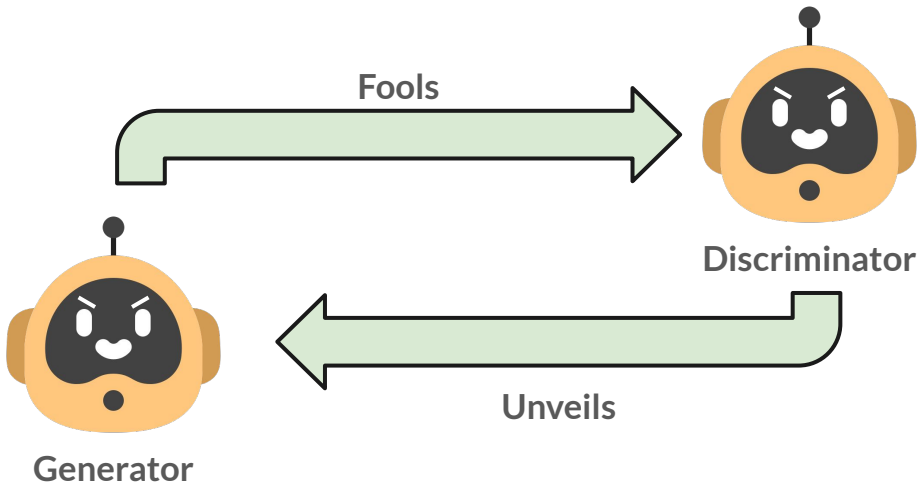
- The **Generator** and the **Discriminator** play a **game against each other**
- Sometimes the **generator** is able to **fool the discriminator**...

Generative Adversarial Networks (GANs) - idea



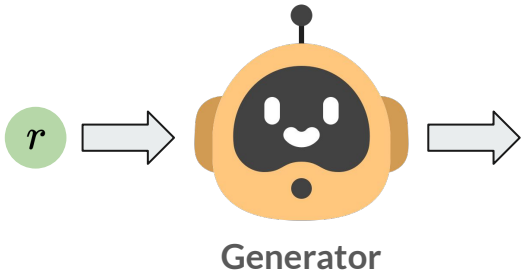
- The **Generator** and the **Discriminator** play a **game against each other**
- Sometimes the **generator** is able to **fool the discriminator**...
- And sometimes the **discriminator** can **unveil the generator**

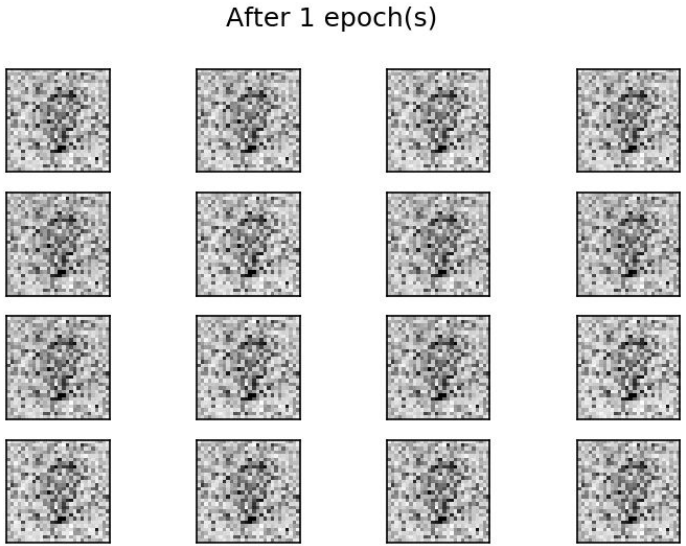
Generative Adversarial Networks (GANs) - idea



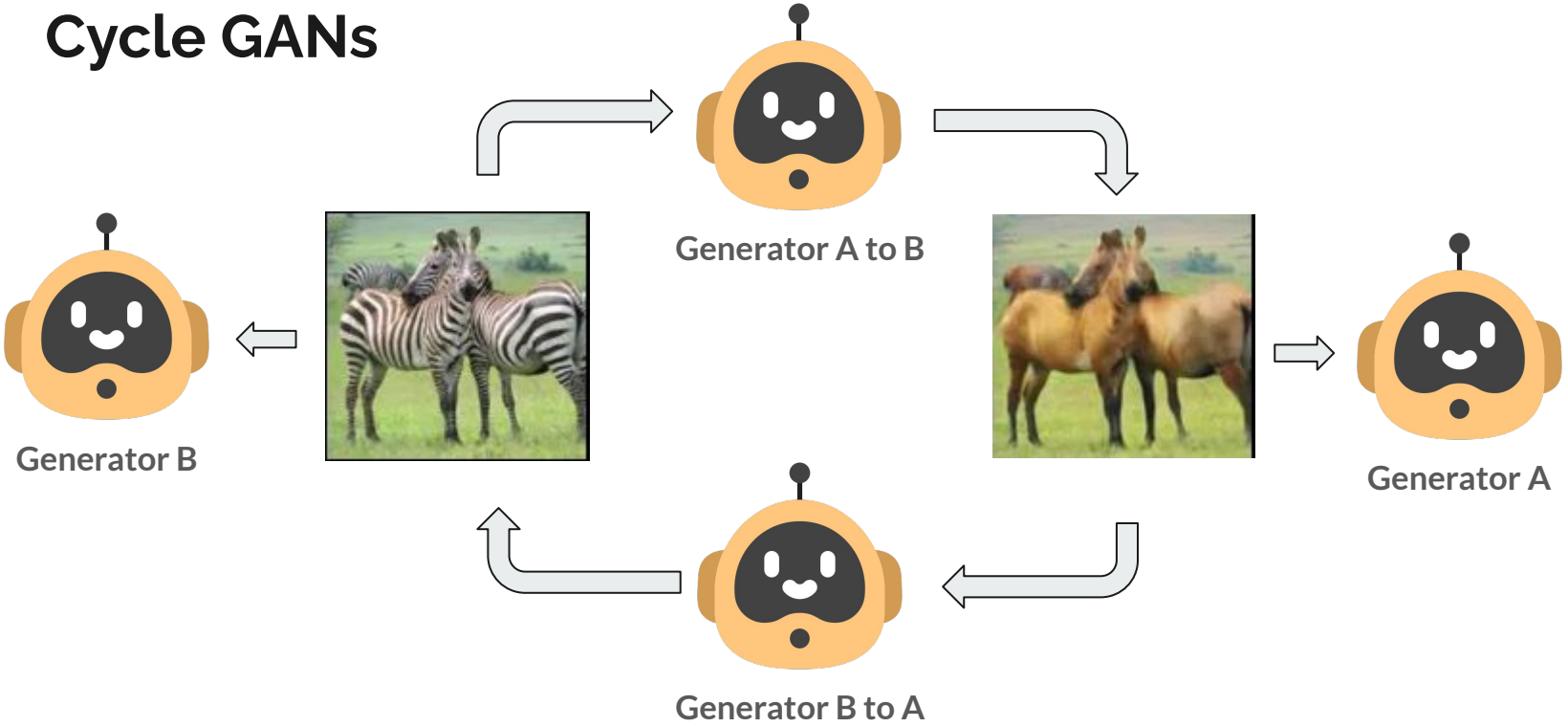
- The **Generator** and the **Discriminator** play a **game against each other**
- Sometimes the **generator** is able to **fool the discriminator**...
- And sometimes the **discriminator** can **unveil the generator**
- They both get better at their job, and **eventually they converge**

Generative Adversarial Networks (GANs)



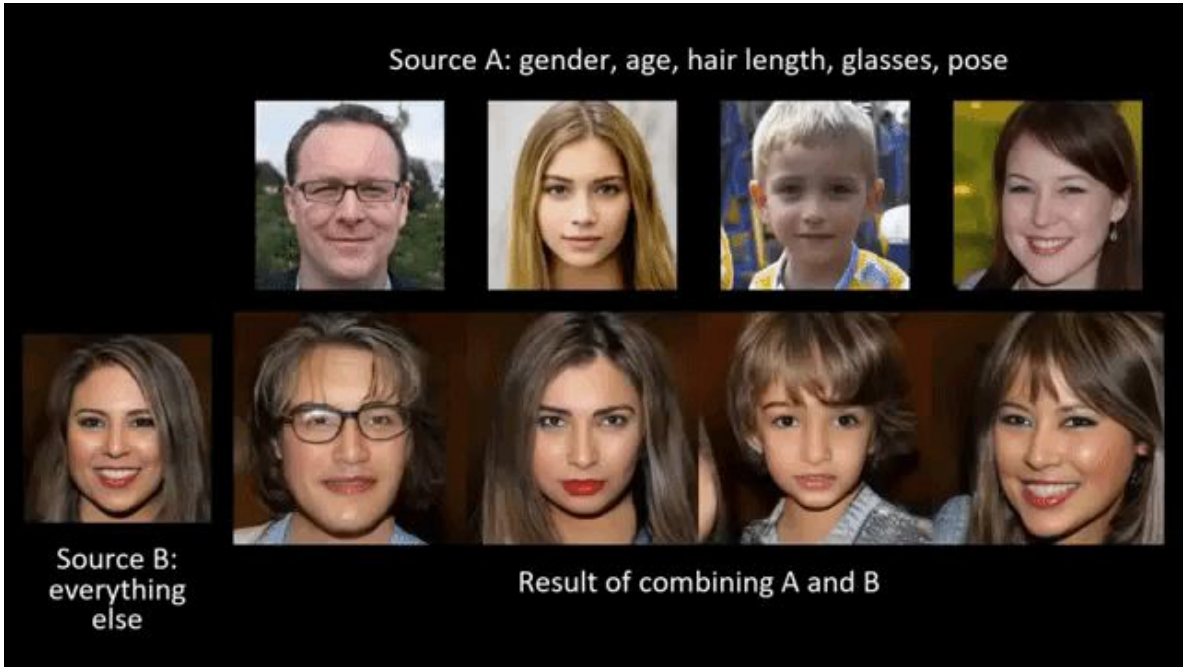


Cycle GANs

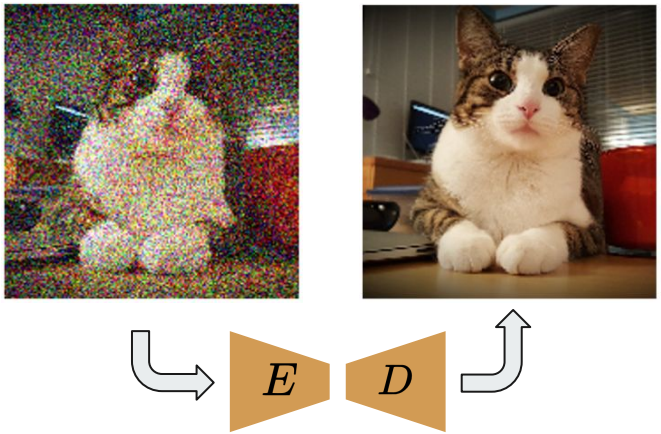


Cycle GANs - Style Transfer

StyleGAN

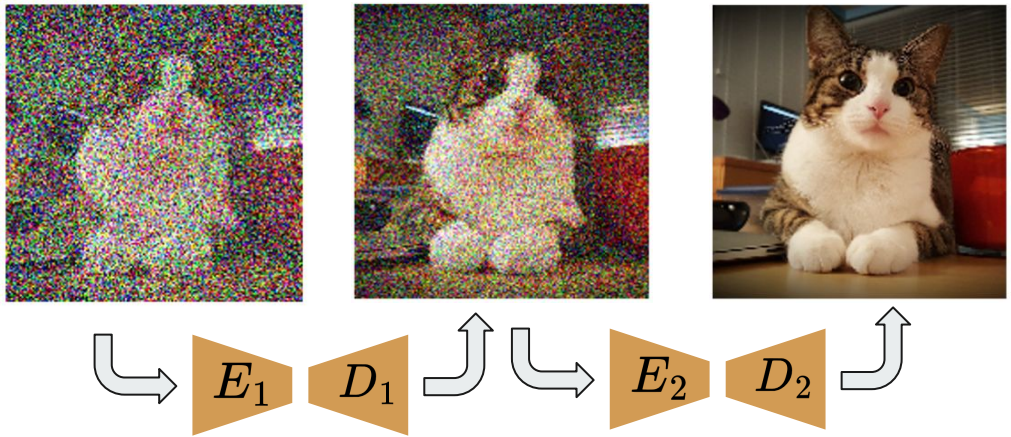

Source A: gender, age, hair length, glasses, pose
Source B: everything else
Result of combining A and B

Diffusion Models - idea



- The idea of reconstruction (from AEs) can be translated to noise reconstruction
- We can teach a network how to denoise the image

Diffusion Models - idea



- The idea of reconstruction (from AEs) can be translated to noise reconstruction
- We can teach a network how to denoise the image
- Even with multiple levels of noise

Diffusion Models - idea

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x_t}; \sqrt{1-\beta_t}\mathbf{x_{t-1}}, \beta_t\mathbf{I})$$

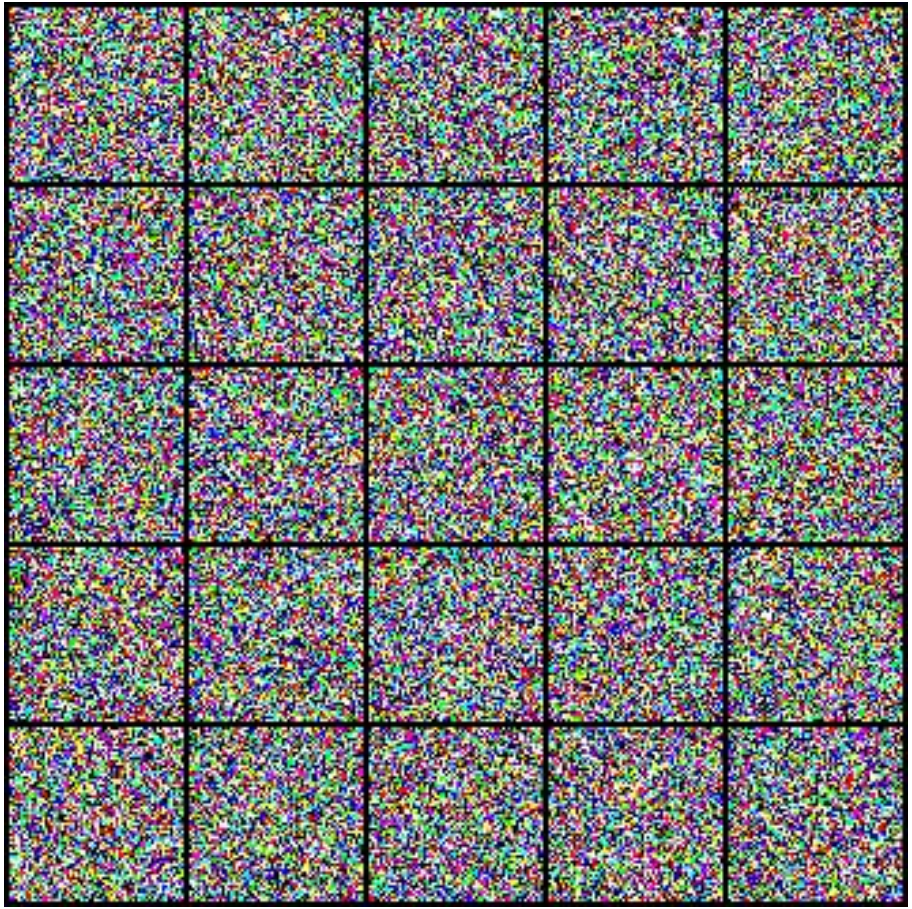
Data

Noise

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \sigma_t^2\mathbf{I})$$

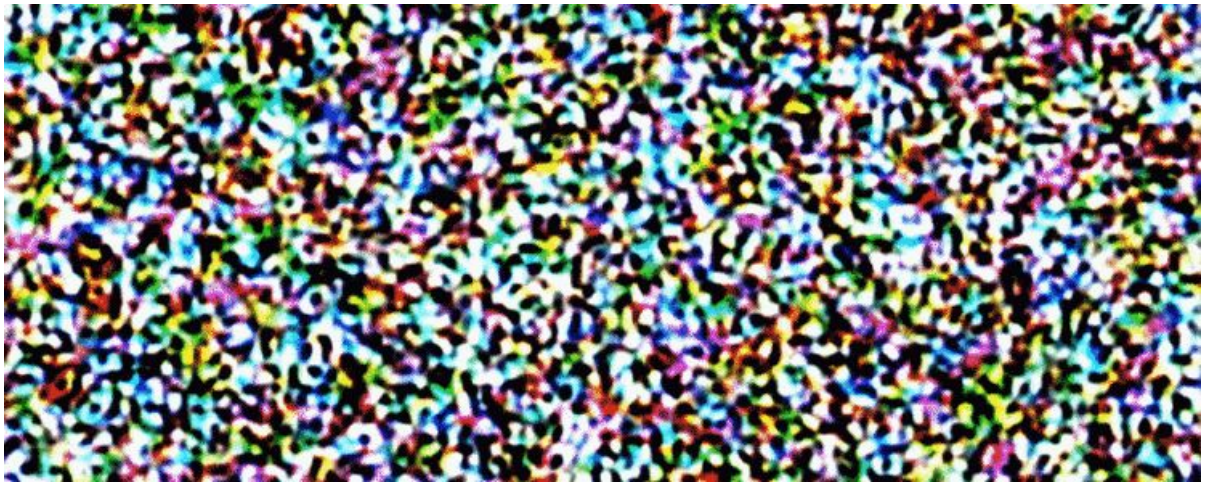
Eventually, the model will learn how to **generate images starting from random noise**!

Diffusion Models



Diffusion Models

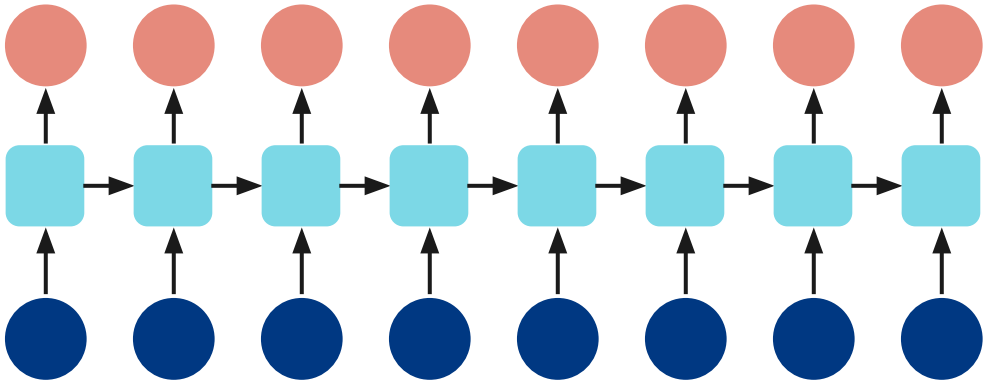
Diffusion Models



Attention Is All You Need!
The Transformer Architecture

To RNN or not to RNN?

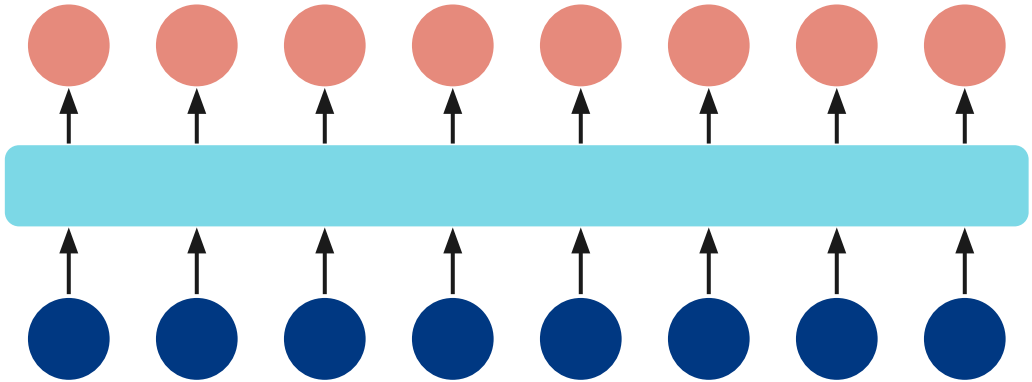
- Despite being specifically conceived for sequential data, RNNs have many issues



1. **Encoding bottleneck**
2. **Poor memory**
3. **No parallelization**

To RNN or not to RNN?

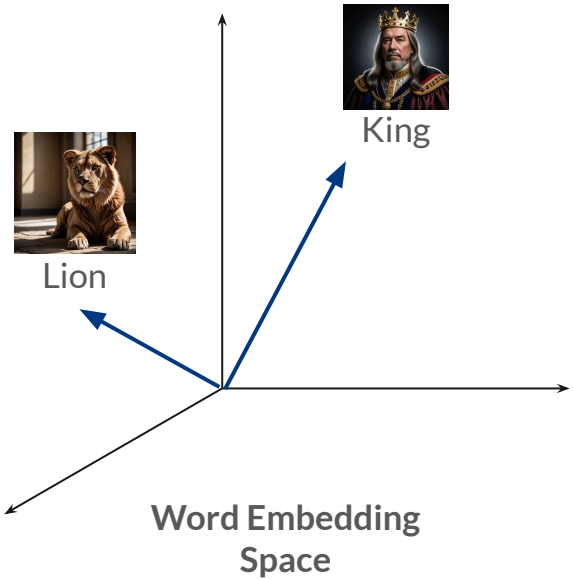
- Despite being specifically conceived for sequential data, RNNs have many issues



1. **Encoding bottleneck**
2. **Poor memory**
3. **No parallelization**

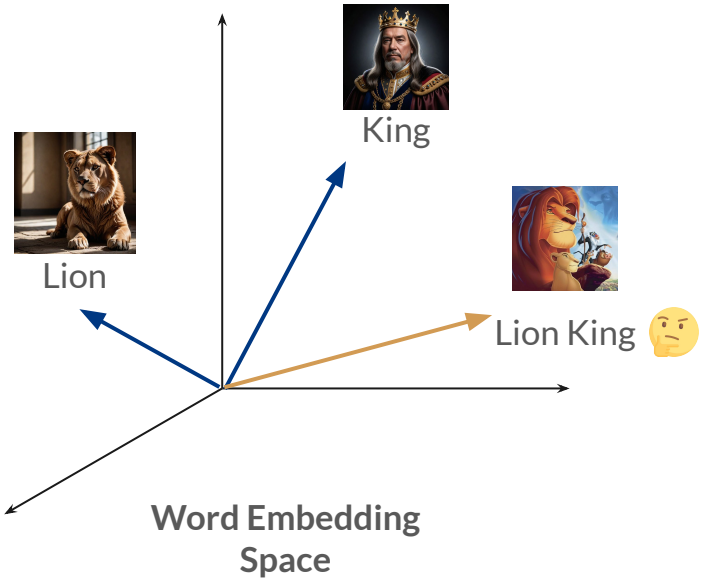
We can aggregate the entire input with a compact solution!

Rethinking the work embeddings



- Each token (word) is encoded using a specific vector
- Two vector may be more or less close to each other

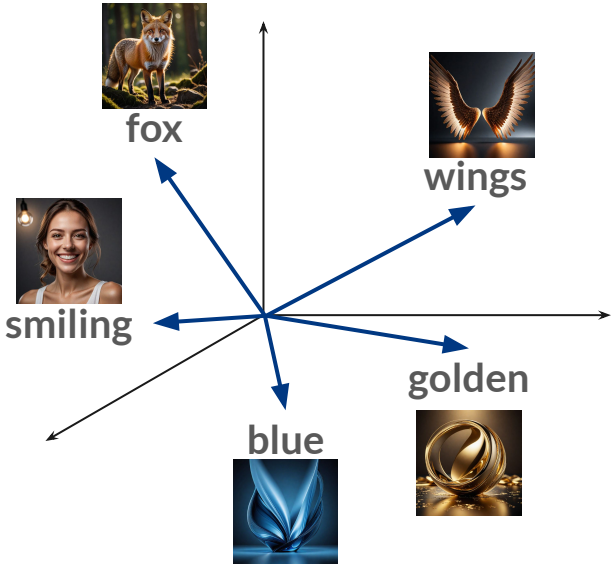
Rethinking the work embeddings



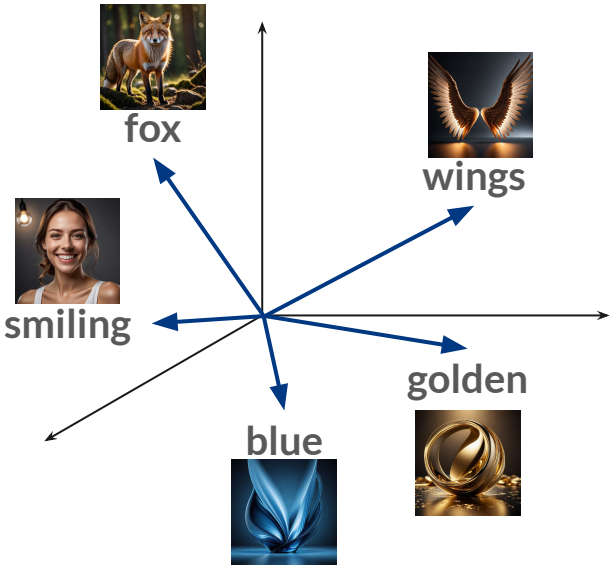
- Each token (word) is encoded using a specific vector
- Two vector may be more or less close to each other
- But context may add a lot of information to a single word, i.e., to an embedded vector!
- Encoding these relationships with would be a great solution… but how?

Too many combinations...

"A smiling blue fox with golden wings"

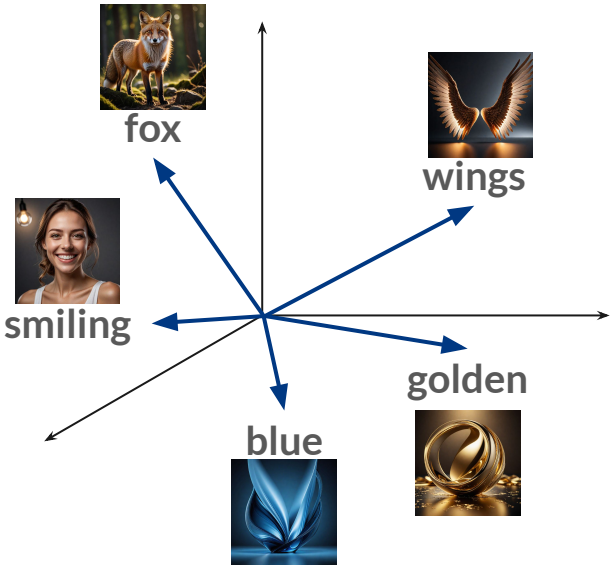


Too many combinations...



“A smiling blue fox with golden wings”

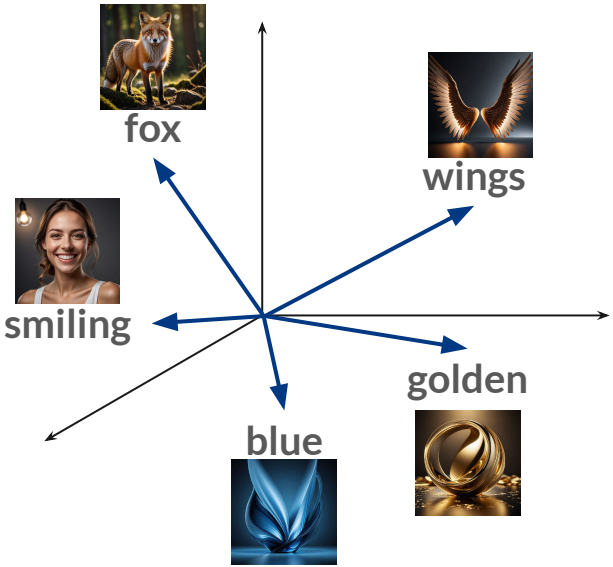
Too many combinations…



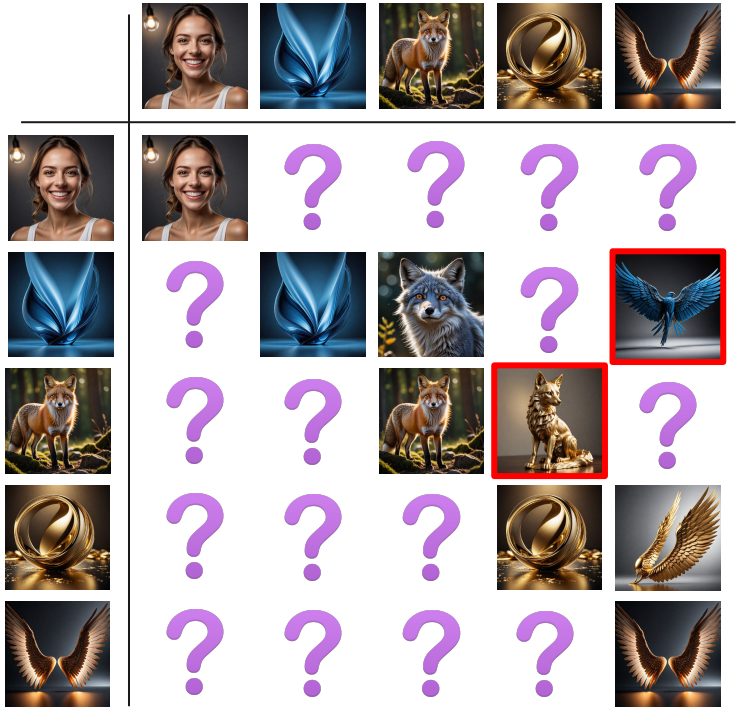
"A smiling blue fox with golden wings"

- Some relations are **useful**…

Too many combinations…



"A smiling blue fox with golden wings"



- Some relations are **useful**…
- But some are just **nonsense in this context**!

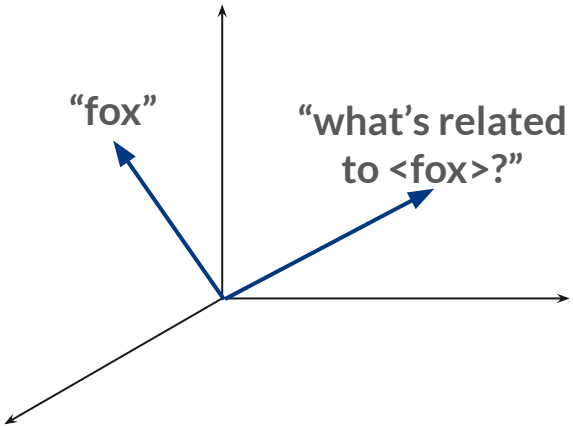
Relations

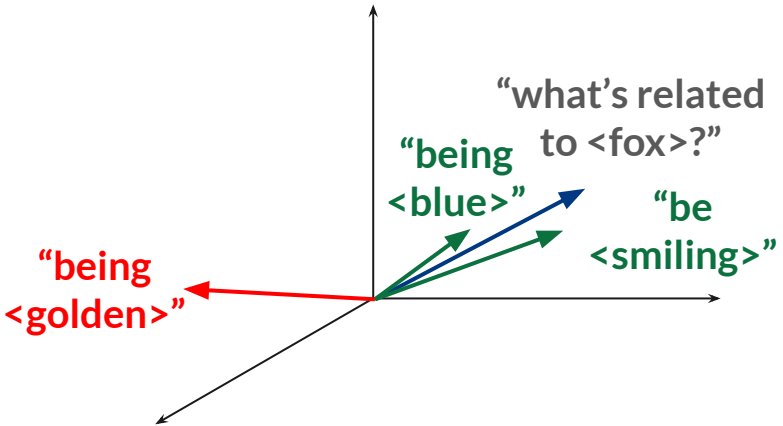
Useless association

Useful association

- Within the context of this sentence, there are some useful and some useless relationships
- The idea is that, for example, the “fox” is associated to the word “blue” but not to the word “gold”
- More in general we can ask the question: “what’s related to the fox?”

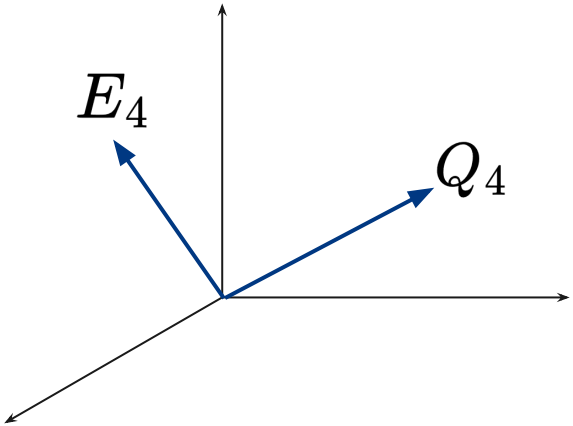
Relations

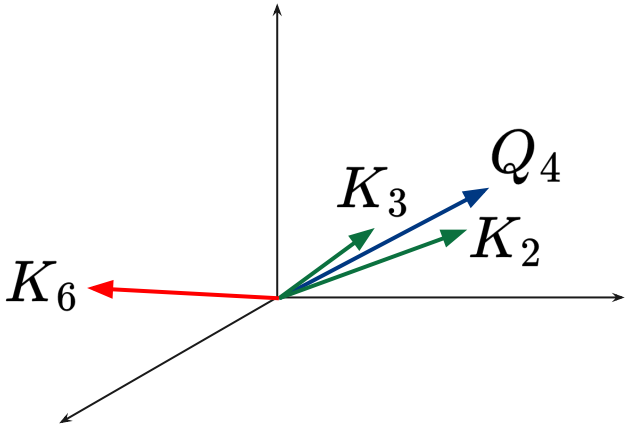




“A smiling blue fox with golden wings”

Keys and Queries

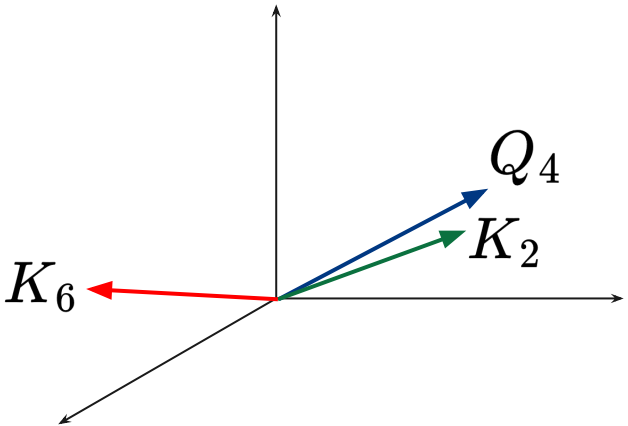




$$Q = E \cdot W_Q$$

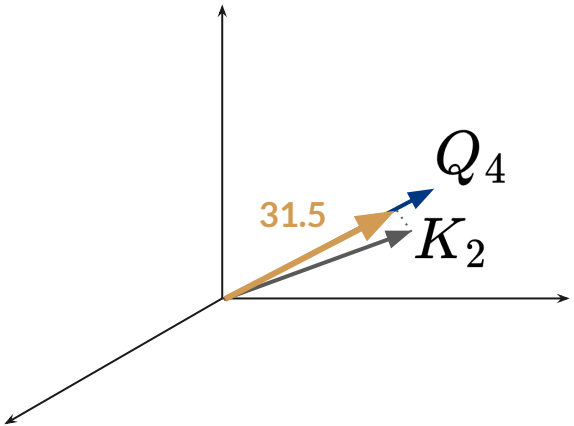
$$K = E \cdot W_K$$

Attention Mechanism



$$\frac{(Q \cdot K^T)}{\sqrt{d_k}}$$

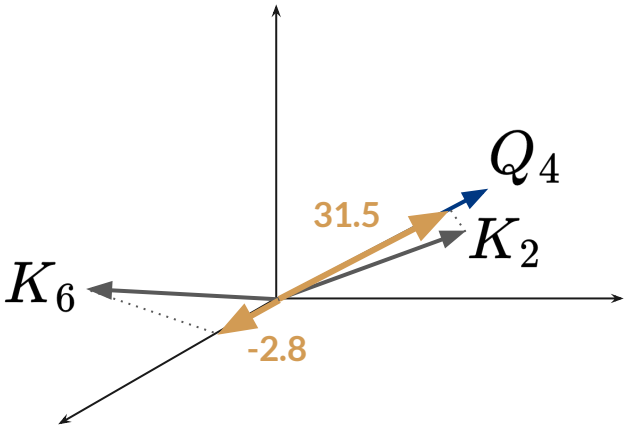
Attention Mechanism



$$\frac{(Q \cdot K^T)}{\sqrt{d_k}}$$

- Keys closer to the Query should be taken into consideration more than keys which are far from the query

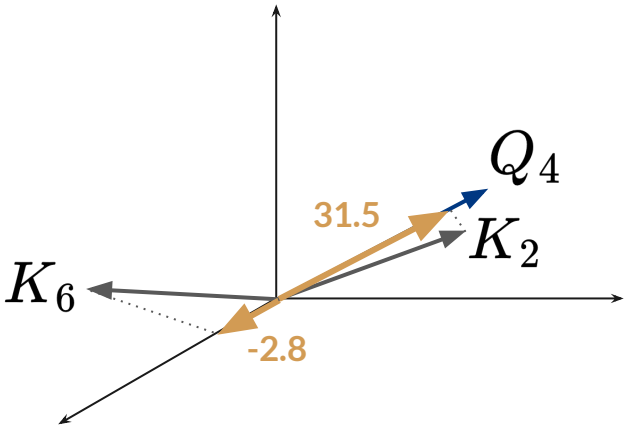
Attention Mechanism



$$\frac{(Q \cdot K^T)}{\sqrt{d_k}}$$

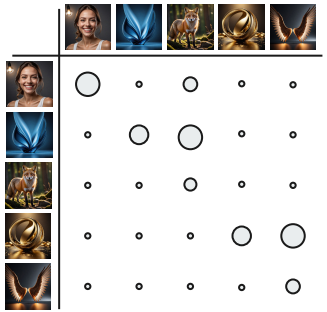
- Keys closer to the Query should be taken into consideration more than keys which are far from the query

Attention Mechanism

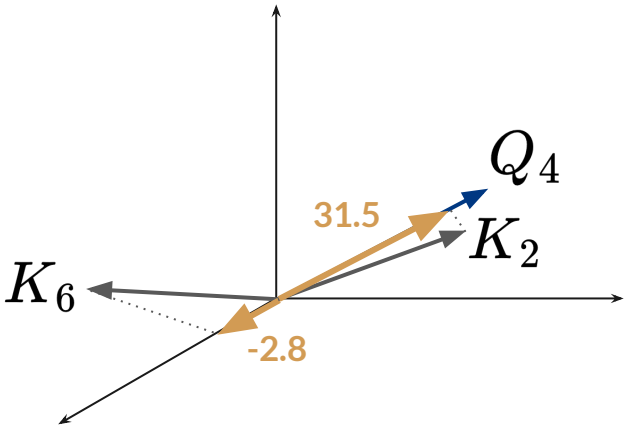


$$\text{softmax}\frac{(\mathrm{Q}\cdot\mathrm{K}^{\mathrm{T}})}{\sqrt{\mathrm{d_k}}}$$

- Keys closer to the Query should be taken into consideration more than keys which are far from the query
- We can normalize the values for obtaining a probability distribution for each query

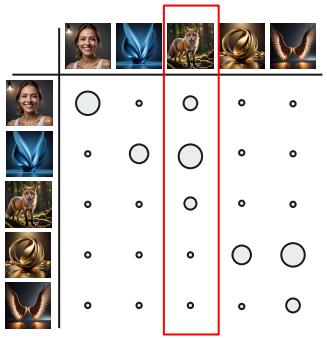


Attention Mechanism

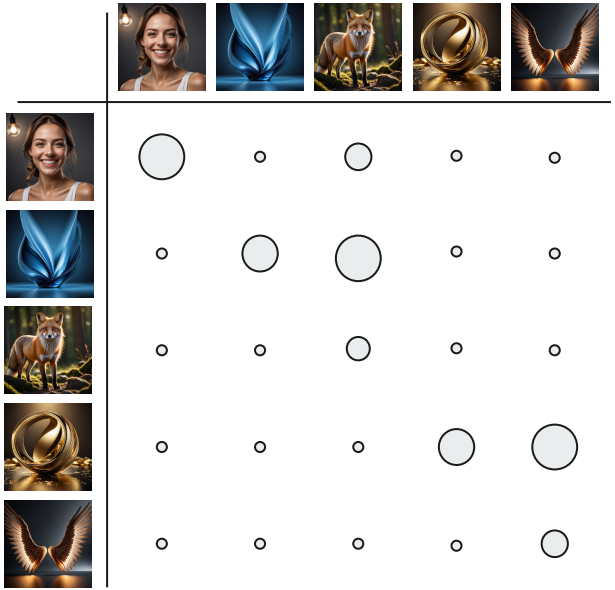


$$\text{softmax}\frac{(\mathrm{Q}\cdot\mathrm{K}^{\mathrm{T}})}{\sqrt{\mathrm{d_k}}}\cdot\mathrm{V}$$

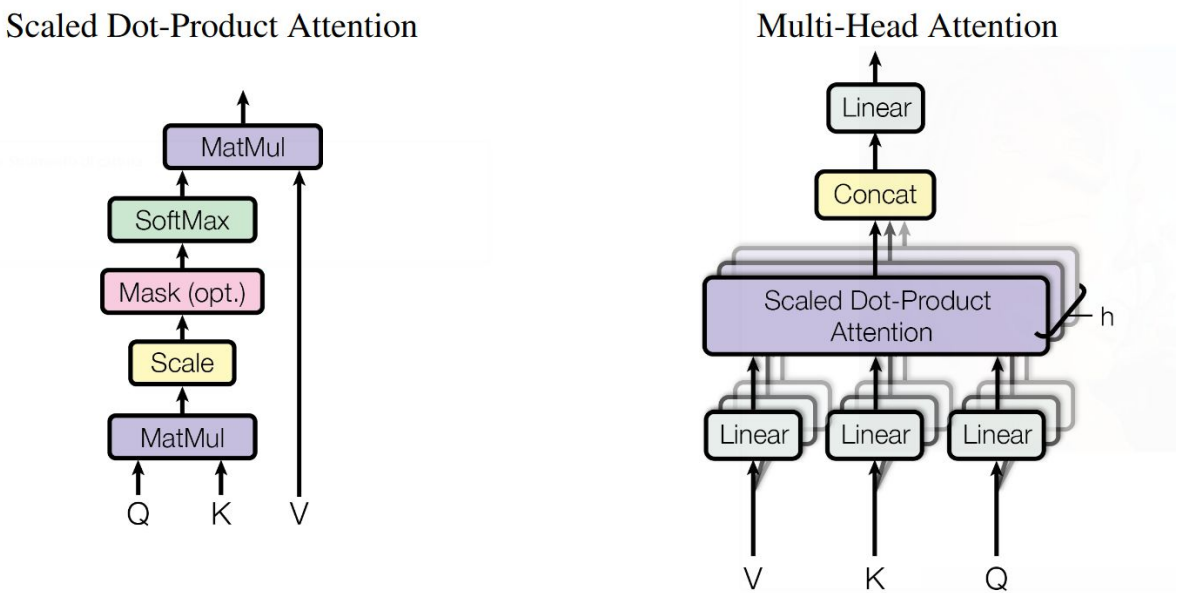
- Keys closer to the Query should be taken into consideration more than keys which are far from the query
- We can normalize the values for obtaining a probability distribution for each query
- Each weighted contribution is multiplied for the respective **value**



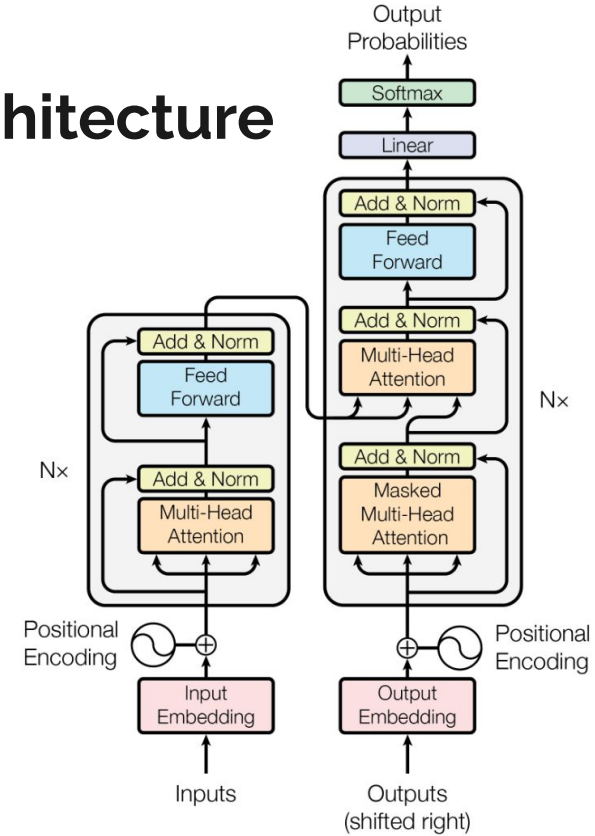
Attention Mechanism



Attention is all you need!

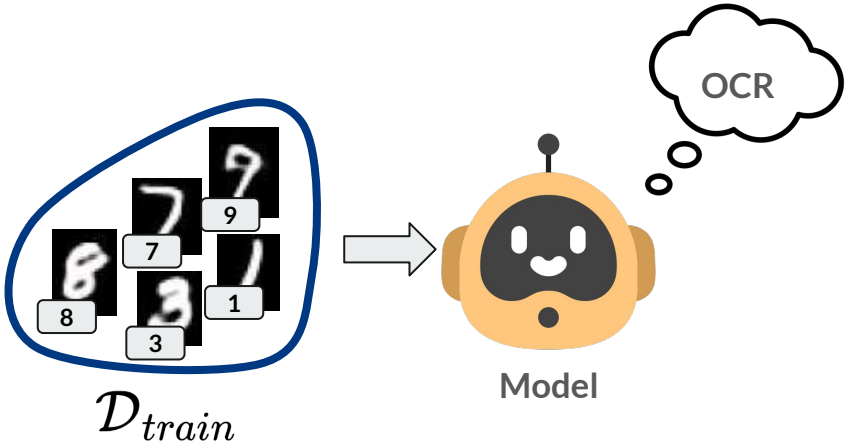


Transformer Architecture

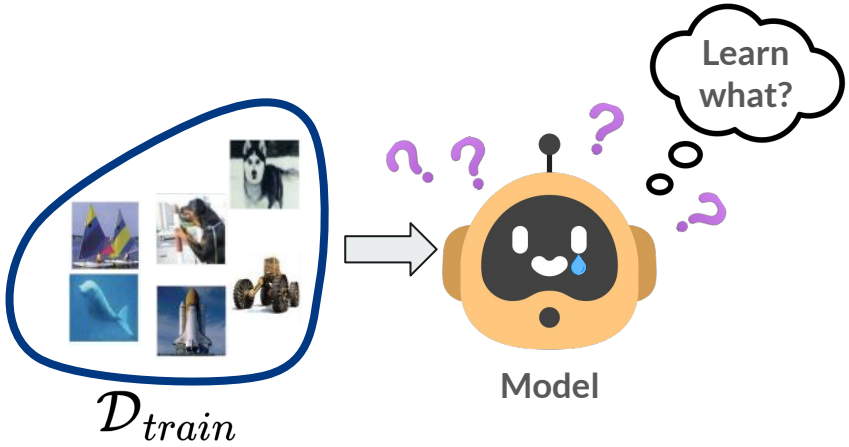


Foundation Models: From Self-Supervision to Multimodality

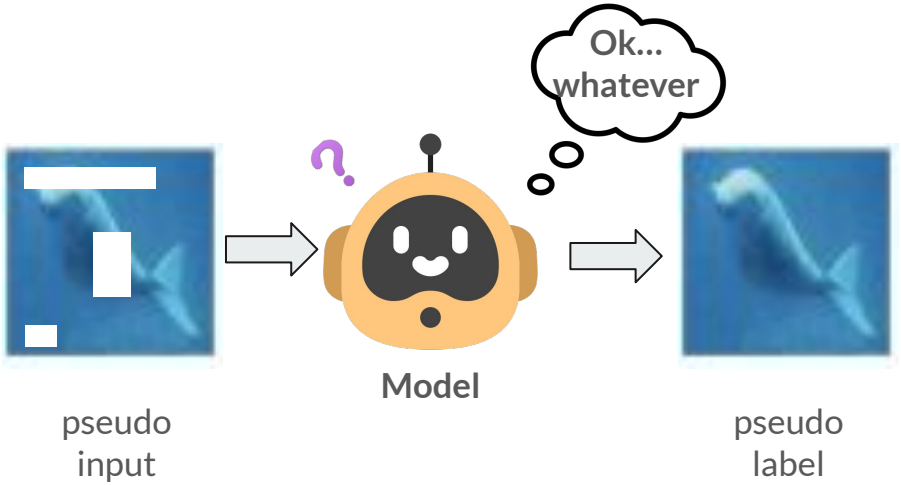
Supervised Learning



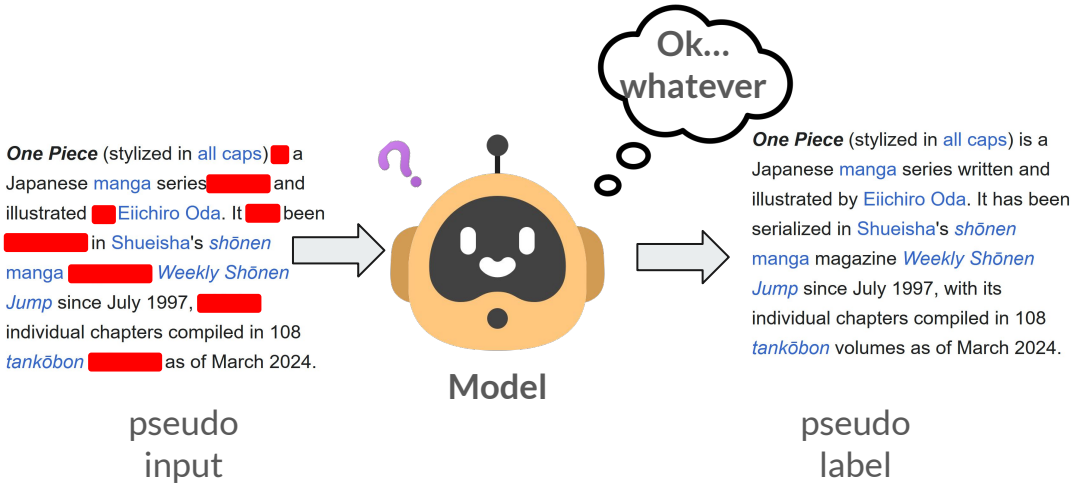
Supervised Learning



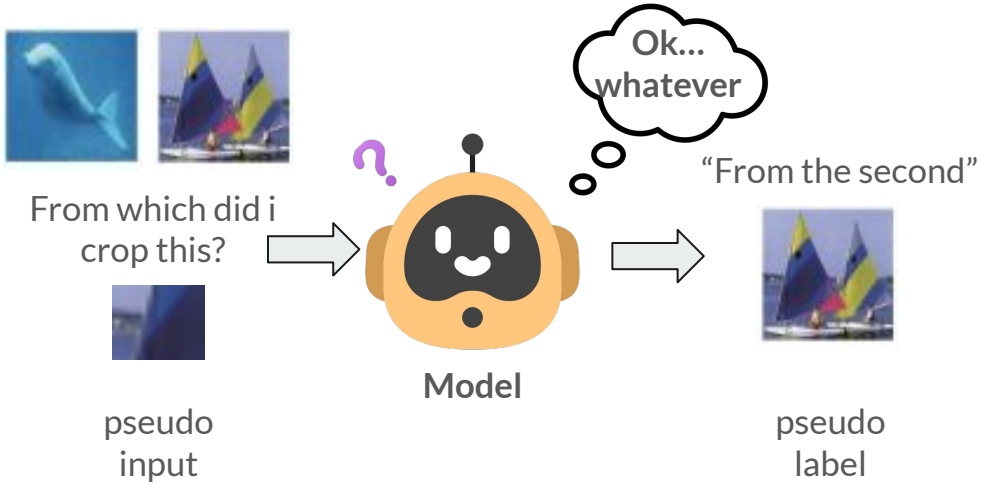
Supervised Learning - Reconstruct the input



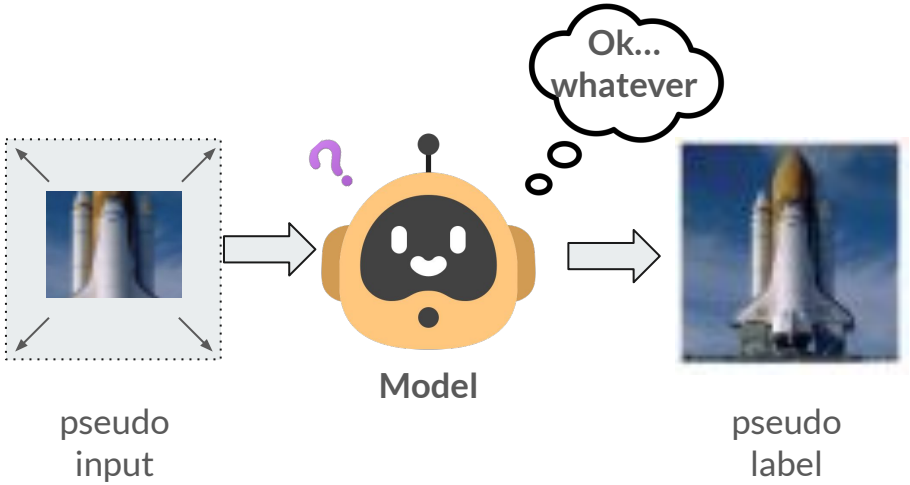
Supervised Learning - Reconstruct the input



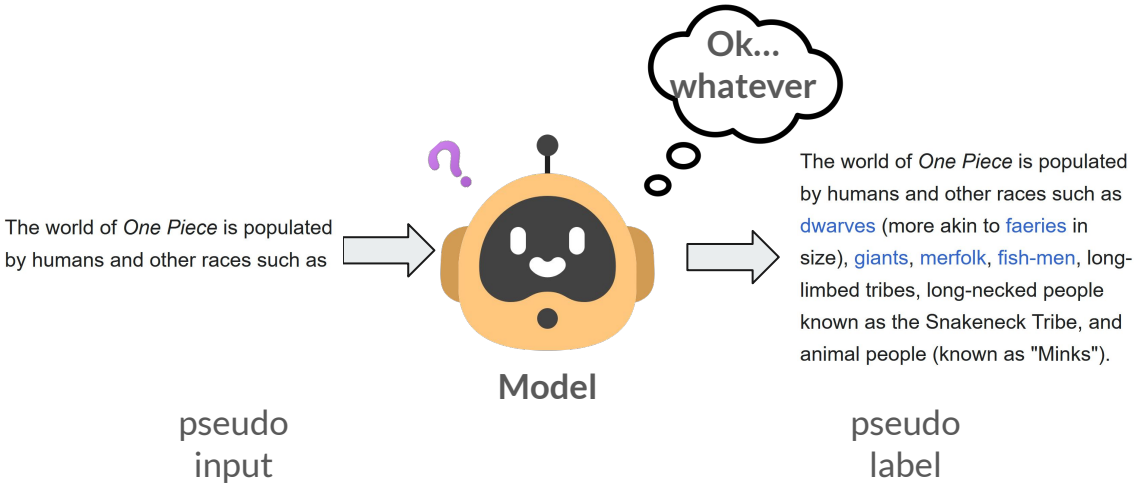
Supervised Learning - Reconstruct the input



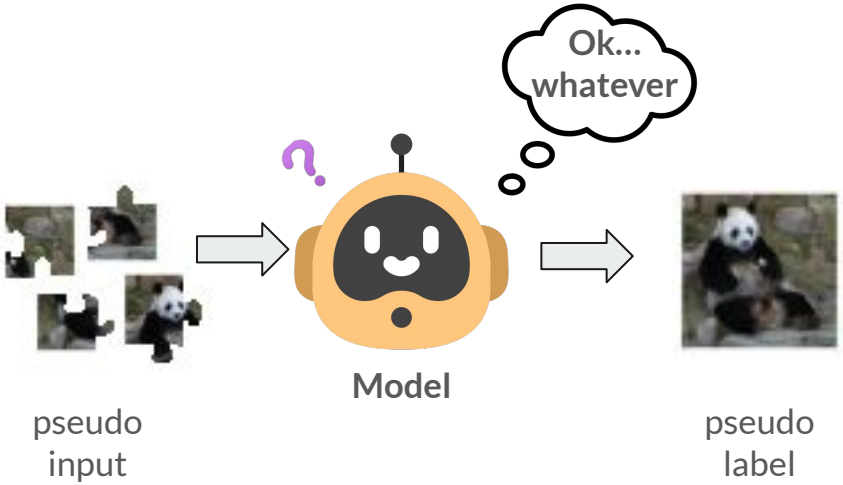
Supervised Learning - Reconstruct the input



Supervised Learning - Reconstruct the input



Supervised Learning - Reconstruct the input



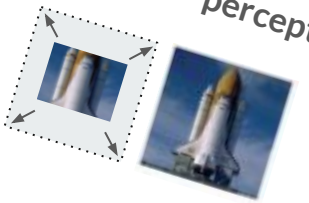
Self-Supervision

“The Dark Matter of Intelligence”

Understanding consistency

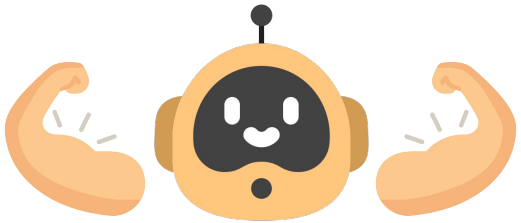
Discrimination capabilities

Low and High-level perception skills



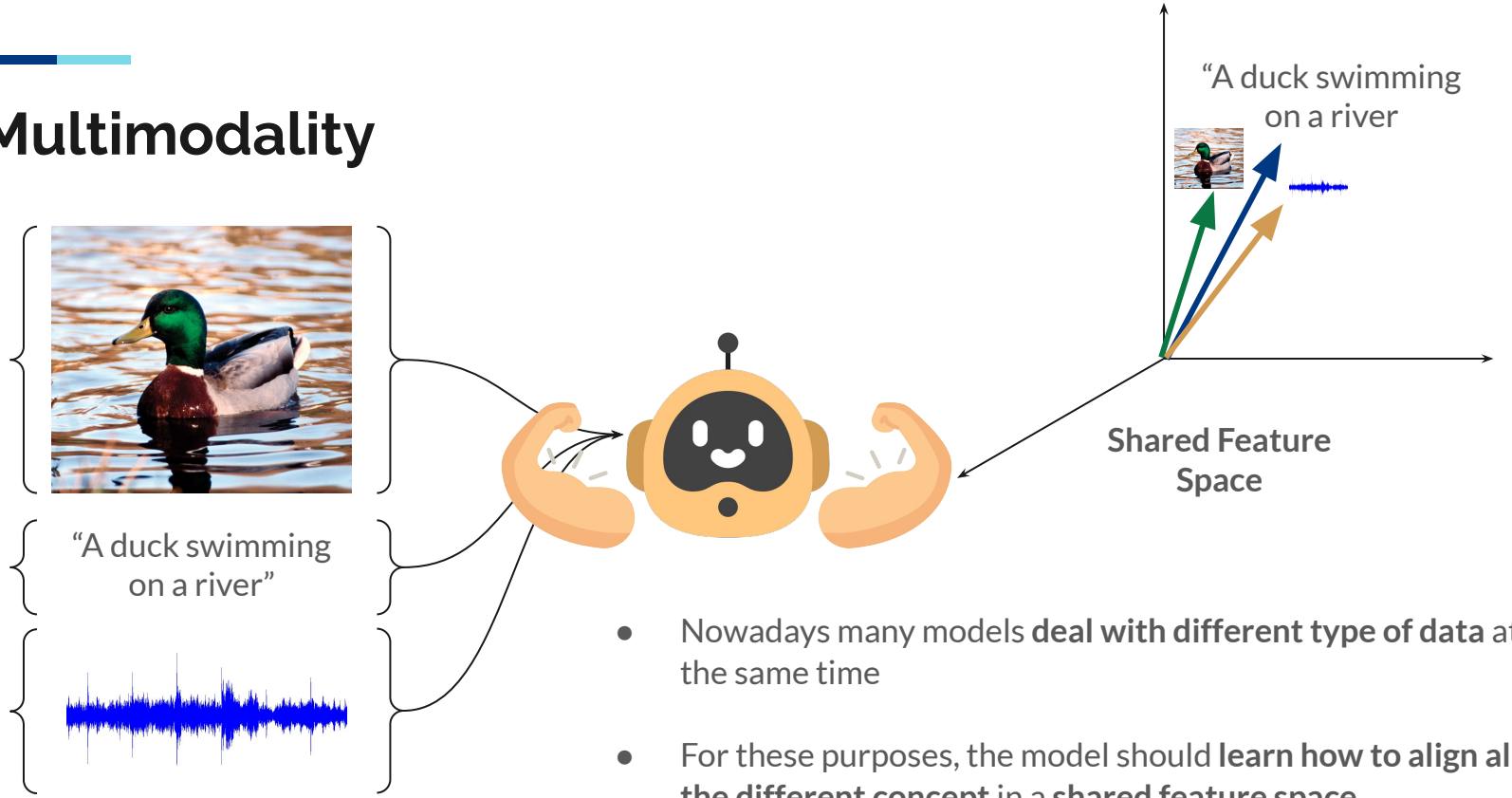

Foundation Models

Foundation Models



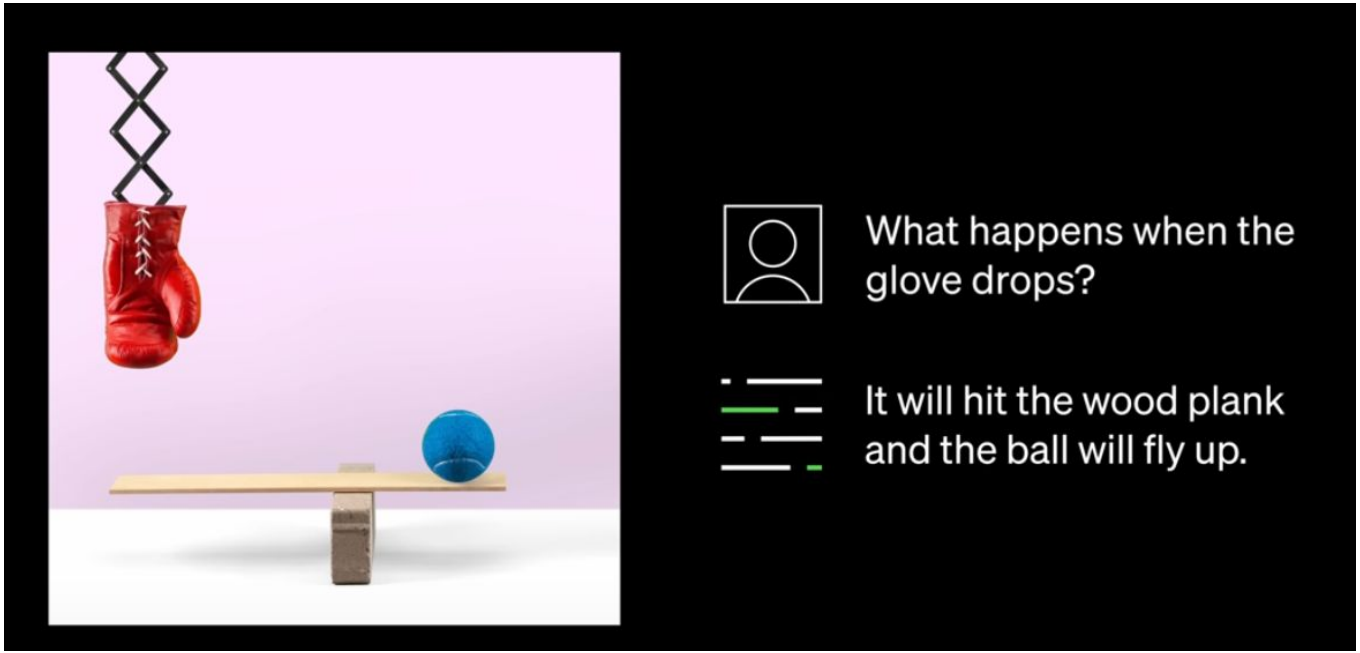
- **Very Large Models** (usually based on Transformer technologies)
- Trained with **huge set of data** (e.g., on the entire Wikipedia!)
- **General Purpose**: have not been trained explicitly for solving any specific “task”
- Massive use of **self-supervision**

Multimodality



- Nowadays many models **deal with different type of data** at the same time
- For these purposes, the model should **learn how to align all the different concept** in a **shared feature space**

Multimodality: GPT 4



Multimodality: Gemini

https://www.youtube.com/watch?v=UIZAiXYceBI