

# Attention and Transformers

CISC 7026: Introduction to Deep Learning

University of Macau

# Agenda

1. GNN Review
2. VAE Review and Coding
3. Attention
4. Keys and Queries
5. Self Attention
6. Positional Encoding
7. Transformers
8. Coding

# Agenda

1. GNN Review
2. VAE Review and Coding
3. Attention
4. Keys and Queries
5. Self Attention
6. Positional Encoding
7. Transformers
8. Coding

Graph is a structure of nodes (vertices) and edges

$$G = (\mathbf{X}, \mathbf{E})$$

$$\mathbf{X} \in \mathbb{R}^{T \times d_x}$$

$$\mathbf{E} \in \mathcal{P}(\mathbb{Z}_T \times \mathbb{Z}_T)$$

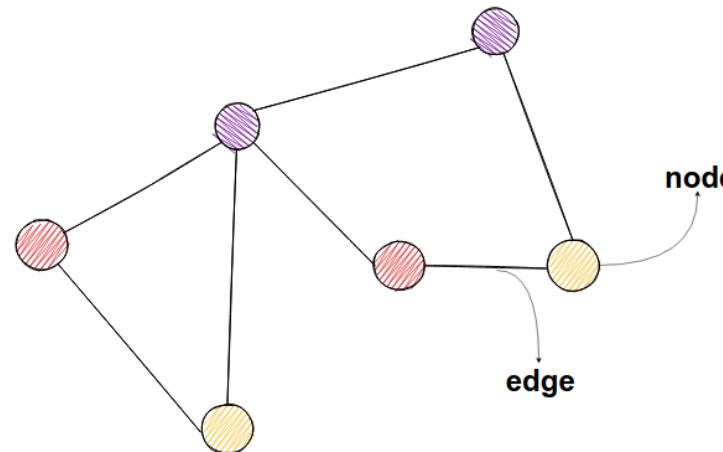
# GNN Review

Graph is a structure of nodes (vertices) and edges

$$G = (X, E)$$

$$X \in \mathbb{R}^{T \times d_x}$$

$$E \in \mathcal{P}(\mathbb{Z}_T \times \mathbb{Z}_T)$$



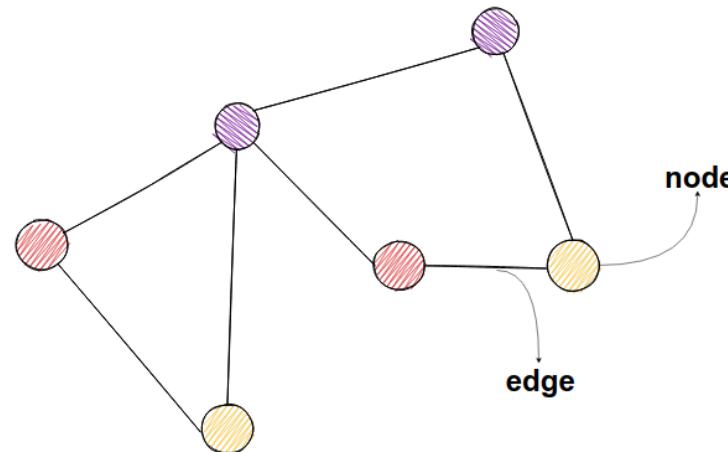
# GNN Review

Graph is a structure of nodes (vertices) and edges

$$G = (X, E)$$

$$X \in \mathbb{R}^{T \times d_x}$$

$$E \in \mathcal{P}(\mathbb{Z}_T \times \mathbb{Z}_T)$$



A **node** is a vector of information

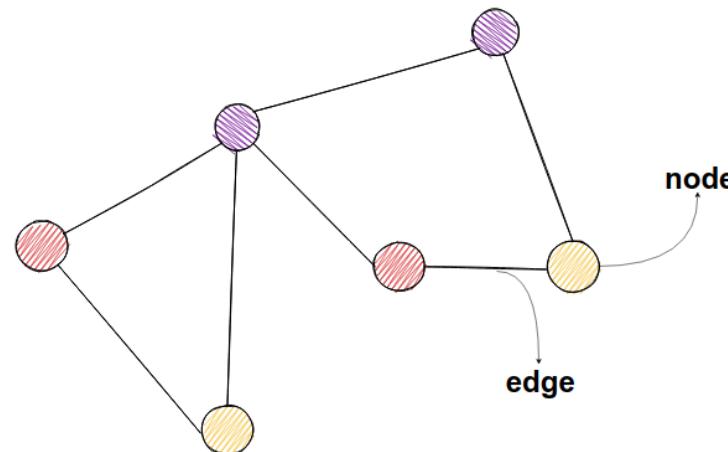
# GNN Review

Graph is a structure of nodes (vertices) and edges

$$G = (X, E)$$

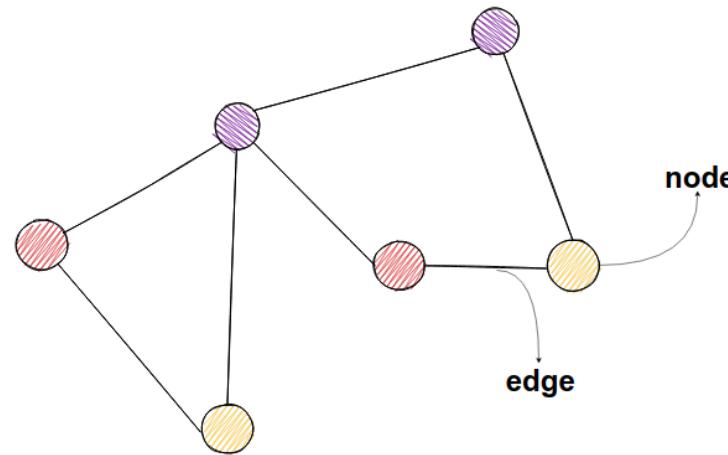
$$X \in \mathbb{R}^{T \times d_x}$$

$$E \in \mathcal{P}(\mathbb{Z}_T \times \mathbb{Z}_T)$$



A **node** is a vector of information

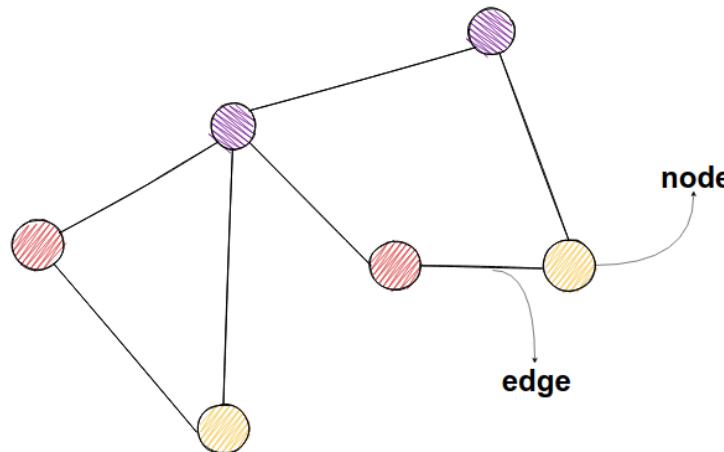
An **edge** connects two nodes



A **node** is a vector of information

An **edge** connects two nodes

If we connect nodes  $i$  and  $j$  with edge  $(i, j)$ , then  $i$  and  $j$  are **neighbors**



A **node** is a vector of information

An **edge** connects two nodes

If we connect nodes  $i$  and  $j$  with edge  $(i, j)$ , then  $i$  and  $j$  are **neighbors**

The **neighborhood**  $N(j)$  contains all neighbors of node  $j$

Prof. Li introduced the **graph convolution layer**

Prof. Li introduced the **graph convolution layer**

For a node  $j$ , the graph convolution layer is:

Prof. Li introduced the **graph convolution layer**

For a node  $j$ , the graph convolution layer is:

$$f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_j = \sigma \left( \boldsymbol{\theta}_1^\top \bar{\mathbf{x}}_j + \sum_{i \in N(j)} \boldsymbol{\theta}_2^\top \mathbf{x}_i \right)$$

Prof. Li introduced the **graph convolution layer**

For a node  $j$ , the graph convolution layer is:

$$f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_j = \sigma \left( \boldsymbol{\theta}_1^\top \bar{\mathbf{x}}_j + \sum_{i \in N(j)} \boldsymbol{\theta}_2^\top \mathbf{x}_i \right)$$

Combine information from current node  $\mathbf{x}_j$  with neighbors  $\mathbf{x}_i$

Prof. Li introduced the **graph convolution layer**

For a node  $j$ , the graph convolution layer is:

$$f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_j = \sigma \left( \boldsymbol{\theta}_1^\top \bar{\mathbf{x}}_j + \sum_{i \in N(j)} \boldsymbol{\theta}_2^\top \mathbf{x}_i \right)$$

Combine information from current node  $\mathbf{x}_j$  with neighbors  $\mathbf{x}_i$

This is just one node, we use this graph layer for all nodes in the graph

Graph convolution over all nodes in the graph

Graph convolution over all nodes in the graph

$$f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta}) = \begin{bmatrix} f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_1 \\ f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_2 \\ \vdots \\ f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_T \end{bmatrix} = \begin{bmatrix} \sigma\left(\boldsymbol{\theta}_1^\top \bar{\mathbf{x}}_1 + \sum_{i \in N(1)} \boldsymbol{\theta}_2^\top \mathbf{x}_i\right) \\ \sigma\left(\boldsymbol{\theta}_1^\top \bar{\mathbf{x}}_2 + \sum_{i \in N(2)} \boldsymbol{\theta}_2^\top \mathbf{x}_i\right) \\ \vdots \\ \sigma\left(\boldsymbol{\theta}_1^\top \bar{\mathbf{x}}_T + \sum_{i \in N(T)} \boldsymbol{\theta}_2^\top \mathbf{x}_i\right) \end{bmatrix}$$

Graph convolution over all nodes in the graph

$$f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta}) = \begin{bmatrix} f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_1 \\ f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_2 \\ \vdots \\ f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_T \end{bmatrix} = \begin{bmatrix} \sigma\left(\boldsymbol{\theta}_1^\top \bar{\mathbf{x}}_1 + \sum_{i \in N(1)} \boldsymbol{\theta}_2^\top \mathbf{x}_i\right) \\ \sigma\left(\boldsymbol{\theta}_1^\top \bar{\mathbf{x}}_2 + \sum_{i \in N(2)} \boldsymbol{\theta}_2^\top \mathbf{x}_i\right) \\ \vdots \\ \sigma\left(\boldsymbol{\theta}_1^\top \bar{\mathbf{x}}_T + \sum_{i \in N(T)} \boldsymbol{\theta}_2^\top \mathbf{x}_i\right) \end{bmatrix}$$

How does this compare to regular convolution (images, sound, etc)?

## Standard convolution

$$\begin{bmatrix} \sigma(\theta_1^\top \bar{x}_1 + \sum_{i=1}^k \theta_2^\top x_i) \\ \sigma(\theta_1^\top \bar{x}_2 + \sum_{i=2}^{k+1} \theta_2^\top x_i) \\ \vdots \\ \sigma(\theta_1^\top \bar{x}_T + \sum_{i=T-k}^T \theta_2^\top x_i) \end{bmatrix}$$

## Standard convolution

$$\begin{bmatrix} \sigma\left(\theta_1^\top \bar{x}_1 + \sum_{i=1}^k \theta_2^\top x_i\right) \\ \sigma\left(\theta_1^\top \bar{x}_2 + \sum_{i=2}^{k+1} \theta_2^\top x_i\right) \\ \vdots \\ \sigma\left(\theta_1^\top \bar{x}_T + \sum_{i=T-k}^T \theta_2^\top x_i\right) \end{bmatrix}$$

## Graph convolution

$$\begin{bmatrix} \sigma\left(\theta_1^\top \bar{x}_1 + \sum_{i \in N(1)} \theta_2^\top x_i\right) \\ \sigma\left(\theta_1^\top \bar{x}_2 + \sum_{i \in N(2)} \theta_2^\top x_i\right) \\ \vdots \\ \sigma\left(\theta_1^\top \bar{x}_T + \sum_{i \in N(T)} \theta_2^\top x_i\right) \end{bmatrix}$$

## Standard convolution

$$\begin{bmatrix} \sigma\left(\theta_1^\top \bar{x}_1 + \sum_{i=1}^k \theta_2^\top x_i\right) \\ \sigma\left(\theta_1^\top \bar{x}_2 + \sum_{i=2}^{k+1} \theta_2^\top x_i\right) \\ \vdots \\ \sigma\left(\theta_1^\top \bar{x}_T + \sum_{i=T-k}^T \theta_2^\top x_i\right) \end{bmatrix}$$

## Graph convolution

$$\begin{bmatrix} \sigma\left(\theta_1^\top \bar{x}_1 + \sum_{i \in N(1)} \theta_2^\top x_i\right) \\ \sigma\left(\theta_1^\top \bar{x}_2 + \sum_{i \in N(2)} \theta_2^\top x_i\right) \\ \vdots \\ \sigma\left(\theta_1^\top \bar{x}_T + \sum_{i \in N(T)} \theta_2^\top x_i\right) \end{bmatrix}$$

**Question:** What is the output size of standard convolution?

Standard convolution

$$\begin{bmatrix} \sigma\left(\theta_1^\top \bar{x}_1 + \sum_{i=1}^k \theta_2^\top x_i\right) \\ \sigma\left(\theta_1^\top \bar{x}_2 + \sum_{i=2}^{k+1} \theta_2^\top x_i\right) \\ \vdots \\ \sigma\left(\theta_1^\top \bar{x}_T + \sum_{i=T-k}^T \theta_2^\top x_i\right) \end{bmatrix}$$

Graph convolution

$$\begin{bmatrix} \sigma\left(\theta_1^\top \bar{x}_1 + \sum_{i \in N(1)} \theta_2^\top x_i\right) \\ \sigma\left(\theta_1^\top \bar{x}_2 + \sum_{i \in N(2)} \theta_2^\top x_i\right) \\ \vdots \\ \sigma\left(\theta_1^\top \bar{x}_T + \sum_{i \in N(T)} \theta_2^\top x_i\right) \end{bmatrix}$$

**Question:** What is the output size of standard convolution?

**Answer:**  $T - k - 1 \times d_h$

## Standard convolution

$$\begin{bmatrix} \sigma\left(\theta_1^\top \bar{x}_1 + \sum_{i=1}^k \theta_2^\top x_i\right) \\ \sigma\left(\theta_1^\top \bar{x}_2 + \sum_{i=2}^{k+1} \theta_2^\top x_i\right) \\ \vdots \\ \sigma\left(\theta_1^\top \bar{x}_T + \sum_{i=T-k}^T \theta_2^\top x_i\right) \end{bmatrix}$$

## Graph convolution

$$\begin{bmatrix} \sigma\left(\theta_1^\top \bar{x}_1 + \sum_{i \in N(1)} \theta_2^\top x_i\right) \\ \sigma\left(\theta_1^\top \bar{x}_2 + \sum_{i \in N(2)} \theta_2^\top x_i\right) \\ \vdots \\ \sigma\left(\theta_1^\top \bar{x}_T + \sum_{i \in N(T)} \theta_2^\top x_i\right) \end{bmatrix}$$

**Question:** What is the output size of graph convolution?

Standard convolution

$$\begin{bmatrix} \sigma\left(\theta_1^\top \bar{x}_1 + \sum_{i=1}^k \theta_2^\top x_i\right) \\ \sigma\left(\theta_1^\top \bar{x}_2 + \sum_{i=2}^{k+1} \theta_2^\top x_i\right) \\ \vdots \\ \sigma\left(\theta_1^\top \bar{x}_T + \sum_{i=T-k}^T \theta_2^\top x_i\right) \end{bmatrix}$$

Graph convolution

$$\begin{bmatrix} \sigma\left(\theta_1^\top \bar{x}_1 + \sum_{i \in N(1)} \theta_2^\top x_i\right) \\ \sigma\left(\theta_1^\top \bar{x}_2 + \sum_{i \in N(2)} \theta_2^\top x_i\right) \\ \vdots \\ \sigma\left(\theta_1^\top \bar{x}_T + \sum_{i \in N(T)} \theta_2^\top x_i\right) \end{bmatrix}$$

**Question:** What is the output size of graph convolution?

**Answer:**  $T \times d_h$

We can use pooling with graph convolutions too

$$\text{SumPool} \left( \begin{bmatrix} \sigma(\theta_1^\top \bar{x}_1 + \sum_{i \in N(1)} \theta_2^\top x_i) \\ \sigma(\theta_1^\top \bar{x}_2 + \sum_{i \in N(2)} \theta_2^\top x_i) \\ \vdots \\ \sigma(\theta_1^\top \bar{x}_T + \sum_{i \in N(T)} \theta_2^\top x_i) \end{bmatrix} \right) = \sigma \left( \theta_1^\top \bar{x}_1 + \sum_{i \in N(1)} \theta_2^\top x_i \right) + \sigma \left( \theta_1^\top \bar{x}_2 + \sum_{i \in N(2)} \theta_2^\top x_i \right) + \dots$$

Standard convolution has an ordering, graph convolution is  
**permutation invariant**

Standard convolution has an ordering, graph convolution is **permutation invariant**

$$\text{SumPool}(f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})) = \text{SumPool} \left( \begin{bmatrix} f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_1 \\ f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_2 \\ \vdots \\ f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_T \end{bmatrix} \right)$$
$$= \text{SumPool} \left( \begin{bmatrix} f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_2 \\ f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_T \\ \vdots \\ f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_1 \end{bmatrix} \right)$$

Standard convolution has an ordering, graph convolution is **permutation invariant**

$$\text{SumPool}(f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})) = \text{SumPool} \left( \begin{bmatrix} f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_1 \\ f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_2 \\ \vdots \\ f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_T \end{bmatrix} \right)$$
$$= \text{SumPool} \left( \begin{bmatrix} f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_2 \\ f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_T \\ \vdots \\ f(\mathbf{X}, \mathbf{E}, \boldsymbol{\theta})_1 \end{bmatrix} \right)$$

# Agenda

1. GNN Review
2. VAE Review and Coding
3. Attention
4. Keys and Queries
5. Self Attention
6. Positional Encoding
7. Transformers
8. Coding

# Agenda

1. GNN Review
2. VAE Review and Coding
3. Attention
4. Keys and Queries
5. Self Attention
6. Positional Encoding
7. Transformers
8. Coding

# VAE Review and Coding

Autoencoders are useful for compression and denoising

# VAE Review and Coding

Autoencoders are useful for compression and denoising

But we can also use them as **generative models**

# VAE Review and Coding

Autoencoders are useful for compression and denoising

But we can also use them as **generative models**

A generative model learns the structure of data

# VAE Review and Coding

Autoencoders are useful for compression and denoising

But we can also use them as **generative models**

A generative model learns the structure of data

Using this structure, it generates **new** data

# VAE Review and Coding

Autoencoders are useful for compression and denoising

But we can also use them as **generative models**

A generative model learns the structure of data

Using this structure, it generates **new** data

- Train on face dataset, generate **new** pictures

# VAE Review and Coding

Autoencoders are useful for compression and denoising

But we can also use them as **generative models**

A generative model learns the structure of data

Using this structure, it generates **new** data

- Train on face dataset, generate **new** pictures
- Train on book dataset, write a **new** book

# VAE Review and Coding

Autoencoders are useful for compression and denoising

But we can also use them as **generative models**

A generative model learns the structure of data

Using this structure, it generates **new** data

- Train on face dataset, generate **new** pictures
- Train on book dataset, write a **new** book
- Train on protein dataset, create **new** proteins

# VAE Review and Coding

Autoencoders are useful for compression and denoising

But we can also use them as **generative models**

A generative model learns the structure of data

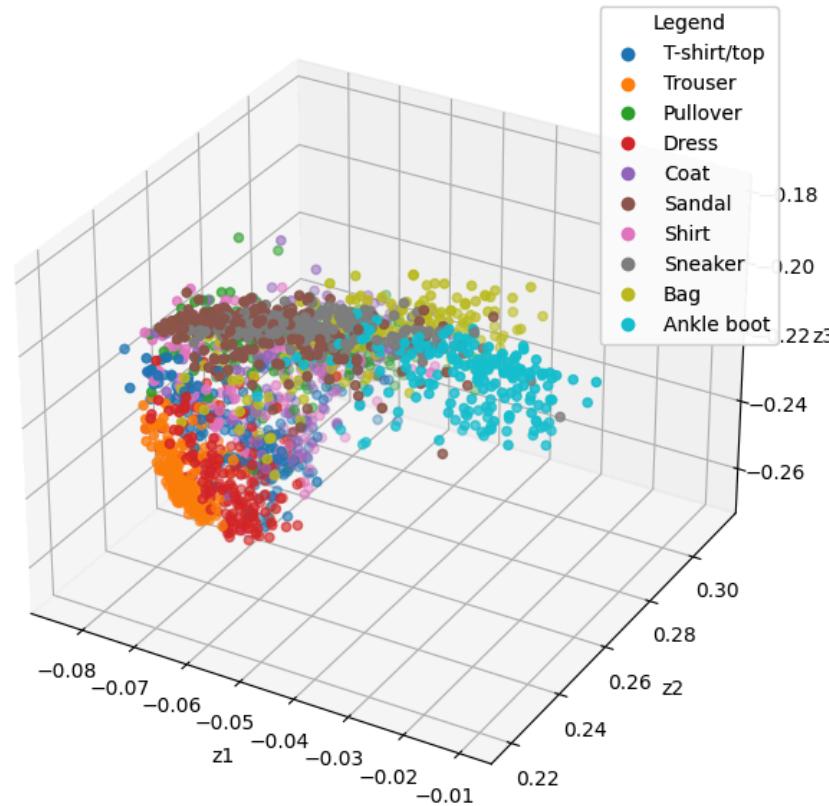
Using this structure, it generates **new** data

- Train on face dataset, generate **new** pictures
- Train on book dataset, write a **new** book
- Train on protein dataset, create **new** proteins

How does this work?

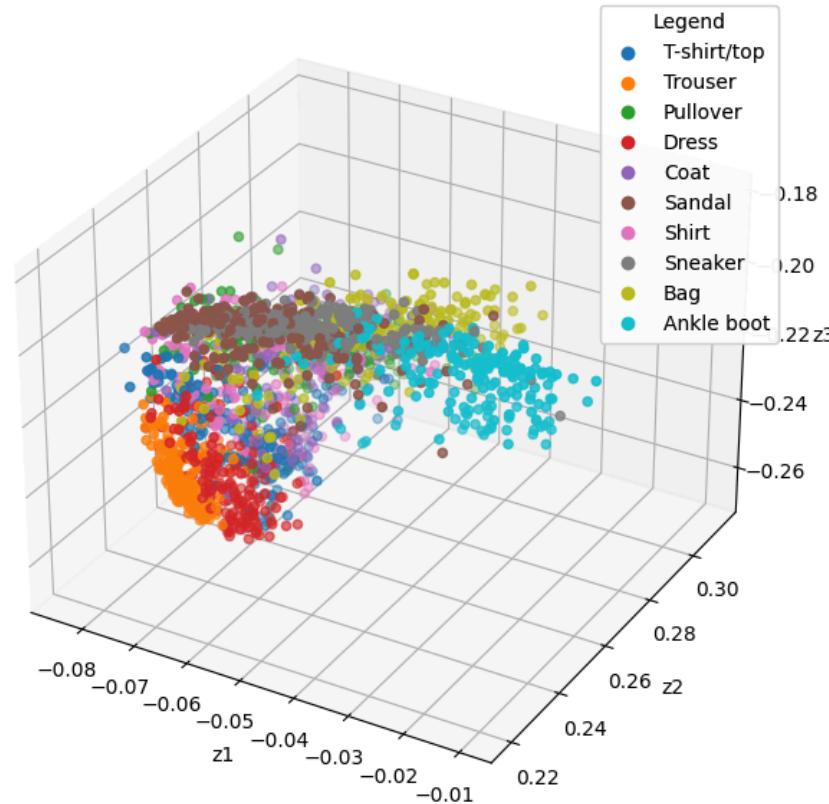
# VAE Review and Coding

Latent space  $Z$  for autoencoder on the clothes dataset with  $d_z = 3$

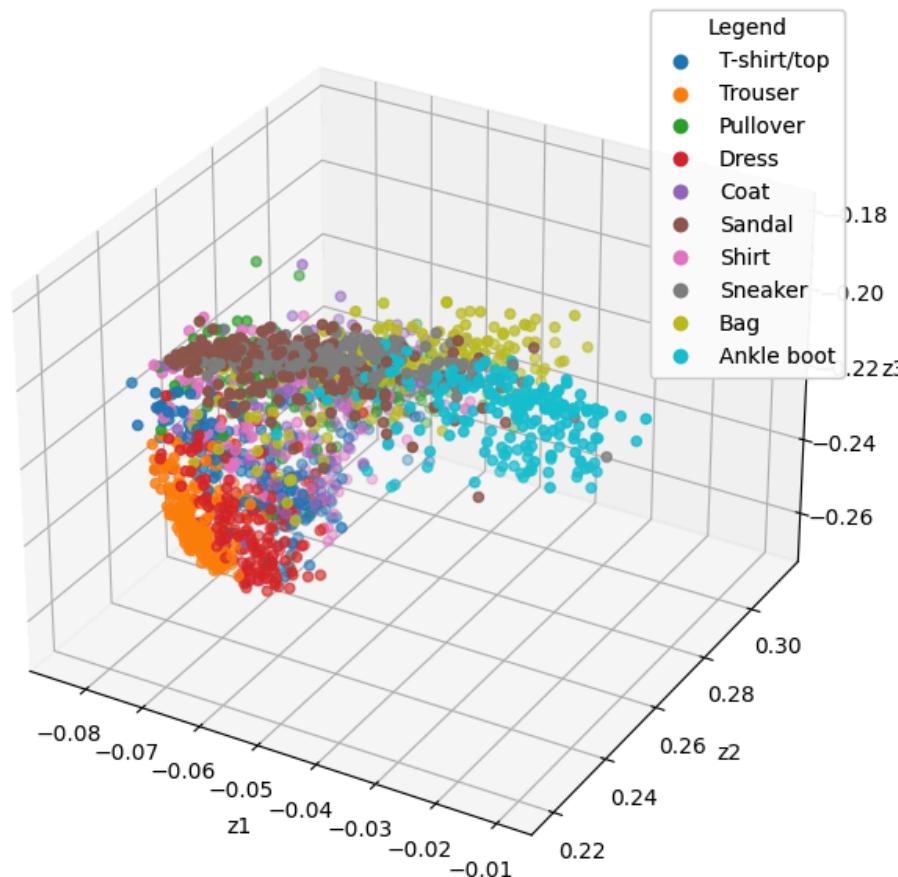


# VAE Review and Coding

What happens if we decode a new point?

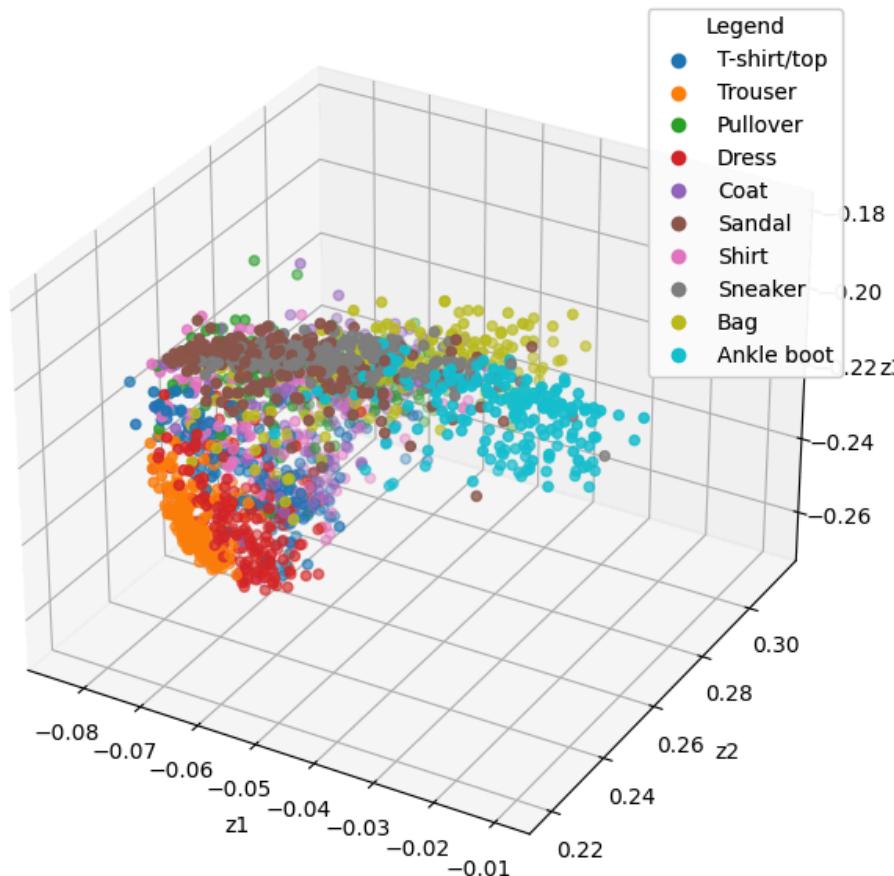


# VAE Review and Coding



Autoencoder generative model:

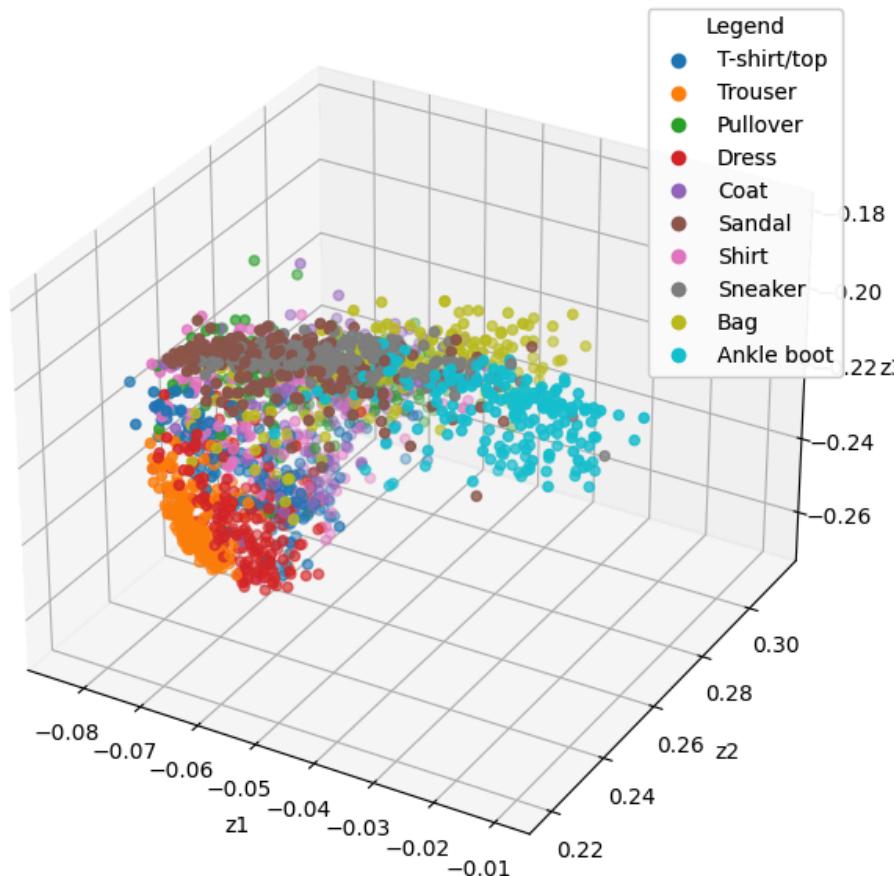
# VAE Review and Coding



Autoencoder generative model:

Encode  $\begin{bmatrix} \mathbf{x}_{[1]} \\ \vdots \\ \mathbf{x}_{[n]} \end{bmatrix}$  into  $\begin{bmatrix} \mathbf{z}_{[1]} \\ \vdots \\ \mathbf{z}_{[n]} \end{bmatrix}$

# VAE Review and Coding

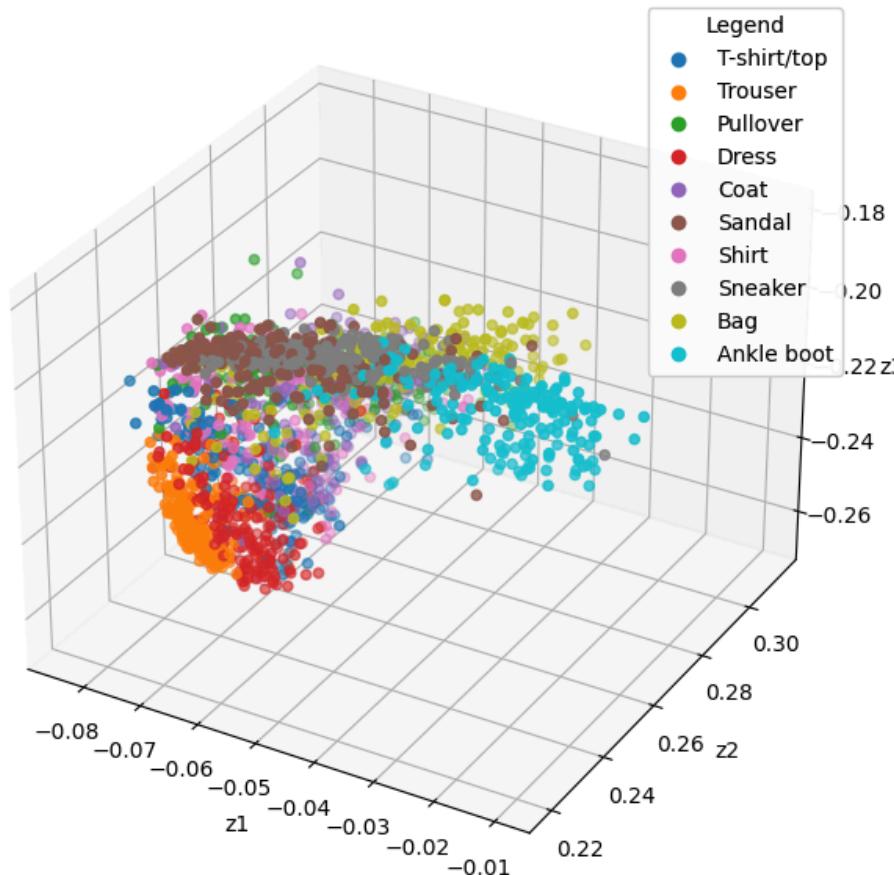


Autoencoder generative model:

Encode  $\begin{bmatrix} \mathbf{x}_{[1]} \\ \vdots \\ \mathbf{x}_{[n]} \end{bmatrix}$  into  $\begin{bmatrix} \mathbf{z}_{[1]} \\ \vdots \\ \mathbf{z}_{[n]} \end{bmatrix}$

Pick a point  $\mathbf{z}_{[k]}$

# VAE Review and Coding



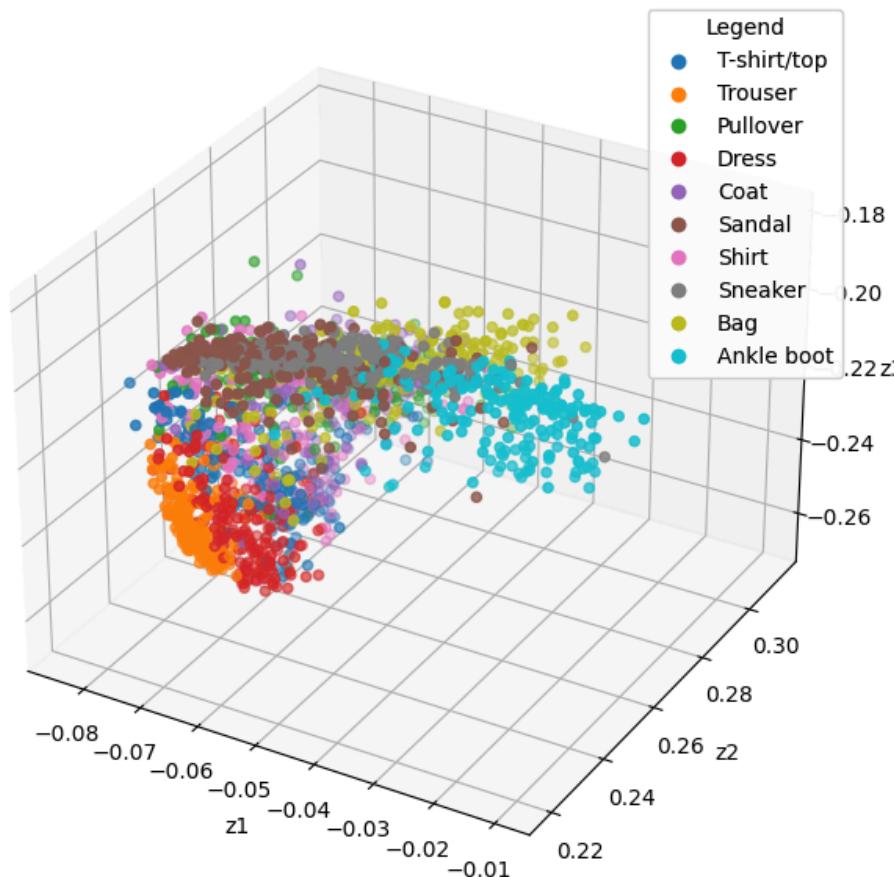
Autoencoder generative model:

Encode  $\begin{bmatrix} \mathbf{x}_{[1]} \\ \vdots \\ \mathbf{x}_{[n]} \end{bmatrix}$  into  $\begin{bmatrix} \mathbf{z}_{[1]} \\ \vdots \\ \mathbf{z}_{[n]} \end{bmatrix}$

Pick a point  $\mathbf{z}_{[k]}$

Add some noise  $\mathbf{z}_{\text{new}} = \mathbf{z}_{[k]} + \boldsymbol{\varepsilon}$

# VAE Review and Coding



Autoencoder generative model:

Encode  $\begin{bmatrix} \mathbf{x}_{[1]} \\ \vdots \\ \mathbf{x}_{[n]} \end{bmatrix}$  into  $\begin{bmatrix} \mathbf{z}_{[1]} \\ \vdots \\ \mathbf{z}_{[n]} \end{bmatrix}$

Pick a point  $\mathbf{z}_{[k]}$

Add some noise  $\mathbf{z}_{\text{new}} = \mathbf{z}_{[k]} + \boldsymbol{\varepsilon}$

Decode  $\mathbf{z}_{\text{new}}$  into  $\mathbf{x}_{\text{new}}$

# VAE Review and Coding



# VAE Review and Coding



$$f^{-1}(z_k + \epsilon, \theta_d)$$

But there is a problem, the **curse  
of dimensionality**

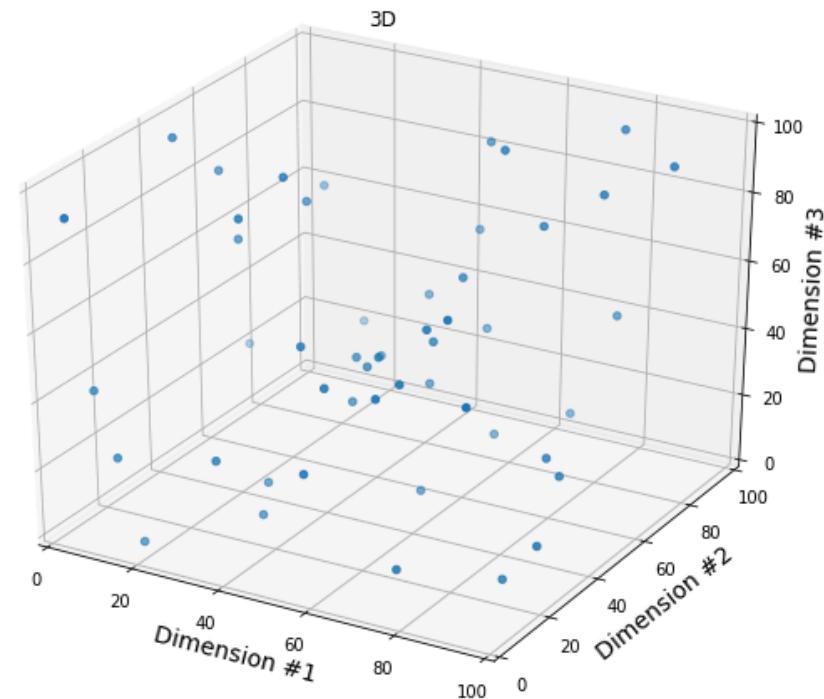
But there is a problem, the **curse  
of dimensionality**

As  $d_z$  increases, points move  
further and further apart

# VAE Review and Coding

But there is a problem, the **curse of dimensionality**

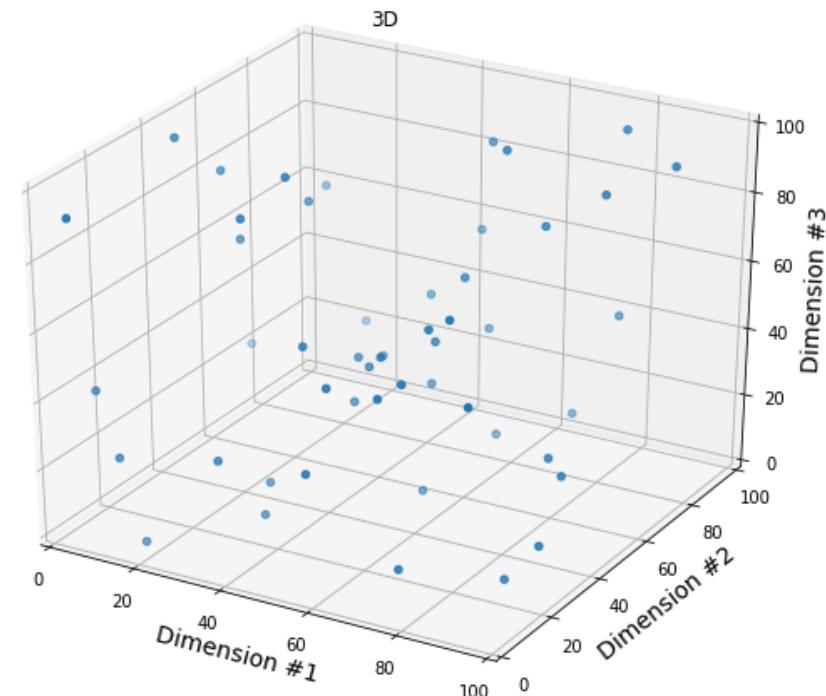
As  $d_z$  increases, points move further and further apart



# VAE Review and Coding

But there is a problem, the **curse of dimensionality**

As  $d_z$  increases, points move further and further apart



$f^{-1}(z + \varepsilon)$  will produce either garbage, or  $z$

# VAE Review and Coding

Variational autoencoders (VAEs) do three things:

# VAE Review and Coding

Variational autoencoders (VAEs) do three things:

1. Make it easy to sample random  $z$

# VAE Review and Coding

Variational autoencoders (VAEs) do three things:

1. Make it easy to sample random  $z$
2. Keep all  $z_{[1]}, \dots, z_{[n]}$  close together in a small region

# VAE Review and Coding

Variational autoencoders (VAEs) do three things:

1. Make it easy to sample random  $z$
2. Keep all  $z_{[1]}, \dots, z_{[n]}$  close together in a small region
3. Ensure that  $z + \epsilon$  is always meaningful

# VAE Review and Coding

Variational autoencoders (VAEs) do three things:

1. Make it easy to sample random  $z$
2. Keep all  $z_{[1]}, \dots, z_{[n]}$  close together in a small region
3. Ensure that  $z + \epsilon$  is always meaningful

How?

# VAE Review and Coding

Variational autoencoders (VAEs) do three things:

1. Make it easy to sample random  $z$
2. Keep all  $z_{[1]}, \dots, z_{[n]}$  close together in a small region
3. Ensure that  $z + \epsilon$  is always meaningful

How?

Make  $z_{[1]}, \dots, z_{[n]}$  normally distributed

# VAE Review and Coding

Variational autoencoders (VAEs) do three things:

1. Make it easy to sample random  $z$
2. Keep all  $z_{[1]}, \dots, z_{[n]}$  close together in a small region
3. Ensure that  $z + \epsilon$  is always meaningful

How?

Make  $z_{[1]}, \dots, z_{[n]}$  normally distributed

$$z \sim \mathcal{N}(\mu, \sigma), \quad \mu = 0, \sigma = 1$$

# VAE Review and Coding

If  $z_{[1]}, \dots, z_{[n]}$  are distributed following  $\mathcal{N}(0, 1)$ :

# VAE Review and Coding

If  $z_{[1]}, \dots, z_{[n]}$  are distributed following  $\mathcal{N}(0, 1)$ :

1. 99.7% of  $z_{[1]}, \dots, z_{[n]}$  lie within  $3\sigma = [-3, 3]$

# VAE Review and Coding

If  $z_{[1]}, \dots, z_{[n]}$  are distributed following  $\mathcal{N}(0, 1)$ :

1. 99.7% of  $z_{[1]}, \dots, z_{[n]}$  lie within  $3\sigma = [-3, 3]$
2. Make it easy to generate new  $z$ , just sample  $z \sim \mathcal{N}(0, 1)$

# VAE Review and Coding

**Key idea 1:** We want to model the distribution over the dataset  $X$

$$P(x; \theta), \quad x \sim X$$

# VAE Review and Coding

**Key idea 1:** We want to model the distribution over the dataset  $X$

$$P(x; \theta), \quad x \sim X$$

We want to learn  $\theta$  that best models the distribution of possible faces

# VAE Review and Coding

**Key idea 1:** We want to model the distribution over the dataset  $X$

$$P(x; \theta), \quad x \sim X$$

We want to learn  $\theta$  that best models the distribution of possible faces

Large  $P(x; \theta)$



$P(x; \theta) \approx 0$



# VAE Review and Coding

**Key idea 2:** There is some latent variable  $z$  which generates data  $x$

# VAE Review and Coding

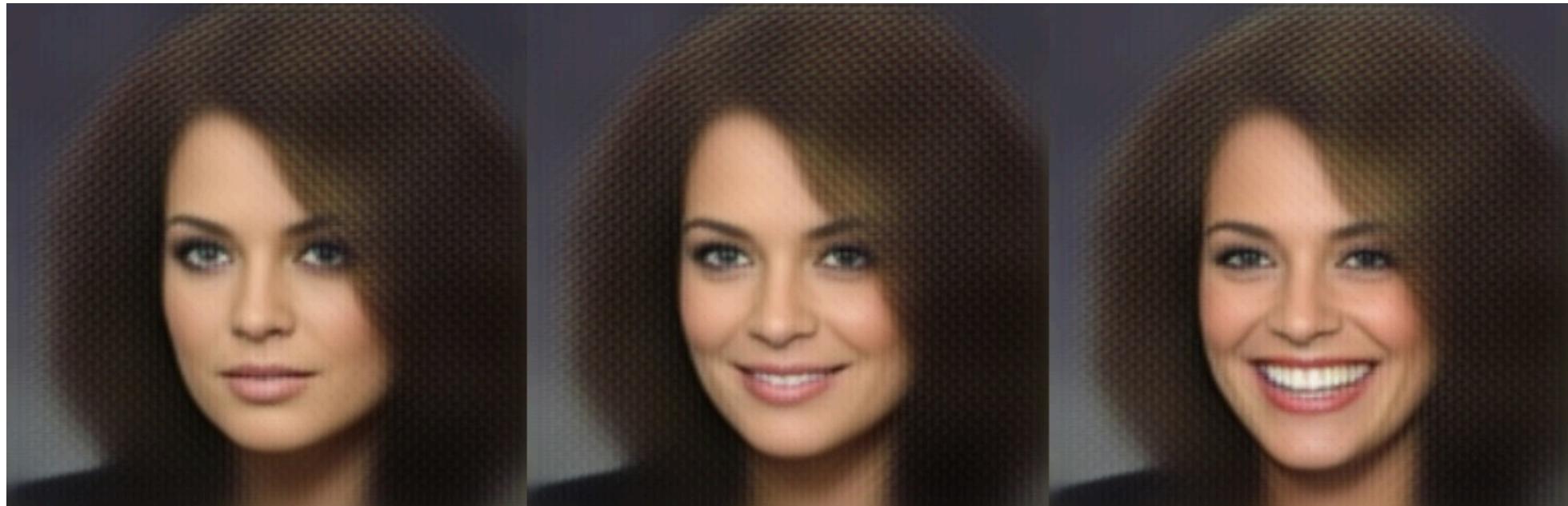
**Key idea 2:** There is some latent variable  $z$  which generates data  $x$

$x :$



$z : [\text{woman brown hair (frown} \mid \text{smile)}]$

# VAE Review and Coding



# VAE Review and Coding



Network can only see  $x$ , it cannot directly observe  $z$

# VAE Review and Coding



Network can only see  $x$ , it cannot directly observe  $z$

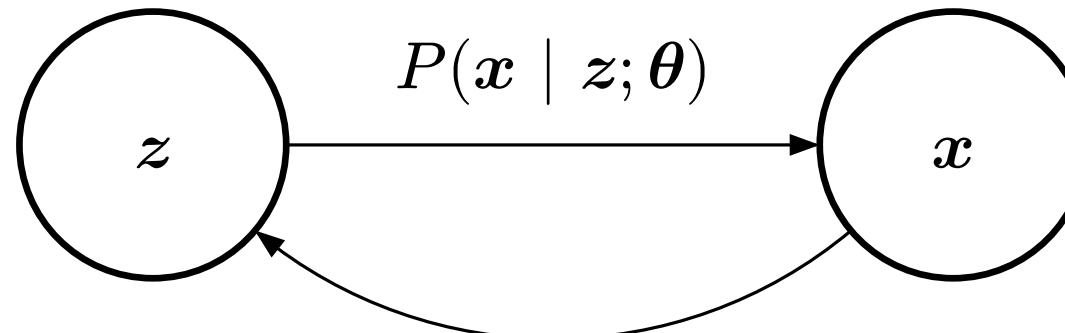
Given  $x$ , find the probability that the person is smiling  $P(z \mid x; \theta)$

# VAE Review and Coding

We cast the autoencoding task as a **variational inference** problem

# VAE Review and Coding

We cast the autoencoding task as a **variational inference** problem



Decoder

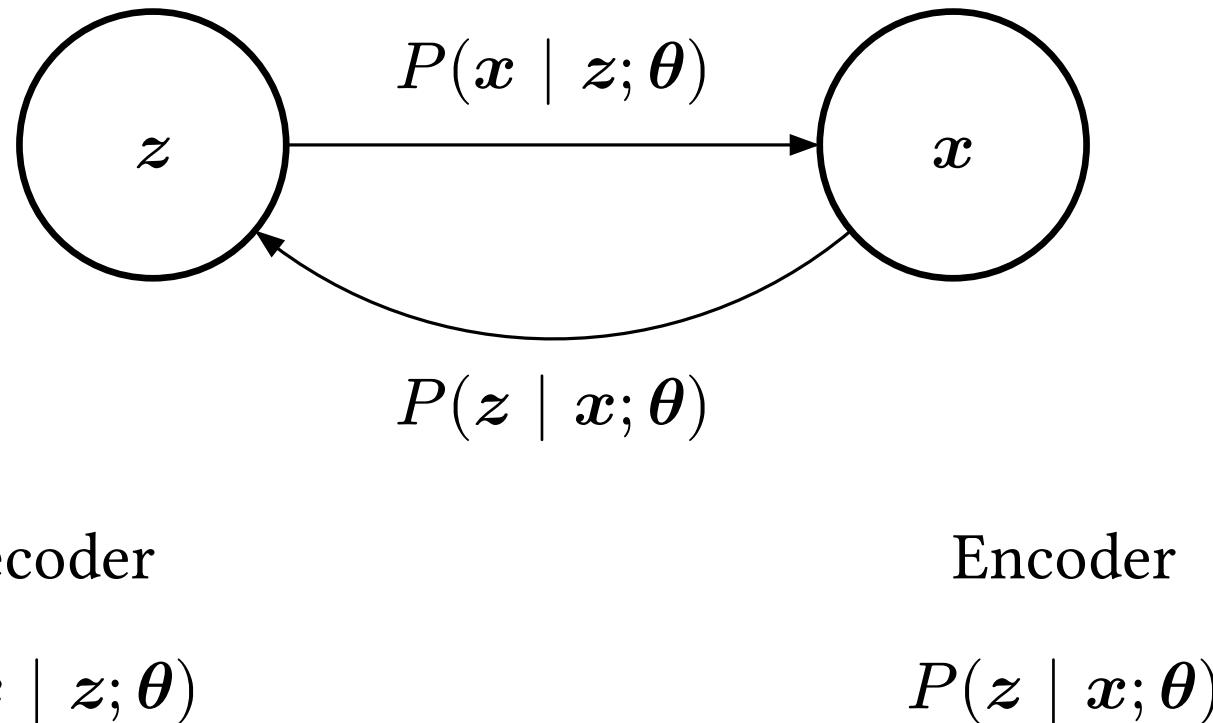
$$P(x | z; \theta)$$

Encoder

$$P(z | x; \theta)$$

# VAE Review and Coding

We cast the autoencoding task as a **variational inference** problem



We want to learn both the encoder and decoder:  $P(z, x; \theta)$

# VAE Review and Coding

How do we implement  $f$  (i.e.,  $P(z \mid x; \theta)$ )?

# VAE Review and Coding

How do we implement  $f$  (i.e.,  $P(z | x; \theta)$ )?

$$f : X \times \Theta \mapsto \Delta Z$$

# VAE Review and Coding

How do we implement  $f$  (i.e.,  $P(z | x; \theta)$ )?

$$f : X \times \Theta \mapsto \Delta Z$$

Normal distribution has a mean  $\mu \in \mathbb{R}$  and standard deviation  $\sigma \in \mathbb{R}_+$

# VAE Review and Coding

How do we implement  $f$  (i.e.,  $P(z | x; \theta)$ )?

$$f : X \times \Theta \mapsto \Delta Z$$

Normal distribution has a mean  $\mu \in \mathbb{R}$  and standard deviation  $\sigma \in \mathbb{R}_+$

Our encoder should output  $d_z$  means and  $d_z$  standard deviations

# VAE Review and Coding

How do we implement  $f$  (i.e.,  $P(z | x; \theta)$ )?

$$f : X \times \Theta \mapsto \Delta Z$$

Normal distribution has a mean  $\mu \in \mathbb{R}$  and standard deviation  $\sigma \in \mathbb{R}_+$

Our encoder should output  $d_z$  means and  $d_z$  standard deviations

$$f : X \times \Theta \mapsto \mathbb{R}^{d_z} \times \mathbb{R}_+^{d_z}$$

# VAE Review and Coding

```
core = nn.Sequential(...)  
mu_layer = nn.Linear(d_h, d_z)  
# Neural networks output real numbers  
# But sigma must be positive  
# Output log sigma, because e^(sigma) is always positive  
log_sigma_layer = nn.Linear(d_h, d_z)  
# Alternatively, one sigma for all data  
log_sigma = jnp.ones((d_z,))  
  
tmp = core(x)  
mu = mu_layer(tmp)  
log_sigma = log_sigma_layer(tmp)  
distribution = (mu, exp(sigma))
```

# VAE Review and Coding

We covered the encoder

$$f : X \times \Theta \mapsto \Delta Z$$

We can use the same decoder as a standard autoencoder

# VAE Review and Coding

We covered the encoder

$$f : X \times \Theta \mapsto \Delta Z$$

We can use the same decoder as a standard autoencoder

$$f^{-1} : Z \times \Theta \mapsto X$$

Encoder outputs a distribution  $\Delta Z$  but decoder input is  $Z$

# VAE Review and Coding

We covered the encoder

$$f : X \times \Theta \mapsto \Delta Z$$

We can use the same decoder as a standard autoencoder

$$f^{-1} : Z \times \Theta \mapsto X$$

Encoder outputs a distribution  $\Delta Z$  but decoder input is  $Z$

**Solution:** Sample a vector  $z$  from the distribution  $\Delta Z$

# VAE Review and Coding

Put it all together

# VAE Review and Coding

Put it all together

**Step 1:** Encode the input to a normal distribution

$$\mu, \sigma = f(x, \theta_e)$$

# VAE Review and Coding

Put it all together

**Step 1:** Encode the input to a normal distribution

$$\mu, \sigma = f(x, \theta_e)$$

**Step 2:** Generate a sample from distribution

$$z = \mu + \sigma \odot \varepsilon$$

# VAE Review and Coding

Put it all together

**Step 1:** Encode the input to a normal distribution

$$\mu, \sigma = f(x, \theta_e)$$

**Step 2:** Generate a sample from distribution

$$z = \mu + \sigma \odot \varepsilon$$

**Step 3:** Decode the sample

$$x = f^{-1}(z, \theta_d)$$

# VAE Review and Coding

```
# Create normal distribution from input
mu, sigma = model.f(x)

# Randomly sample a z vector from our distribution
epsilon = jax.random.normal(key, x.shape[0])
z = mu + sigma * epsilon

# Decode/reconstruct z back into x
pred_x = model.f_inverse(z)
```

# VAE Review and Coding

Now, all we must do is find  $\theta$  that best explains the dataset distribution

# VAE Review and Coding

Now, all we must do is find  $\theta$  that best explains the dataset distribution

Learned distribution  $P(x; \theta)$  to be close to dataset  $P(x)$ ,  $x \sim X$

# VAE Review and Coding

Now, all we must do is find  $\theta$  that best explains the dataset distribution

Learned distribution  $P(x; \theta)$  to be close to dataset  $P(x)$ ,  $x \sim X$

We started with the KL divergence

$$\arg \min_{\theta} \text{KL}(P(x), P(x; \theta))$$

# VAE Review and Coding

Now, all we must do is find  $\theta$  that best explains the dataset distribution

Learned distribution  $P(x; \theta)$  to be close to dataset  $P(x)$ ,  $x \sim X$

We started with the KL divergence

$$\arg \min_{\theta} \text{KL}(P(x), P(x; \theta))$$

# VAE Review and Coding

From the KL divergence, we derived the **ELBO** loss for the VAE

$$\mathcal{L}(\mathbf{X}, \boldsymbol{\theta}) = \underbrace{\frac{m}{n} \sum_{i=1}^n \sum_{j=1}^{d_z} \left( x_{[i],j} - f^{-1}\left(f\left(\mathbf{x}_{[i]}, \boldsymbol{\theta}_e\right), \boldsymbol{\theta}_d\right)_j \right)^2}_{\text{Reconstruct } \mathbf{x}} - \underbrace{\beta \left( \sum_{i=1}^n \sum_{j=1}^{d_z} \mu_{[i],j}^2 + \sigma_{[i],j}^2 - \log(\sigma_{[i],j}^2) - 1 \right)}_{\text{Make } \mathbf{z} \text{ normally distributed}}$$

# VAE Review and Coding

```
def L(model, x, m, n, key):
    mu, sigma = model.f(x) # Encode input into distribution
    # Sample from distribution
    z = mu + sigma * jax.random.normal(key, x.shape[0])
    # Reconstruct input
    pred_x = model.f_inverse(z)
    # Compute reconstruction and kl loss terms
    recon = jnp.sum((x - pred_x) ** 2)
    kl = jnp.sum(mu ** 2 + sigma ** 2 - jnp.log(sigma ** 2) -
    1)
    # Loss function contains reconstruction and kl terms
    return m / n * recon + kl
```

# VAE Review and Coding

[https://colab.research.google.com/drive/1UyR\\_W6NDIujaJXYlHZh6O3NfaCAMscpH#scrollTo=nmyQ8aE2pSbb](https://colab.research.google.com/drive/1UyR_W6NDIujaJXYlHZh6O3NfaCAMscpH#scrollTo=nmyQ8aE2pSbb)

# Agenda

1. GNN Review
2. VAE Review and Coding
3. Attention
4. Keys and Queries
5. Self Attention
6. Positional Encoding
7. Transformers
8. Coding

# Agenda

1. GNN Review
2. VAE Review and Coding
3. **Attention**
4. Keys and Queries
5. Self Attention
6. Positional Encoding
7. Transformers
8. Coding

# Attention

Today, we will investigate attention and transformers

# Attention

Today, we will investigate attention and transformers

Attention and transformers are the “hottest” topic in deep learning

# Attention

Today, we will investigate attention and transformers

Attention and transformers are the “hottest” topic in deep learning

People use them for almost every task (even if they shouldn’t!)

# Attention

Today, we will investigate attention and transformers

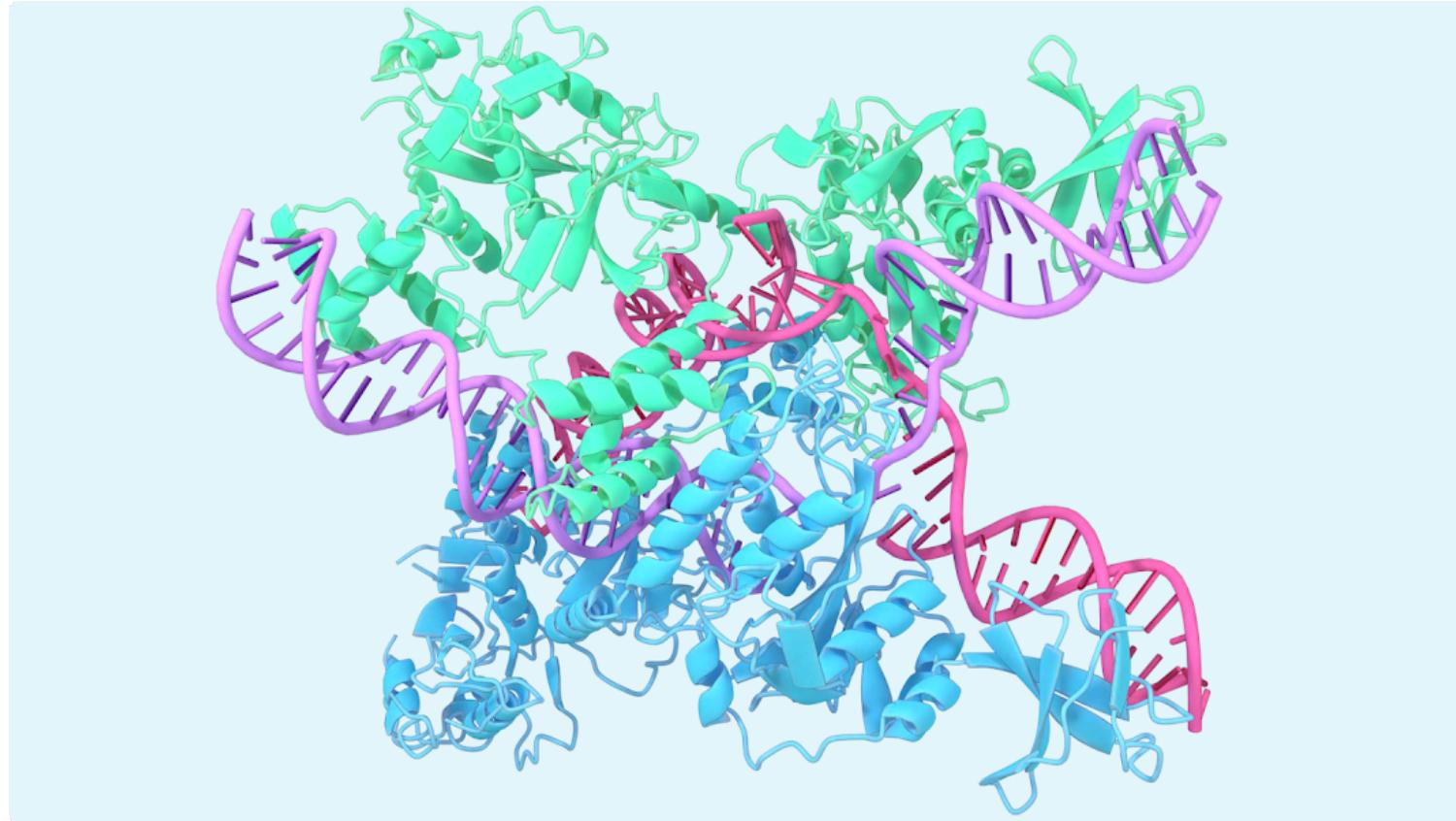
Attention and transformers are the “hottest” topic in deep learning

People use them for almost every task (even if they shouldn’t!)

Let’s review some products based on attention

# Attention

AlphaFold (Nobel prize)



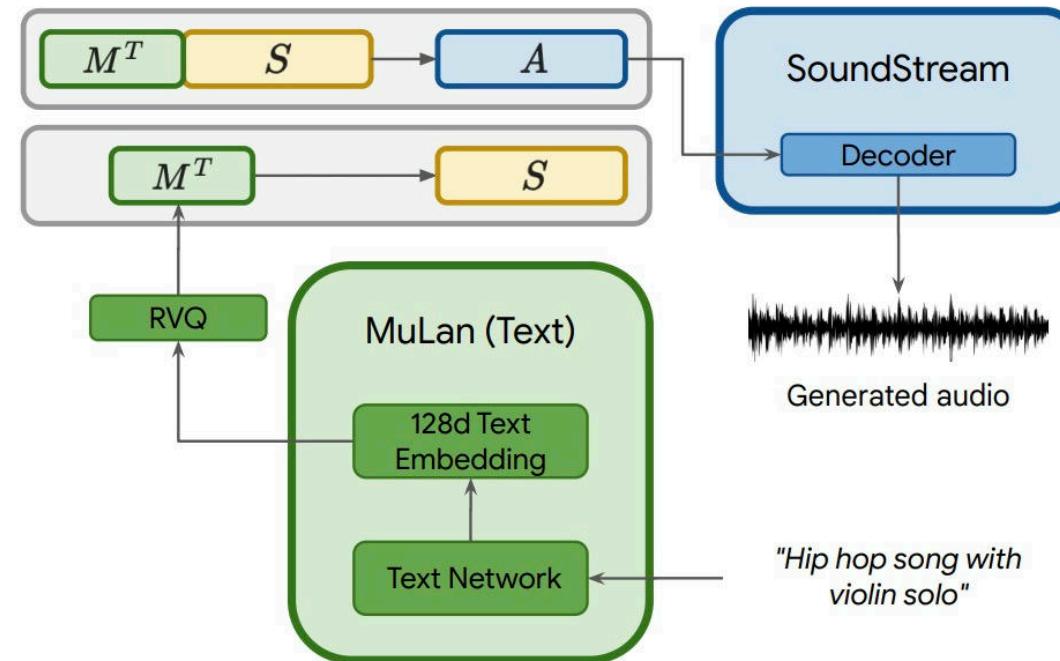
# Attention

ChatGPT, Qwen, LLaMA, Mistral, Doubou, Ernie chatbots



# Attention

## MusicTransformer, MuLan



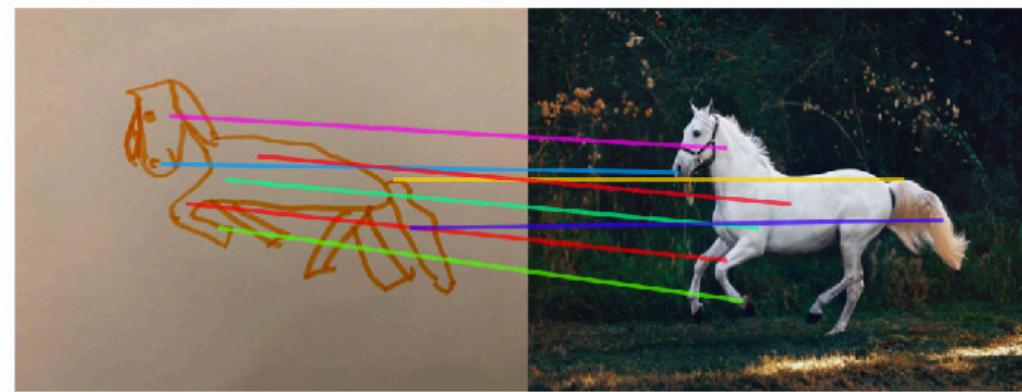
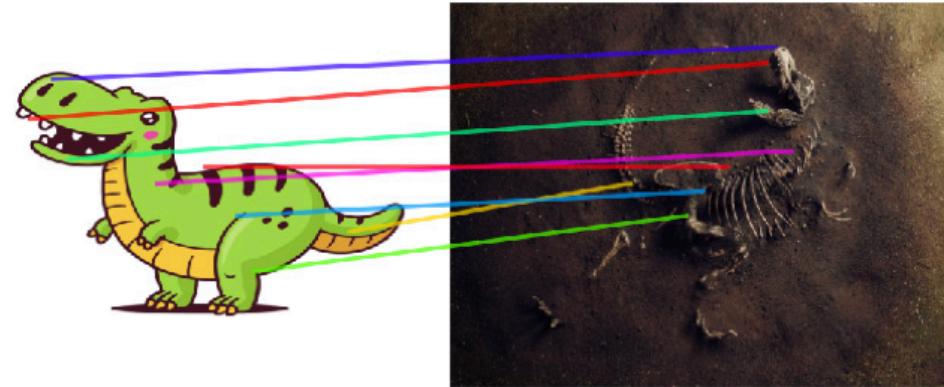
# Attention

Google Translate, Baidu Translate, Apple Translate



# Attention

ViT, DinoV2



# Attention

All these models are **transformers**

# Attention

All these models are **transformers**

At the core of each transformer is **attention**

# Attention

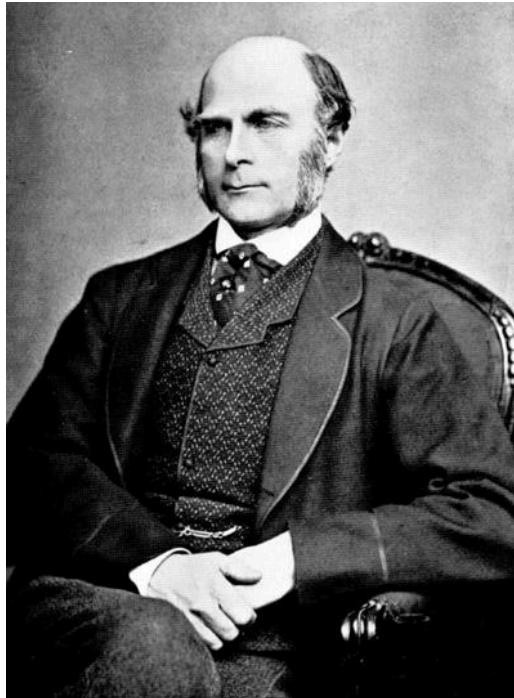
All these models are **transformers**

At the core of each transformer is **attention**

We can derive attention from composite memory

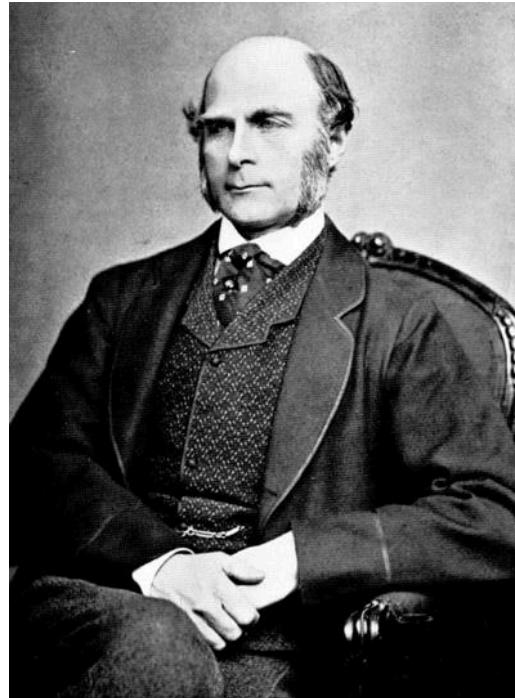
# Attention

Francis Galton (1822-1911)  
photo composite memory

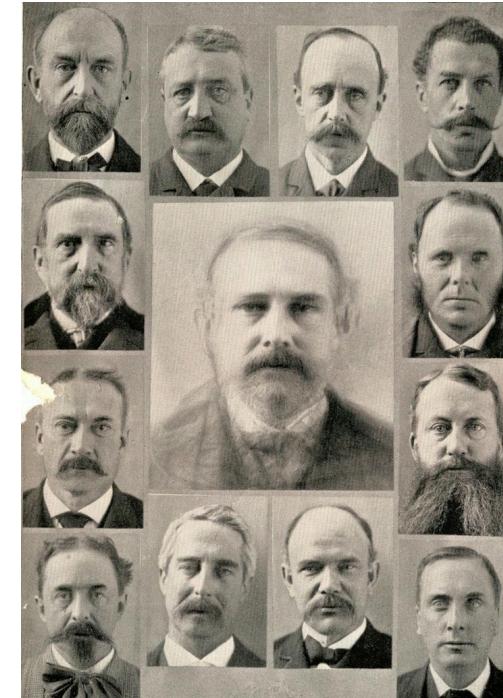


# Attention

Francis Galton (1822-1911)  
photo composite memory



Composite photo of members of a party



# Attention

**Task:** Find a mathematical model of how our mind represents memories

# Attention

**Task:** Find a mathematical model of how our mind represents memories

$X : \mathbb{R}^{h \times w}$  People you see at the party

# Attention

**Task:** Find a mathematical model of how our mind represents memories

$X : \mathbb{R}^{h \times w}$  People you see at the party

$H : \mathbb{R}^{h \times w}$  The image in your mind

# Attention

**Task:** Find a mathematical model of how our mind represents memories

$X : \mathbb{R}^{h \times w}$  People you see at the party

$H : \mathbb{R}^{h \times w}$  The image in your mind

$f : X^T \times \Theta \mapsto H$

# Attention

**Task:** Find a mathematical model of how our mind represents memories

$X : \mathbb{R}^{h \times w}$  People you see at the party

$H : \mathbb{R}^{h \times w}$  The image in your mind

$f : X^T \times \Theta \mapsto H$

Composite photography/memory uses a weighted sum

# Attention

**Task:** Find a mathematical model of how our mind represents memories

$$X : \mathbb{R}^{h \times w} \quad \text{People you see at the party}$$

$$H : \mathbb{R}^{h \times w} \quad \text{The image in your mind}$$

$$f : X^T \times \Theta \mapsto H$$

Composite photography/memory uses a weighted sum

$$f(\mathbf{x}, \boldsymbol{\theta}) = \sum_{i=1}^T \boldsymbol{\theta}^\top \overline{\mathbf{x}}_i$$

# Attention

Limited space, cannot fit everything

# Attention

Limited space, cannot fit everything

Introduced forgetting

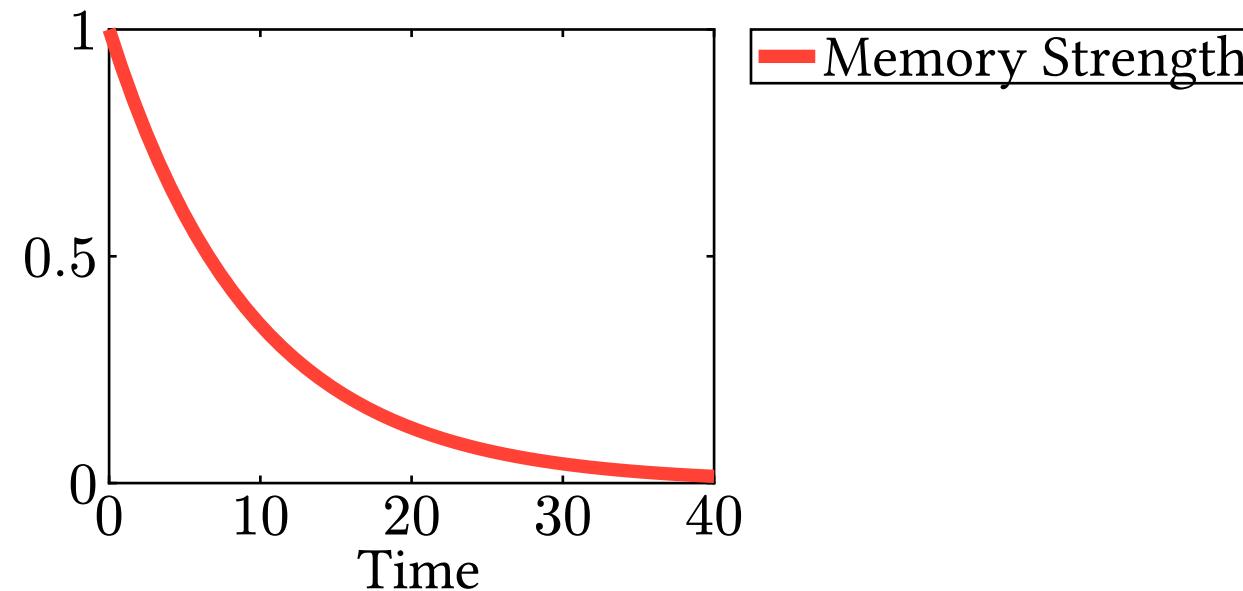
$$\sum_{i=1}^n \gamma^{n-i} \theta^\top x_i$$

# Attention

Limited space, cannot fit everything

Introduced forgetting

$$\sum_{i=1}^n \gamma^{n-i} \theta^\top x_i$$

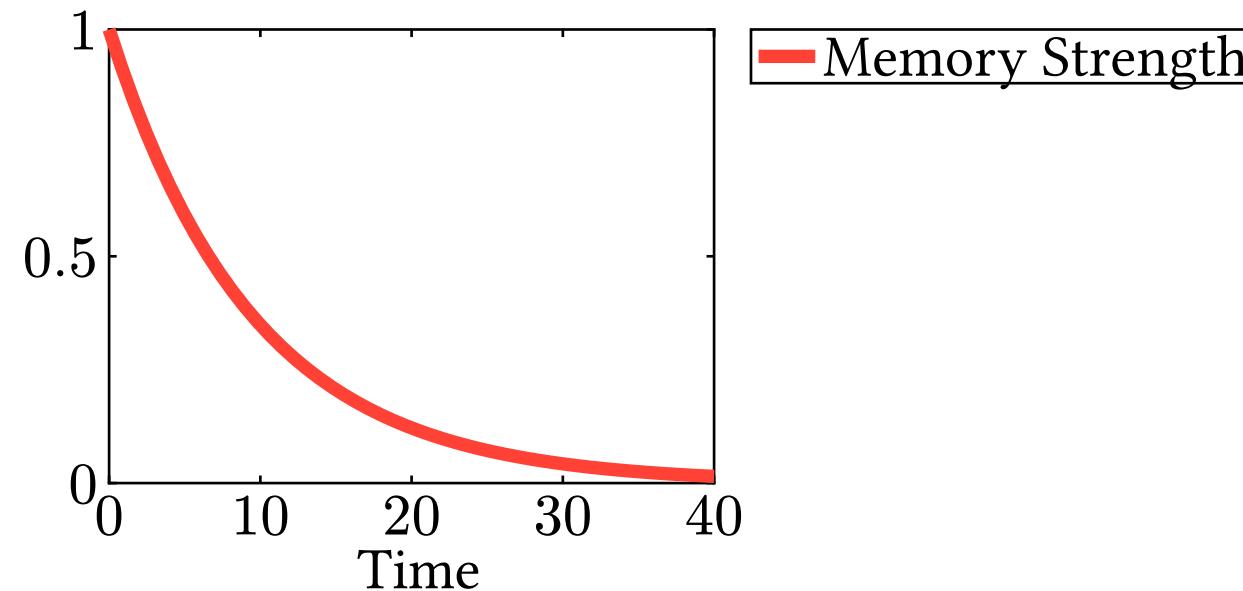


# Attention

Limited space, cannot fit everything

Introduced forgetting

$$\sum_{i=1}^n \gamma^{n-i} \theta^\top x_i$$



**Question:** Does this accurately model what you remember?

# Attention

You go to a party and meet these people in order



# Attention

According to forgetting, the memories should fade with time



$$\gamma^3 \boldsymbol{\theta}^\top \overline{\boldsymbol{x}}_1$$

# Attention

According to forgetting, the memories should fade with time



$$\gamma^3 \boldsymbol{\theta}^\top \bar{\boldsymbol{x}}_1$$

$$\gamma^2 \boldsymbol{\theta}^\top \bar{\boldsymbol{x}}_2$$

# Attention

According to forgetting, the memories should fade with time



$$\gamma^3 \boldsymbol{\theta}^\top \bar{\boldsymbol{x}}_1$$

$$\gamma^2 \boldsymbol{\theta}^\top \bar{\boldsymbol{x}}_2$$

$$\gamma^1 \boldsymbol{\theta}^\top \bar{\boldsymbol{x}}_3$$

# Attention

According to forgetting, the memories should fade with time



$$\gamma^3 \boldsymbol{\theta}^\top \bar{\boldsymbol{x}}_1$$

$$\gamma^2 \boldsymbol{\theta}^\top \bar{\boldsymbol{x}}_2$$

$$\gamma^1 \boldsymbol{\theta}^\top \bar{\boldsymbol{x}}_3$$

$$\gamma^0 \boldsymbol{\theta}^\top \bar{\boldsymbol{x}}_4$$

# Attention

Consider the following case

# Attention

Consider the following case



# Attention

Consider the following case



$$\gamma^4 \boldsymbol{\theta}^\top \overline{\mathbf{x}}_1$$

$$\gamma^3 \boldsymbol{\theta}^\top \overline{\mathbf{x}}_2$$

$$\gamma^2 \boldsymbol{\theta}^\top \overline{\mathbf{x}}_3$$

$$\gamma^1 \boldsymbol{\theta}^\top \overline{\mathbf{x}}_4$$

$$\gamma^0 \boldsymbol{\theta}^\top \overline{\mathbf{x}}_5$$

# Attention

Consider the following case



$$\gamma^4 \theta^\top \bar{x}_1$$

$$\gamma^3 \theta^\top \bar{x}_2$$

$$\gamma^2 \theta^\top \bar{x}_3$$

$$\gamma^1 \theta^\top \bar{x}_4$$

$$\gamma^0 \theta^\top \bar{x}_5$$

**Question:** What will happen to Taylor Swift?

# Attention



$$\gamma^4 \theta^\top \bar{x}_1$$

$$\gamma^3 \theta^\top \bar{x}_2$$

$$\gamma^2 \theta^\top \bar{x}_3$$

$$\gamma^1 \theta^\top \bar{x}_4$$

$$\gamma^0 \theta^\top \bar{x}_5$$

# Attention



$$\gamma^4 \theta^\top \bar{x}_1$$

$$\gamma^3 \theta^\top \bar{x}_2$$

$$\gamma^2 \theta^\top \bar{x}_3$$

$$\gamma^1 \theta^\top \bar{x}_4$$

$$\gamma^0 \theta^\top \bar{x}_5$$

We will forget meeting her!

# Attention



$$\gamma^4 \theta^\top \bar{x}_1$$

$$\gamma^3 \theta^\top \bar{x}_2$$

$$\gamma^2 \theta^\top \bar{x}_3$$

$$\gamma^1 \theta^\top \bar{x}_4$$

$$\gamma^0 \theta^\top \bar{x}_5$$

We will forget meeting her!

Would you forget Taylor Swift at a party?

# Attention

Our model of memory is incomplete

# Attention

Our model of memory is incomplete

Memories are not created equal, some are more important than others

My memory might actually be



$$1.0 \cdot \theta^\top \bar{x}_1$$

$$0.1 \cdot \theta^\top \bar{x}_2$$

$$0.1 \cdot \theta^\top \bar{x}_3$$

$$0.5 \cdot \theta^\top \bar{x}_4$$

$$0.1 \cdot \theta^\top \bar{x}_5$$

My memory might actually be



$$1.0 \cdot \theta^\top \bar{x}_1$$

$$0.1 \cdot \theta^\top \bar{x}_2$$

$$0.1 \cdot \theta^\top \bar{x}_3$$

$$0.5 \cdot \theta^\top \bar{x}_4$$

$$0.1 \cdot \theta^\top \bar{x}_5$$

**Question:** How can we achieve this forgetting?

# Attention

**Question:** How did we achieve forgetting in our recurrent neural network?

# Attention

**Question:** How did we achieve forgetting in our recurrent neural network?

$$f_{\text{forget}}(\boldsymbol{x}, \boldsymbol{\theta}) = \sigma(\boldsymbol{\theta}_\lambda^\top \overline{\boldsymbol{x}})$$

# Attention

**Question:** How did we achieve forgetting in our recurrent neural network?

$$f_{\text{forget}}(\mathbf{x}, \boldsymbol{\theta}) = \sigma(\boldsymbol{\theta}_\lambda^\top \overline{\mathbf{x}})$$

$$f(\mathbf{h}, \mathbf{x}, \boldsymbol{\theta}) = f_{\text{forget}}(\mathbf{x}, \boldsymbol{\theta}) \odot \mathbf{h} + \boldsymbol{\theta}_x^\top \overline{\mathbf{x}}$$

# Attention

**Question:** How did we achieve forgetting in our recurrent neural network?

$$f_{\text{forget}}(\mathbf{x}, \boldsymbol{\theta}) = \sigma(\boldsymbol{\theta}_\lambda^\top \overline{\mathbf{x}})$$

$$f(\mathbf{h}, \mathbf{x}, \boldsymbol{\theta}) = f_{\text{forget}}(\mathbf{x}, \boldsymbol{\theta}) \odot \mathbf{h} + \boldsymbol{\theta}_x^\top \overline{\mathbf{x}}$$

Let us do something similar

# Attention

**Question:** How did we achieve forgetting in our recurrent neural network?

$$f_{\text{forget}}(\mathbf{x}, \boldsymbol{\theta}) = \sigma(\boldsymbol{\theta}_\lambda^\top \overline{\mathbf{x}})$$

$$f(\mathbf{h}, \mathbf{x}, \boldsymbol{\theta}) = f_{\text{forget}}(\mathbf{x}, \boldsymbol{\theta}) \odot \mathbf{h} + \boldsymbol{\theta}_x^\top \overline{\mathbf{x}}$$

Let us do something similar

However, we will write it slightly differently (without recurrence)

# Attention

First, write our forgetting function

# Attention

First, write our forgetting function

$$\lambda(x, \theta_\lambda) = \sigma(\theta_\lambda^\top \bar{x}); \quad \theta_\lambda \in \mathbb{R}^{(d_x+1) \times 1}$$

# Attention

First, write our forgetting function

$$\lambda(x, \theta_\lambda) = \sigma(\theta_\lambda^\top \bar{x}); \quad \theta_\lambda \in \mathbb{R}^{(d_x+1) \times 1}$$

Then, write our composite memory model with forgetting

# Attention

First, write our forgetting function

$$\lambda(x, \theta_\lambda) = \sigma(\theta_\lambda^\top \bar{x}); \quad \theta_\lambda \in \mathbb{R}^{(d_x+1) \times 1}$$

Then, write our composite memory model with forgetting

$$f\left(\begin{bmatrix} x_1 \\ \vdots \\ x_T \end{bmatrix}, \theta\right) = \sum_{i=1}^T \theta^\top x_i \cdot \lambda(x_i, \theta_\lambda)$$

# Attention

First, write our forgetting function

$$\lambda(x, \theta_\lambda) = \sigma(\theta_\lambda^\top \bar{x}); \quad \theta_\lambda \in \mathbb{R}^{(d_x+1) \times 1}$$

Then, write our composite memory model with forgetting

$$f\left(\begin{bmatrix} x_1 \\ \vdots \\ x_T \end{bmatrix}, \theta\right) = \sum_{i=1}^T \theta^\top x_i \cdot \lambda(x_i, \theta_\lambda)$$

This is one form of **attention**

# Attention

First, write our forgetting function

$$\lambda(x, \theta_\lambda) = \sigma(\theta_\lambda^\top \bar{x}); \quad \theta_\lambda \in \mathbb{R}^{(d_x+1) \times 1}$$

Then, write our composite memory model with forgetting

$$f\left(\begin{bmatrix} x_1 \\ \vdots \\ x_T \end{bmatrix}, \theta\right) = \sum_{i=1}^T \theta^\top x_i \cdot \lambda(x_i, \theta_\lambda)$$

This is one form of **attention**

We only pay attention to specific inputs

# Attention

We can use this simple form of attention to pay attention to Taylor Swift

# Attention

We can use this simple form of attention to pay attention to Taylor Swift



# Attention

$$\begin{array}{ccccc} \lambda(x_1, \theta_\lambda) & \lambda(x_2, \theta_\lambda) & \lambda(x_3, \theta_\lambda) & \lambda(x_4, \theta_\lambda) & \lambda(x_5, \theta_\lambda) \\ \cdot \theta^\top \bar{x}_1 & \cdot \theta^\top \bar{x}_2 & \cdot \theta^\top \bar{x}_3 & \cdot \theta^\top \bar{x}_4 & \cdot \theta^\top \bar{x}_5 \end{array}$$

# Attention

$$\begin{array}{ccccc} \lambda(x_1, \theta_\lambda) & \lambda(x_2, \theta_\lambda) & \lambda(x_3, \theta_\lambda) & \lambda(x_4, \theta_\lambda) & \lambda(x_5, \theta_\lambda) \\ \cdot \theta^\top \bar{x}_1 & \cdot \theta^\top \bar{x}_2 & \cdot \theta^\top \bar{x}_3 & \cdot \theta^\top \bar{x}_4 & \cdot \theta^\top \bar{x}_5 \end{array}$$

**Question:** What do the images look like now?

# Attention

$$\lambda(x_1, \theta_\lambda) \\ \cdot \theta^\top \bar{x}_1$$

$$\lambda(x_2, \theta_\lambda) \\ \cdot \theta^\top \bar{x}_2$$

$$\lambda(x_3, \theta_\lambda) \\ \cdot \theta^\top \bar{x}_3$$

$$\lambda(x_4, \theta_\lambda) \\ \cdot \theta^\top \bar{x}_4$$

$$\lambda(x_5, \theta_\lambda) \\ \cdot \theta^\top \bar{x}_5$$

**Question:** What do the images look like now?



# Attention

This form of attention will learn to pay attention to everyone!

# Attention

This form of attention will learn to pay attention to everyone!

$$1.0 \cdot \theta^\top \bar{x}_1 \quad 1.0 \cdot \theta^\top \bar{x}_2 \quad 1.0 \cdot \theta^\top \bar{x}_3 \quad 1.0 \cdot \theta^\top \bar{x}_4 \quad 1.0 \cdot \theta^\top \bar{x}_5$$

# Attention

This form of attention will learn to pay attention to everyone!

$$1.0 \cdot \theta^\top \bar{x}_1$$

$$1.0 \cdot \theta^\top \bar{x}_2$$

$$1.0 \cdot \theta^\top \bar{x}_3$$

$$1.0 \cdot \theta^\top \bar{x}_4$$

$$1.0 \cdot \theta^\top \bar{x}_5$$



# Attention

This form of attention will learn to pay attention to everyone!

$$1.0 \cdot \theta^\top \bar{x}_1$$

$$1.0 \cdot \theta^\top \bar{x}_2$$

$$1.0 \cdot \theta^\top \bar{x}_3$$

$$1.0 \cdot \theta^\top \bar{x}_4$$

$$1.0 \cdot \theta^\top \bar{x}_5$$



Not a good model of attention!

# Attention

We should normalize  $\lambda(x, \theta_\lambda)$  to model a finite attention span

# Attention

We should normalize  $\lambda(x, \theta_\lambda)$  to model a finite attention span

For example, we can make

$$\sum_{i=1}^T \lambda(x, \theta_\lambda) = 1$$

# Attention

We should normalize  $\lambda(\mathbf{x}, \theta_\lambda)$  to model a finite attention span

For example, we can make

$$\sum_{i=1}^T \lambda(\mathbf{x}, \theta_\lambda) = 1$$

This will limit the total amount of attention that we have

# Attention

We should normalize  $\lambda(\mathbf{x}, \theta_\lambda)$  to model a finite attention span

For example, we can make

$$\sum_{i=1}^T \lambda(\mathbf{x}, \theta_\lambda) = 1$$

This will limit the total amount of attention that we have

**Question:** Do we know of any functions with this property?

# Attention

We should normalize  $\lambda(\mathbf{x}, \theta_\lambda)$  to model a finite attention span

For example, we can make

$$\sum_{i=1}^T \lambda(\mathbf{x}, \theta_\lambda) = 1$$

This will limit the total amount of attention that we have

**Question:** Do we know of any functions with this property?

**Answer:** Softmax!

# Attention

The softmax function maps real numbers to the simplex (probabilities)

$$\text{softmax} : \mathbb{R}^k \mapsto \Delta^{k-1}$$

# Attention

The softmax function maps real numbers to the simplex (probabilities)

$$\text{softmax} : \mathbb{R}^k \mapsto \Delta^{k-1}$$

$$\text{softmax} \left( \begin{bmatrix} x_1 \\ \vdots \\ x_k \end{bmatrix} \right) = \frac{\exp(\mathbf{x})}{\sum_{i=1}^k \exp(x_i)} = \begin{bmatrix} \frac{\exp(x_1)}{\exp(x_1)+\exp(x_2)+\dots+\exp(x_k)} \\ \frac{\exp(x_2)}{\exp(x_1)+\exp(x_2)+\dots+\exp(x_k)} \\ \vdots \\ \frac{\exp(x_k)}{\exp(x_1)+\exp(x_2)+\dots+\exp(x_k)} \end{bmatrix}$$

# Attention

Let us rewrite attention using softmax

# Attention

Let us rewrite attention using softmax

The attention we pay to person  $i$  is

$$\lambda \left( \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \boldsymbol{\theta}_\lambda \right)_i = \frac{\exp(\boldsymbol{\theta}_\lambda^\top \bar{\mathbf{x}}_i)}{\sum_{j=1}^T \exp(\boldsymbol{\theta}_\lambda^\top \bar{\mathbf{x}}_j)}$$

# Attention

$$\lambda \left( \begin{bmatrix} x_1 \\ \vdots \\ x_5 \end{bmatrix}, \theta_\lambda \right)_1 \lambda \left( \begin{bmatrix} x_1 \\ \vdots \\ x_5 \end{bmatrix}, \theta_\lambda \right)_2 \lambda \left( \begin{bmatrix} x_1 \\ \vdots \\ x_5 \end{bmatrix}, \theta_\lambda \right)_3 \lambda \left( \begin{bmatrix} x_1 \\ \vdots \\ x_5 \end{bmatrix}, \theta_\lambda \right)_4 \lambda \left( \begin{bmatrix} x_1 \\ \vdots \\ x_5 \end{bmatrix}, \theta_\lambda \right)_5$$
$$\cdot \theta^\top \bar{x}_1 \quad \cdot \theta^\top \bar{x}_2 \quad \cdot \theta^\top \bar{x}_3 \quad \cdot \theta^\top \bar{x}_4 \quad \cdot \theta^\top \bar{x}_5$$

# Attention

$$\lambda \left( \begin{bmatrix} x_1 \\ \vdots \\ x_5 \end{bmatrix}, \theta_\lambda \right)_1 \lambda \left( \begin{bmatrix} x_1 \\ \vdots \\ x_5 \end{bmatrix}, \theta_\lambda \right)_2 \lambda \left( \begin{bmatrix} x_1 \\ \vdots \\ x_5 \end{bmatrix}, \theta_\lambda \right)_3 \lambda \left( \begin{bmatrix} x_1 \\ \vdots \\ x_5 \end{bmatrix}, \theta_\lambda \right)_4 \lambda \left( \begin{bmatrix} x_1 \\ \vdots \\ x_5 \end{bmatrix}, \theta_\lambda \right)_5$$
$$\cdot \theta^\top \bar{x}_1 \quad \cdot \theta^\top \bar{x}_2 \quad \cdot \theta^\top \bar{x}_3 \quad \cdot \theta^\top \bar{x}_4 \quad \cdot \theta^\top \bar{x}_5$$



# Attention

$$0.70 \cdot \theta^\top \bar{x}_1 \quad 0.04 \cdot \theta^\top \bar{x}_2 \quad 0.03 \cdot \theta^\top \bar{x}_3 \quad 0.20 \cdot \theta^\top \bar{x}_4 \quad 0.03 \cdot \theta^\top \bar{x}_5$$

# Attention

$$0.70 \cdot \theta^\top \bar{x}_1 \quad 0.04 \cdot \theta^\top \bar{x}_2 \quad 0.03 \cdot \theta^\top \bar{x}_3 \quad 0.20 \cdot \theta^\top \bar{x}_4 \quad 0.03 \cdot \theta^\top \bar{x}_5$$



# Attention

$$0.70 \cdot \theta^\top \bar{x}_1 \quad 0.04 \cdot \theta^\top \bar{x}_2 \quad 0.03 \cdot \theta^\top \bar{x}_3 \quad 0.20 \cdot \theta^\top \bar{x}_4 \quad 0.03 \cdot \theta^\top \bar{x}_5$$



$$0.70 + 0.04 + 0.03 + 0.20 + 0.03 = 1.0$$

# Attention

$$\lambda \left( \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \boldsymbol{\theta}_\lambda \right)_i = \frac{\exp(\boldsymbol{\theta}_\lambda^\top \bar{\mathbf{x}})}{\sum_{j=1}^T \exp(\boldsymbol{\theta}_\lambda^\top \bar{\mathbf{x}}_j)}$$

$$\lambda \left( \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \boldsymbol{\theta}_\lambda \right) = \begin{bmatrix} \frac{\exp(\boldsymbol{\theta}_\lambda^\top \bar{\mathbf{x}}_1)}{\sum_{j=1}^T \exp(\boldsymbol{\theta}_\lambda^\top \bar{\mathbf{x}}_j)} \\ \vdots \\ \frac{\exp(\boldsymbol{\theta}_\lambda^\top \bar{\mathbf{x}}_T)}{\sum_{j=1}^T \exp(\boldsymbol{\theta}_\lambda^\top \bar{\mathbf{x}}_j)} \end{bmatrix}$$

## Keys and Queries

# Attention

The modern form of attention behaves like a database

# Attention

The modern form of attention behaves like a database

We label each person at the party with a **key**

# Attention

The modern form of attention behaves like a database

We label each person at the party with a **key**

The key describes the content of each  $x$

# Attention

The modern form of attention behaves like a database

We label each person at the party with a **key**

The key describes the content of each  $x$



# Attention

The modern form of attention behaves like a database

We label each person at the party with a **key**

The key describes the content of each  $x$



Musician

# Attention

The modern form of attention behaves like a database

We label each person at the party with a **key**

The key describes the content of each  $x$



Musician

Lawyer

# Attention

The modern form of attention behaves like a database

We label each person at the party with a **key**

The key describes the content of each  $x$



Musician

Lawyer

Shopkeeper

# Attention

The modern form of attention behaves like a database

We label each person at the party with a **key**

The key describes the content of each  $x$



Musician

Lawyer

Shopkeeper

Chef

# Attention

The modern form of attention behaves like a database

We label each person at the party with a **key**

The key describes the content of each  $x$



Musician

Lawyer

Shopkeeper

Chef

Scientist

# Attention

The modern form of attention behaves like a database

We label each person at the party with a **key**

The key describes the content of each  $x$



Musician

Lawyer

Shopkeeper

Chef

Scientist

# Attention

# Attention

Then we compute a **query** that corresponds to the key

**Query:** Which person will help me on my exam?

# Attention

Then we compute a **query** that corresponds to the key

**Query:** Which person will help me on my exam?

Musician

Lawyer

Shopkeeper

Chef

Scientist

# Attention

Then we compute a **query** that corresponds to the key

**Query:** Which person will help me on my exam?



Musician

Lawyer

Shopkeeper

Chef

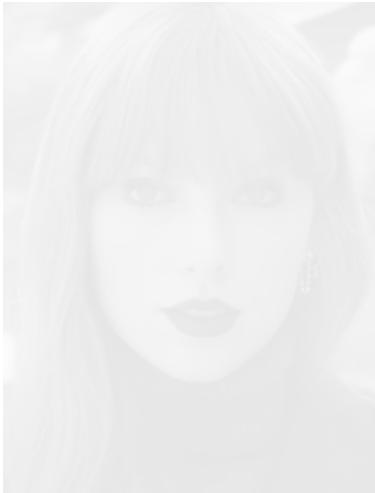
Scientist

# Attention

Then we compute a **query** that corresponds to the key

**Query:** Which person will help me on my exam?

Musician



Lawyer



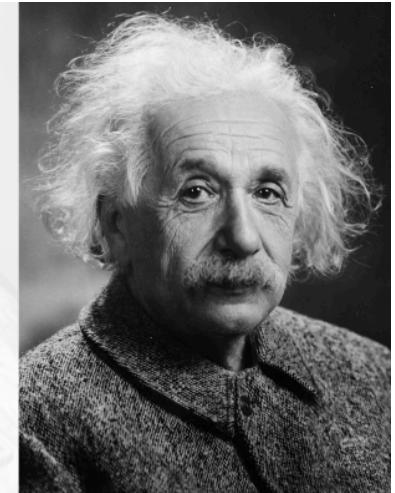
Shopkeeper



Chef



Scientist



# Attention

**Query:** I want to have fun

# Attention

**Query:** I want to have fun

Musician

Lawyer

Shopkeeper

Chef

Scientist

# Attention

**Query:** I want to have fun



Musician

Lawyer

Shopkeeper

Chef

Scientist

# Attention

**Query:** I want to have fun

Musician



Lawyer



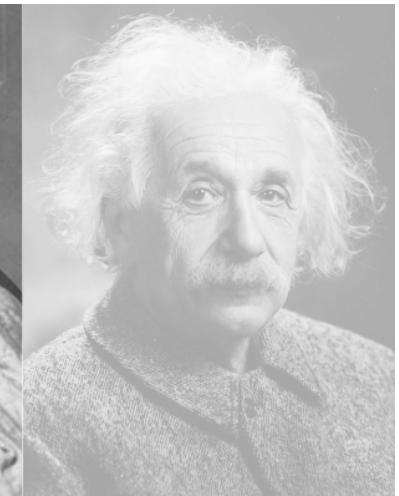
Shopkeeper



Chef



Scientist



How do we represent this mathematically?

# Attention

For each input, we create a key  $k$

# Attention

For each input, we create a key  $k$

$$k_j = \theta_K^\top x_j, \quad k_j \in \mathbb{R}^{d_h}$$

# Attention

For each input, we create a key  $\mathbf{k}$

$$\mathbf{k}_j = \theta_K^\top \mathbf{x}_j, \quad \mathbf{k}_j \in \mathbb{R}^{d_h}$$

Do this for all inputs

# Attention

For each input, we create a key  $\mathbf{k}$

$$\mathbf{k}_j = \theta_K^\top \mathbf{x}_j, \quad \mathbf{k}_j \in \mathbb{R}^{d_h}$$

Do this for all inputs

$$\mathbf{K} = \begin{bmatrix} \mathbf{k}_1 \\ \mathbf{k}_2 \\ \vdots \\ \mathbf{k}_T \end{bmatrix} = \begin{bmatrix} \theta_K^\top \mathbf{x}_1 \\ \theta_K^\top \mathbf{x}_2 \\ \vdots \\ \theta_K^\top \mathbf{x}_T \end{bmatrix}, \quad \mathbf{K} \in \mathbb{R}^{T \times d_h}$$

# Attention

Now, create a query from some  $x_q$

# Attention

Now, create a query from some  $x_q$

$$q = \theta_Q^\top x_q, \quad q \in \mathbb{R}^{d_h}$$

# Attention

Now, create a query from some  $x_q$

$$q = \theta_Q^\top x_q, \quad q \in \mathbb{R}^{d_h}$$

We determine how well a key matches the query with the dot product

# Attention

Now, create a query from some  $\mathbf{x}_q$

$$\mathbf{q} = \boldsymbol{\theta}_Q^\top \mathbf{x}_q, \quad \mathbf{q} \in \mathbb{R}^{d_h}$$

We determine how well a key matches the query with the dot product

$$\mathbf{q}^\top \mathbf{k}_i = (\boldsymbol{\theta}_Q^\top \mathbf{x}_q)^\top (\boldsymbol{\theta}_K^\top \mathbf{x}_i)$$

# Attention

Now, create a query from some  $\mathbf{x}_q$

$$\mathbf{q} = \boldsymbol{\theta}_Q^\top \mathbf{x}_q, \quad \mathbf{q} \in \mathbb{R}^{d_h}$$

We determine how well a key matches the query with the dot product

$$\mathbf{q}^\top \mathbf{k}_i = (\boldsymbol{\theta}_Q^\top \mathbf{x}_q)^\top (\boldsymbol{\theta}_K^\top \mathbf{x}_i)$$

**Question:** What is the shape of  $\mathbf{q}^\top \mathbf{k}_i$ ?

# Attention

Now, create a query from some  $\mathbf{x}_q$

$$\mathbf{q} = \boldsymbol{\theta}_Q^\top \mathbf{x}_q, \quad \mathbf{q} \in \mathbb{R}^{d_h}$$

We determine how well a key matches the query with the dot product

$$\mathbf{q}^\top \mathbf{k}_i = (\boldsymbol{\theta}_Q^\top \mathbf{x}_q)^\top (\boldsymbol{\theta}_K^\top \mathbf{x}_i)$$

**Question:** What is the shape of  $\mathbf{q}^\top \mathbf{k}_i$ ?

**Answer:**  $(1, d_h) \times (d_h, 1) = 1$ , the output is a scalar

# Attention

Now, create a query from some  $\mathbf{x}_q$

$$\mathbf{q} = \boldsymbol{\theta}_Q^\top \mathbf{x}_q, \quad \mathbf{q} \in \mathbb{R}^{d_h}$$

We determine how well a key matches the query with the dot product

$$\mathbf{q}^\top \mathbf{k}_i = (\boldsymbol{\theta}_Q^\top \mathbf{x}_q)^\top (\boldsymbol{\theta}_K^\top \mathbf{x}_i)$$

**Question:** What is the shape of  $\mathbf{q}^\top \mathbf{k}_i$ ?

**Answer:**  $(1, d_h) \times (d_h, 1) = 1$ , the output is a scalar

# Attention

Example:

$$k_i = \theta_K^\top$$



# Attention

Example:

$$k_i = \theta_K^\top$$



$$q = \theta_Q^\top \text{ Musician}$$

# Attention

Example:

$$k_i = \theta_K^\top$$



$$q = \theta_Q^\top \text{ Musician}$$

$$q^\top k_i = (\theta_Q^\top \text{ Musician})^\top \left( \theta_K^\top \begin{array}{c} \\ \text{Musician} \\ \end{array} \right) = 5.6$$

# Attention

Example:

$$k_i = \theta_K^\top$$



$$q = \theta_Q^\top \text{ Musician}$$

$$q^\top k_i = (\theta_Q^\top \text{ Musician})^\top \left( \theta_K^\top \begin{array}{c} \\ \text{Taylor Swift portrait} \\ \end{array} \right) = 5.6$$

Large attention!

# Attention

Example:

$$k_i = \theta_K^\top$$



# Attention

Example:

$$k_i = \theta_K^\top$$



$$q = \theta_Q^\top \text{ Mathematician}$$

# Attention

Example:

$$k_i = \theta_K^\top$$



$$q = \theta_Q^\top \text{ Mathematician}$$

$$q^\top k_i = (\theta_Q^\top \text{ Mathematician})^\top \left( \theta_K^\top \begin{array}{c} \text{Mathematician} \\ \text{Taylor Swift} \end{array} \right) = -4.5$$

# Attention

Example:

$$k_i = \theta_K^\top$$



$$q = \theta_Q^\top \text{ Mathematician}$$

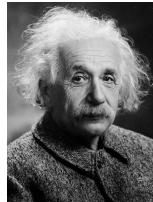
$$q^\top k_i = (\theta_Q^\top \text{ Mathematician})^\top \begin{pmatrix} \theta_K^\top \\ \text{Mathematician} \end{pmatrix} = -4.5$$

Small attention!

# Attention

Example:

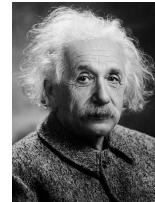
$$k_i = \theta_K^\top$$



# Attention

Example:

$$k_i = \theta_K^\top$$

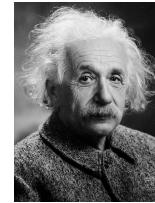


$$q = \theta_Q^\top \text{ Mathematician}$$

# Attention

Example:

$$k_i = \theta_K^\top$$



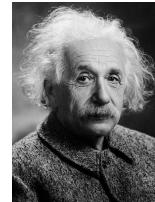
$$q = \theta_Q^\top \text{ Mathematician}$$

$$q^\top k_i = (\theta_Q^\top \text{ Mathematician})^\top \begin{pmatrix} \theta_K^\top \\ \text{Mathematician} \end{pmatrix} = 6.1$$

# Attention

Example:

$$k_i = \theta_K^\top$$



$$q = \theta_Q^\top \text{ Mathematician}$$

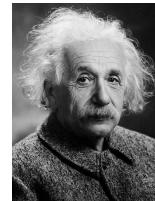
$$q^\top k_i = (\theta_Q^\top \text{ Mathematician})^\top \begin{pmatrix} \theta_K^\top \\ \text{Mathematician} \end{pmatrix} = 6.1$$

Large attention!

# Attention

Example:

$$k_i = \theta_K^\top$$



$$q = \theta_Q^\top \text{ Mathematician}$$

$$q^\top k_i = (\theta_Q^\top \text{ Mathematician})^\top \begin{pmatrix} \theta_K^\top \\ \text{Albert Einstein portrait} \end{pmatrix} = 6.1$$

Large attention!

Remember, there are multiple inputs to pay attention to

# Attention

We compute attention for each input

$$q^\top K = q^\top \begin{bmatrix} k_1 \\ k_2 \\ \vdots \\ k_T \end{bmatrix} = \begin{bmatrix} (\theta_Q^\top x_q)^\top (\theta_K^\top x_1) \\ (\theta_Q^\top x_q)^\top (\theta_K^\top x_2) \\ \vdots \\ (\theta_Q^\top x_q)^\top (\theta_K^\top x_T) \end{bmatrix}$$

# Attention

Do not forget normalization! Only pay attention to important matches

# Attention

Do not forget normalization! Only pay attention to important matches

$$\text{softmax}(\mathbf{q}^\top \mathbf{K}) = \text{softmax}\left(\mathbf{q}^\top \begin{bmatrix} \mathbf{k}_1 \\ \mathbf{k}_2 \\ \vdots \\ \mathbf{k}_T \end{bmatrix}\right) = \text{softmax}\left(\begin{bmatrix} (\theta_Q^\top \mathbf{x}_q)^\top (\theta_K^\top \mathbf{x}_1) \\ (\theta_Q^\top \mathbf{x}_q)^\top (\theta_K^\top \mathbf{x}_2) \\ \vdots \\ (\theta_Q^\top \mathbf{x}_q)^\top (\theta_K^\top \mathbf{x}_T) \end{bmatrix}\right)$$

# Attention

Do not forget normalization! Only pay attention to important matches

$$\text{softmax}(\mathbf{q}^\top \mathbf{K}) = \text{softmax}\left(\mathbf{q}^\top \begin{bmatrix} \mathbf{k}_1 \\ \mathbf{k}_2 \\ \vdots \\ \mathbf{k}_T \end{bmatrix}\right) = \text{softmax}\left(\begin{bmatrix} (\theta_Q^\top \mathbf{x}_q)^\top (\theta_K^\top \mathbf{x}_1) \\ (\theta_Q^\top \mathbf{x}_q)^\top (\theta_K^\top \mathbf{x}_2) \\ \vdots \\ (\theta_Q^\top \mathbf{x}_q)^\top (\theta_K^\top \mathbf{x}_T) \end{bmatrix}\right)$$

We call this **dot-product attention**

# Attention

**Query:** Which person will help me on my exam?

# Attention

**Query:** Which person will help me on my exam?



$$q^\top k_1$$

$$q^\top k_2$$

$$q^\top k_3$$

$$q^\top k_4$$

$$q^\top k_5$$

# Attention

**Query:** Which person will help me on my exam?



$$q^\top k_1$$

$$q^\top k_2$$

$$q^\top k_3$$

$$q^\top k_4$$

$$q^\top k_5$$

softmax

# Attention

**Query:** Which person will help me on my exam?



$$q^\top k_1$$

$$q^\top k_2$$

$$q^\top k_3$$

$$q^\top k_4$$

$$q^\top k_5$$

softmax

0.01

0.10

0.02

0.03

0.84

# Attention

**Query:** Which person will help me on my exam?



$$q^\top k_1$$

$$q^\top k_2$$

$$q^\top k_3$$

$$q^\top k_4$$

$$q^\top k_5$$

softmax

0.01

0.10

0.02

0.03

0.84

# Attention



$$(q^\top k_1)$$

$$q^\top k_2$$

$$q^\top k_3$$

$$q^\top k_4$$

$$q^\top k_5$$

$$\cdot \theta_V^\top x_1$$

$$\cdot \theta_V^\top x_2$$

$$\cdot \theta_V^\top x_3$$

$$\cdot \theta_V^\top x_4$$

$$\cdot \theta_V^\top x_5$$

# Attention



$$(q^\top k_1)$$

$$q^\top k_2$$

$$q^\top k_3$$

$$q^\top k_4$$

$$q^\top k_5$$

$$\cdot \theta_V^\top x_1$$

$$\cdot \theta_V^\top x_2$$

$$\cdot \theta_V^\top x_3$$

$$\cdot \theta_V^\top x_4$$

$$\cdot \theta_V^\top x_5$$

## Self attention