

Attention and Transformers

CISC 7026: Introduction to Deep Learning

University of Macau

Agenda

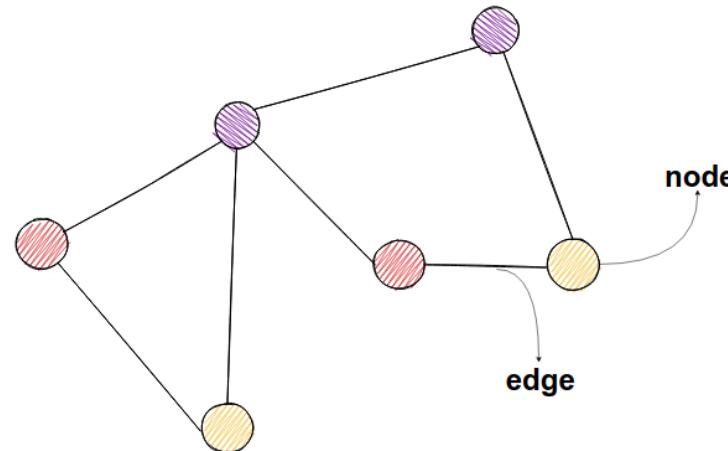
1. GNN Review
2. VAE Review and Coding
3. Attention
4. Keys and Queries
5. Transformer
6. Positional Encoding
7. Coding

Agenda

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

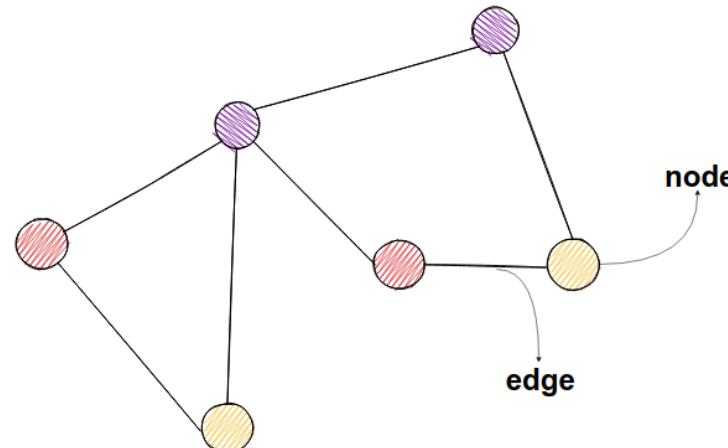
Graph is a structure of nodes (vertices) and edges

Graph is a structure of nodes (vertices) and edges



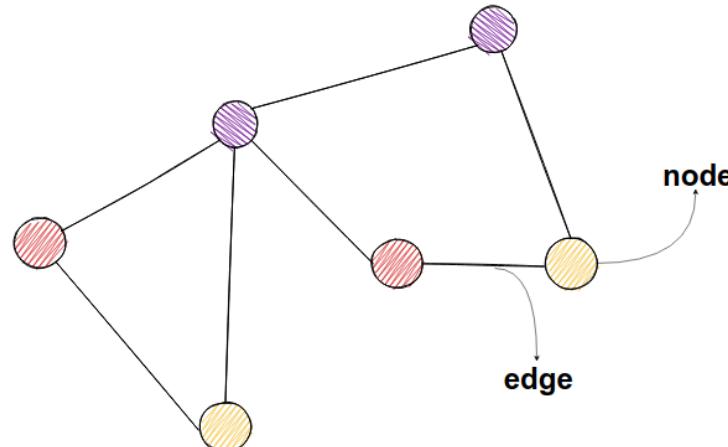
GNN Review

Graph is a structure of nodes (vertices) and edges



A **node** is a vector of information

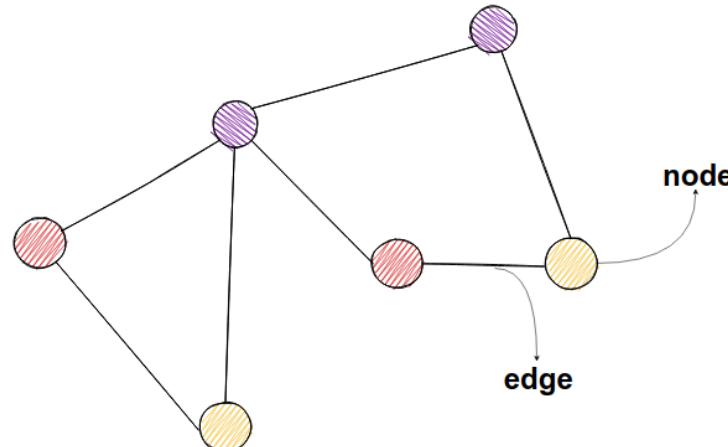
Graph is a structure of nodes (vertices) and edges



A **node** is a vector of information

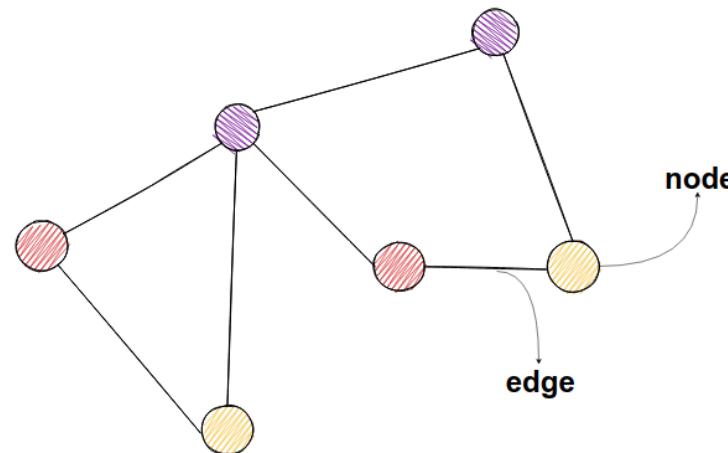
An **edge** connects two nodes

Graph is a structure of nodes (vertices) and edges



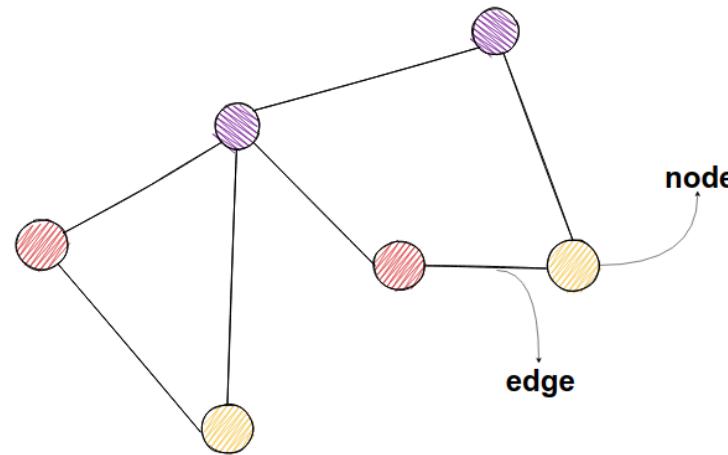
A **node** is a vector of information

An **edge** connects two nodes



A **node** is a vector of information

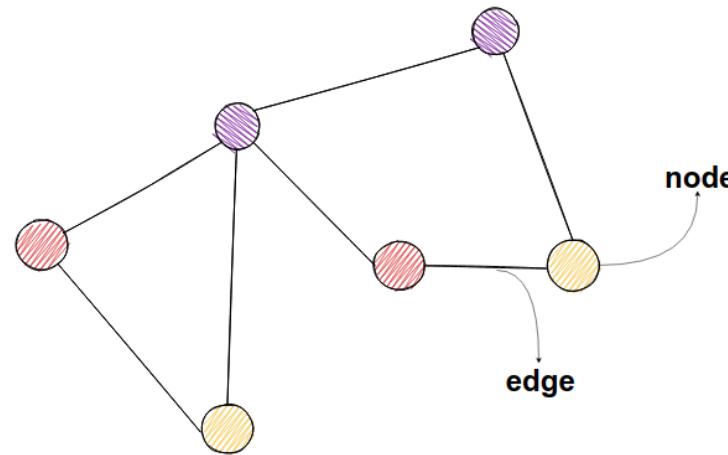
An **edge** connects two nodes



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** $N(i)$ contains all neighbors of node i

Let us think of graphs as **signals**

Let us think of graphs as **signals**

Question: Where did we see signals before?

Let us think of graphs as **signals**

Question: Where did we see signals before?

Rather than time t or space u, v , graphs are a function of index i

Let us think of graphs as **signals**

Question: Where did we see signals before?

Rather than time t or space u, v , graphs are a function of index i

$$\text{Node } i \quad \mathbf{x}(i) \in \mathbb{R}^{d_x}; \quad i \in 1, \dots, T$$

Let us think of graphs as **signals**

Question: Where did we see signals before?

Rather than time t or space u, v , graphs are a function of index i

Node i $\mathbf{x}(i) \in \mathbb{R}^{d_x}; \quad i \in 1, \dots, T$

Neighborhood of i $\mathbf{N}(i) = \begin{bmatrix} i \\ j \\ k \\ \vdots \end{bmatrix}; \quad \mathbf{N}(i) \in \mathcal{P}(i); \quad i \in 1, \dots, T$

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

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

For a node i , the graph convolution layer is:

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

For a node i , the graph convolution layer is:

$$f(\mathbf{x}, \mathbf{N}, \boldsymbol{\theta})(i) = \sigma \left(\sum_{j \in \mathbf{N}(i)} \boldsymbol{\theta}^\top \overline{\mathbf{x}}(j) \right)$$

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

For a node i , the graph convolution layer is:

$$f(\mathbf{x}, \mathbf{N}, \boldsymbol{\theta})(i) = \sigma \left(\sum_{j \in \mathbf{N}(i)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j) \right)$$

Combine information from the neighbors of $\mathbf{x}(i)$

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

For a node i , the graph convolution layer is:

$$f(\mathbf{x}, \mathbf{N}, \boldsymbol{\theta})(i) = \sigma \left(\sum_{j \in \mathbf{N}(i)} \boldsymbol{\theta}^\top \overline{\mathbf{x}}(j) \right)$$

Combine information from the neighbors of $\mathbf{x}(i)$

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

Apply graph convolution over all nodes in the graph

Apply graph convolution over all nodes in the graph

$$f(\mathbf{x}, \mathbf{N}, \boldsymbol{\theta}) = \begin{bmatrix} f(\mathbf{x}, \mathbf{N}, \boldsymbol{\theta})(1) \\ f(\mathbf{x}, \mathbf{N}, \boldsymbol{\theta})(2) \\ \vdots \\ f(\mathbf{x}, \mathbf{N}, \boldsymbol{\theta})(T) \end{bmatrix} = \begin{bmatrix} \sigma\left(\sum_{j \in \mathbf{N}(1)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \\ \sigma\left(\sum_{j \in \mathbf{N}(2)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \\ \vdots \\ \sigma\left(\sum_{j \in \mathbf{N}(T)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \end{bmatrix}$$

Apply graph convolution over all nodes in the graph

$$f(\mathbf{x}, \mathbf{N}, \boldsymbol{\theta}) = \begin{bmatrix} f(\mathbf{x}, \mathbf{N}, \boldsymbol{\theta})(1) \\ f(\mathbf{x}, \mathbf{N}, \boldsymbol{\theta})(2) \\ \vdots \\ f(\mathbf{x}, \mathbf{N}, \boldsymbol{\theta})(T) \end{bmatrix} = \begin{bmatrix} \sigma\left(\sum_{j \in \mathbf{N}(1)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \\ \sigma\left(\sum_{j \in \mathbf{N}(2)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \\ \vdots \\ \sigma\left(\sum_{j \in \mathbf{N}(T)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \end{bmatrix}$$

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

Standard 1D convolution

$$\begin{bmatrix} \sigma\left(\sum_{j=1}^k \theta^\top \bar{x}(j)\right) \\ \sigma\left(\sum_{j=2}^{k+1} \theta^\top \bar{x}(j)\right) \\ \vdots \\ \sigma\left(\sum_{j=T-k}^T \theta^\top \bar{x}(j)\right) \end{bmatrix}$$

Standard 1D convolution

$$\begin{bmatrix} \sigma\left(\sum_{j=1}^k \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \\ \sigma\left(\sum_{j=2}^{k+1} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \\ \vdots \\ \sigma\left(\sum_{j=T-k}^T \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \end{bmatrix}$$

Graph convolution

$$\begin{bmatrix} \sigma\left(\sum_{j \in \mathcal{N}(1)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \\ \sigma\left(\sum_{j \in \mathcal{N}(2)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \\ \vdots \\ \sigma\left(\sum_{j \in \mathcal{N}(T)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \end{bmatrix}$$

Standard 1D convolution

$$\begin{bmatrix} \sigma\left(\sum_{j=1}^k \theta^\top \bar{x}(j)\right) \\ \sigma\left(\sum_{j=2}^{k+1} \theta^\top \bar{x}(j)\right) \\ \vdots \\ \sigma\left(\sum_{j=T-k}^T \theta^\top \bar{x}(j)\right) \end{bmatrix}$$

Graph convolution

$$\begin{bmatrix} \sigma\left(\sum_{j \in N(1)} \theta^\top \bar{x}(j)\right) \\ \sigma\left(\sum_{j \in N(2)} \theta^\top \bar{x}(j)\right) \\ \vdots \\ \sigma\left(\sum_{j \in N(T)} \theta^\top \bar{x}(j)\right) \end{bmatrix}$$

Question: What is the output size of standard convolution?

Standard 1D convolution

$$\begin{bmatrix} \sigma\left(\sum_{j=1}^k \theta^\top \bar{x}(j)\right) \\ \sigma\left(\sum_{j=2}^{k+1} \theta^\top \bar{x}(j)\right) \\ \vdots \\ \sigma\left(\sum_{j=T-k}^T \theta^\top \bar{x}(j)\right) \end{bmatrix}$$

Graph convolution

$$\begin{bmatrix} \sigma\left(\sum_{j \in N(1)} \theta^\top \bar{x}(j)\right) \\ \sigma\left(\sum_{j \in N(2)} \theta^\top \bar{x}(j)\right) \\ \vdots \\ \sigma\left(\sum_{j \in N(T)} \theta^\top \bar{x}(j)\right) \end{bmatrix}$$

Question: What is the output size of standard convolution?

Answer: $(T - k - 1) \times d_h$

Standard 1D convolution

$$\begin{bmatrix} \sigma\left(\sum_{j=1}^k \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \\ \sigma\left(\sum_{j=2}^{k+1} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \\ \vdots \\ \sigma\left(\sum_{j=T-k}^T \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \end{bmatrix}$$

Graph convolution

$$\begin{bmatrix} \sigma\left(\sum_{j \in \mathcal{N}(1)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \\ \sigma\left(\sum_{j \in \mathcal{N}(2)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \\ \vdots \\ \sigma\left(\sum_{j \in \mathcal{N}(T)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \end{bmatrix}$$

Standard 1D convolution

$$\begin{bmatrix} \sigma\left(\sum_{j=1}^k \theta^\top \bar{x}(j)\right) \\ \sigma\left(\sum_{j=2}^{k+1} \theta^\top \bar{x}(j)\right) \\ \vdots \\ \sigma\left(\sum_{j=T-k}^T \theta^\top \bar{x}(j)\right) \end{bmatrix}$$

Graph convolution

$$\begin{bmatrix} \sigma\left(\sum_{j \in N(1)} \theta^\top \bar{x}(j)\right) \\ \sigma\left(\sum_{j \in N(2)} \theta^\top \bar{x}(j)\right) \\ \vdots \\ \sigma\left(\sum_{j \in N(T)} \theta^\top \bar{x}(j)\right) \end{bmatrix}$$

Question: What is the output size of graph convolution?

Standard 1D convolution

$$\begin{bmatrix} \sigma\left(\sum_{j=1}^k \theta^\top \bar{x}(j)\right) \\ \sigma\left(\sum_{j=2}^{k+1} \theta^\top \bar{x}(j)\right) \\ \vdots \\ \sigma\left(\sum_{j=T-k}^T \theta^\top \bar{x}(j)\right) \end{bmatrix}$$

Graph convolution

$$\begin{bmatrix} \sigma\left(\sum_{j \in N(1)} \theta^\top \bar{x}(j)\right) \\ \sigma\left(\sum_{j \in N(2)} \theta^\top \bar{x}(j)\right) \\ \vdots \\ \sigma\left(\sum_{j \in N(T)} \theta^\top \bar{x}(j)\right) \end{bmatrix}$$

Question: What is the output size of graph convolution?

Answer: $T \times d_h$

We can use pooling with graph convolutions too

$$\text{SumPool} \left(\begin{bmatrix} \sigma\left(\sum_{j \in \mathcal{N}(1)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \\ \sigma\left(\sum_{j \in \mathcal{N}(2)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \\ \vdots \\ \sigma\left(\sum_{j \in \mathcal{N}(T)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) \end{bmatrix} \right) = \sigma\left(\sum_{j \in \mathcal{N}(1)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) + \sigma\left(\sum_{j \in \mathcal{N}(2)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right) + \dots + \sigma\left(\sum_{j \in \mathcal{N}(T)} \boldsymbol{\theta}^\top \bar{\mathbf{x}}(j)\right)$$

Agenda

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

Agenda

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

VAE Review and Coding

We can use autoencoders as **generative models**

VAE Review and Coding

We can use autoencoders as **generative models**

A generative model learns the structure of data

VAE Review and Coding

We can use autoencoders as **generative models**

A generative model learns the structure of data

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

VAE Review and Coding

We can use autoencoders as **generative models**

A generative model learns the structure of data

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

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

VAE Review and Coding

We can use autoencoders as **generative models**

A generative model learns the structure of data

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

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

VAE Review and Coding

We can use autoencoders as **generative models**

A generative model learns the structure of data

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

- Train on face dataset, generate **new** faces
- Train on book dataset, write a **new** book
- Train on human knowledge, create **new** knowledge

VAE Review and Coding

We can use autoencoders as **generative models**

A generative model learns the structure of data

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

- Train on face dataset, generate **new** faces
- Train on book dataset, write a **new** book
- Train on human knowledge, create **new** knowledge

Generative models learn the dataset **distribution** $P(x)$; $x \in X$

VAE Review and Coding

Generative models learn the dataset **distribution** $P(x)$; $x \in X$

Virtually **all** generative methods learn the dataset distribution

VAE Review and Coding

Generative models learn the dataset **distribution** $P(x)$; $x \in X$

Virtually **all** generative methods learn the dataset distribution

- Variational autoencoders

VAE Review and Coding

Generative models learn the dataset **distribution** $P(x)$; $x \in X$

Virtually **all** generative methods learn the dataset distribution

- Variational autoencoders
- Diffusion models

VAE Review and Coding

Generative models learn the dataset **distribution** $P(x)$; $x \in X$

Virtually **all** generative methods learn the dataset distribution

- Variational autoencoders
- Diffusion models
- Generative adversarial networks

VAE Review and Coding

Generative models learn the dataset **distribution** $P(x)$; $x \in X$

Virtually **all** generative methods learn the dataset distribution

- Variational autoencoders
- Diffusion models
- Generative adversarial networks

If you like generative models, you should study Bayesian statistics

VAE Review and Coding

Generative models learn the dataset **distribution** $P(x)$; $x \in X$

Virtually **all** generative methods learn the dataset distribution

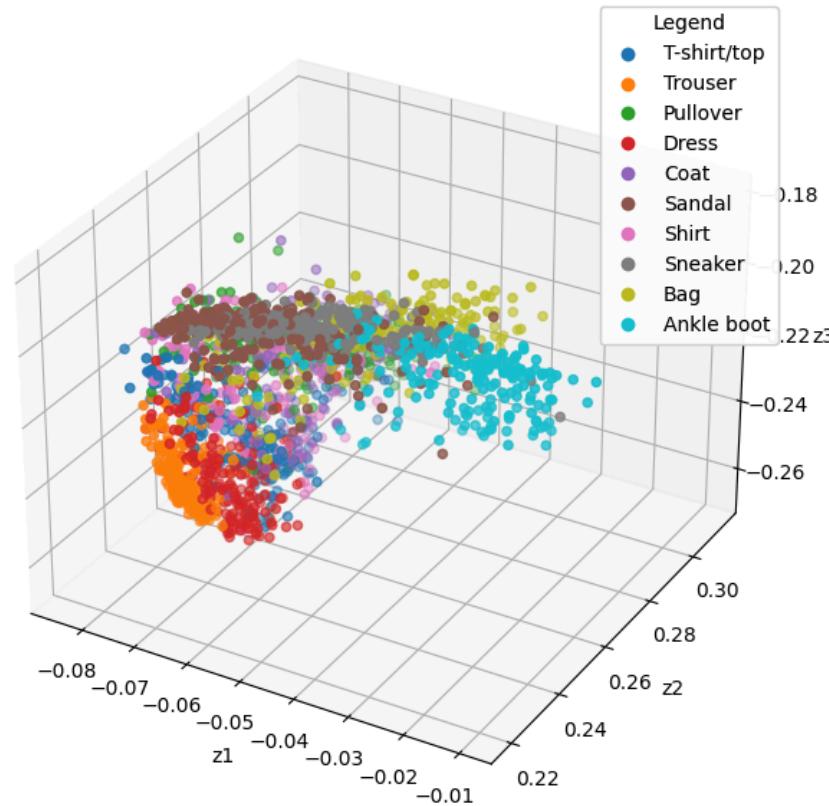
- Variational autoencoders
- Diffusion models
- Generative adversarial networks

If you like generative models, you should study Bayesian statistics

Back to the variational autoencoder

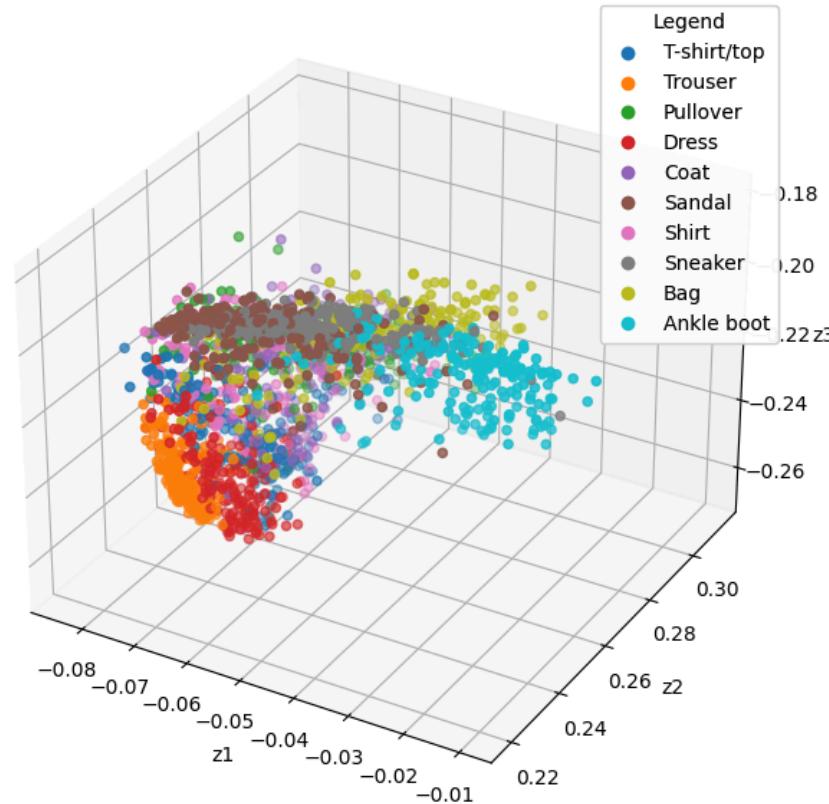
VAE Review and Coding

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

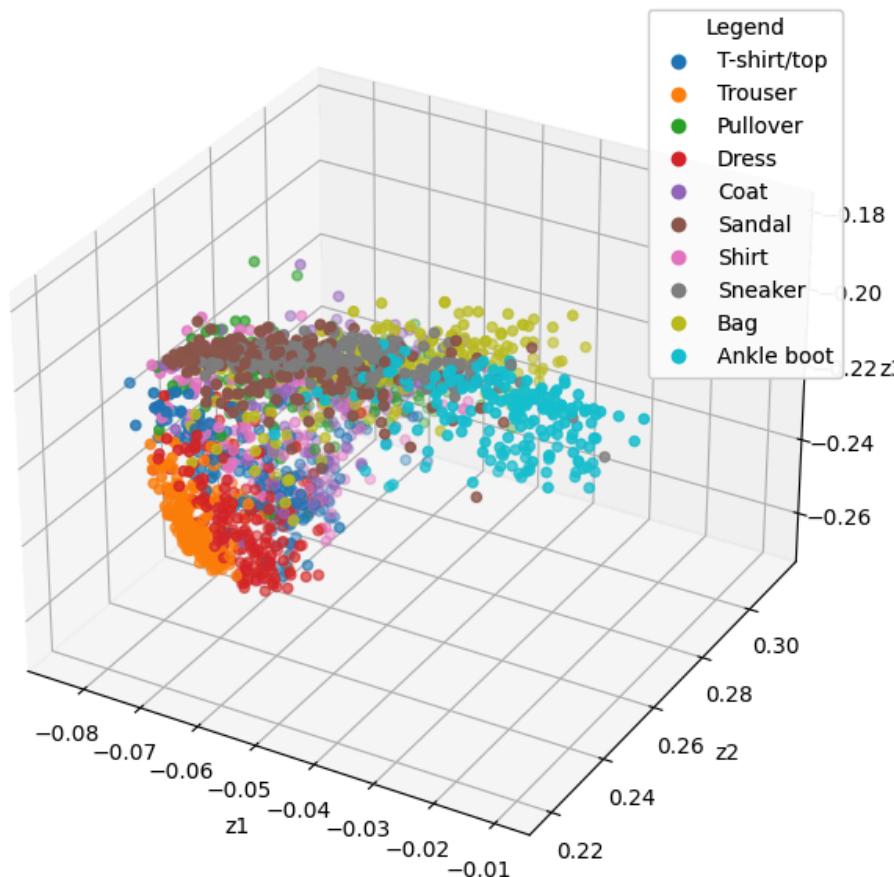


VAE Review and Coding

What happens if we decode a new point?

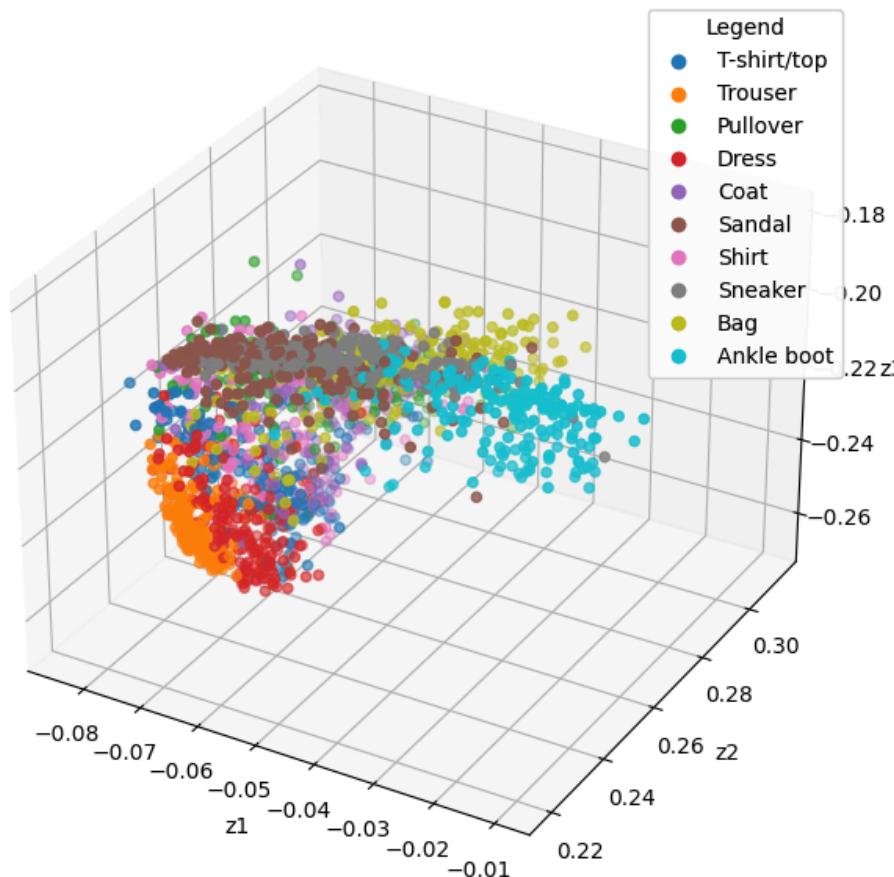


VAE Review and Coding



Autoencoder generative model:

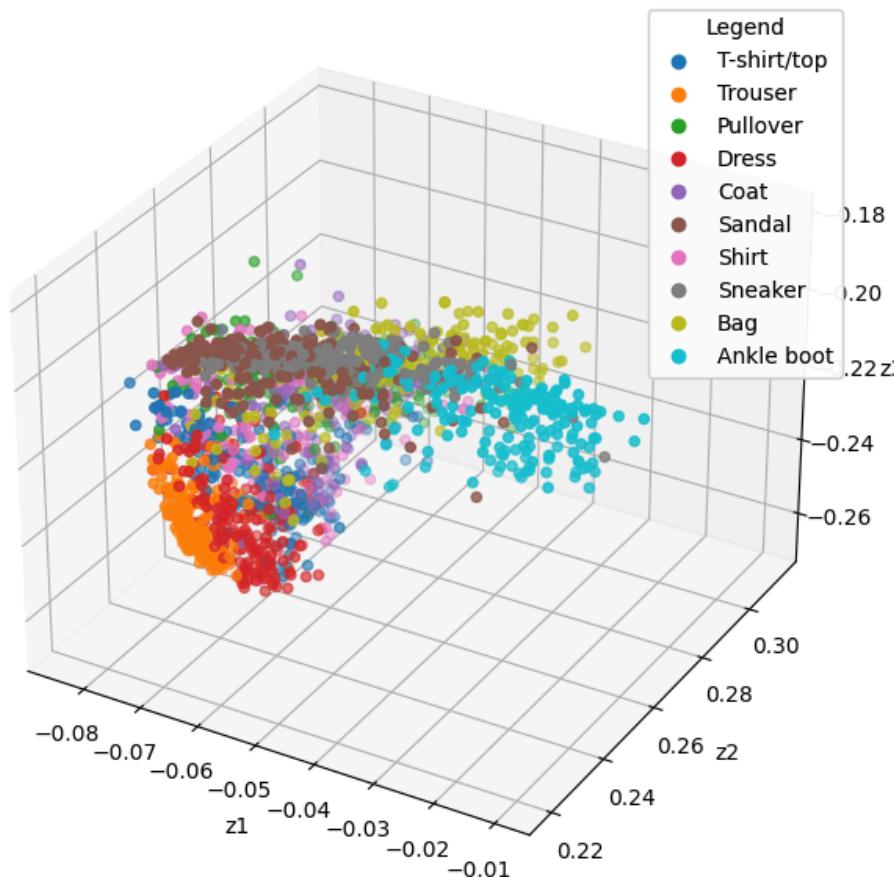
VAE Review and Coding



Autoencoder generative model:

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

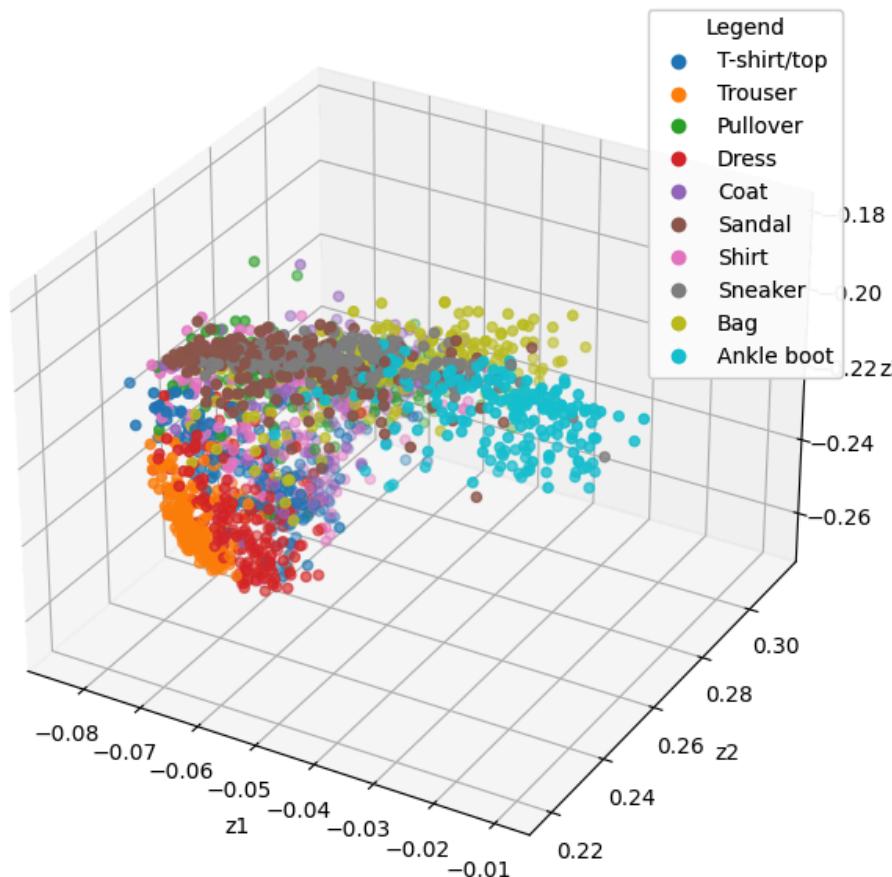
VAE Review and Coding



Autoencoder generative model:

1. Encode $\begin{bmatrix} \mathbf{x}_{[1]} \\ \vdots \\ \mathbf{x}_{[n]} \end{bmatrix}$ into $\begin{bmatrix} \mathbf{z}_{[1]} \\ \vdots \\ \mathbf{z}_{[n]} \end{bmatrix}$
2. Pick a point $\mathbf{z}_{[k]}$

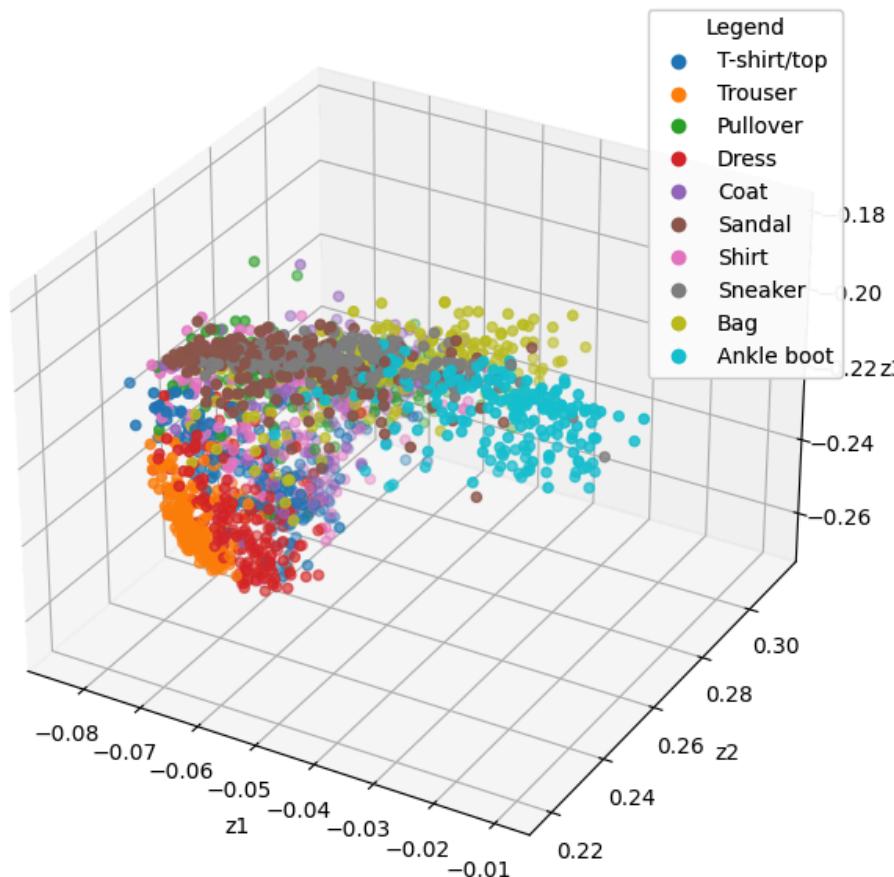
VAE Review and Coding



Autoencoder generative model:

1. Encode $\begin{bmatrix} \mathbf{x}_{[1]} \\ \vdots \\ \mathbf{x}_{[n]} \end{bmatrix}$ into $\begin{bmatrix} \mathbf{z}_{[1]} \\ \vdots \\ \mathbf{z}_{[n]} \end{bmatrix}$
2. Pick a point $\mathbf{z}_{[k]}$
3. Add some noise $\mathbf{z}_{\text{new}} = \mathbf{z}_{[k]} + \boldsymbol{\varepsilon}$

VAE Review and Coding



Autoencoder generative model:

1. Encode $\begin{bmatrix} \mathbf{x}_{[1]} \\ \vdots \\ \mathbf{x}_{[n]} \end{bmatrix}$ into $\begin{bmatrix} \mathbf{z}_{[1]} \\ \vdots \\ \mathbf{z}_{[n]} \end{bmatrix}$
2. Pick a point $\mathbf{z}_{[k]}$
3. Add some noise $\mathbf{z}_{\text{new}} = \mathbf{z}_{[k]} + \boldsymbol{\varepsilon}$
4. Decode \mathbf{z}_{new} into \mathbf{x}_{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**

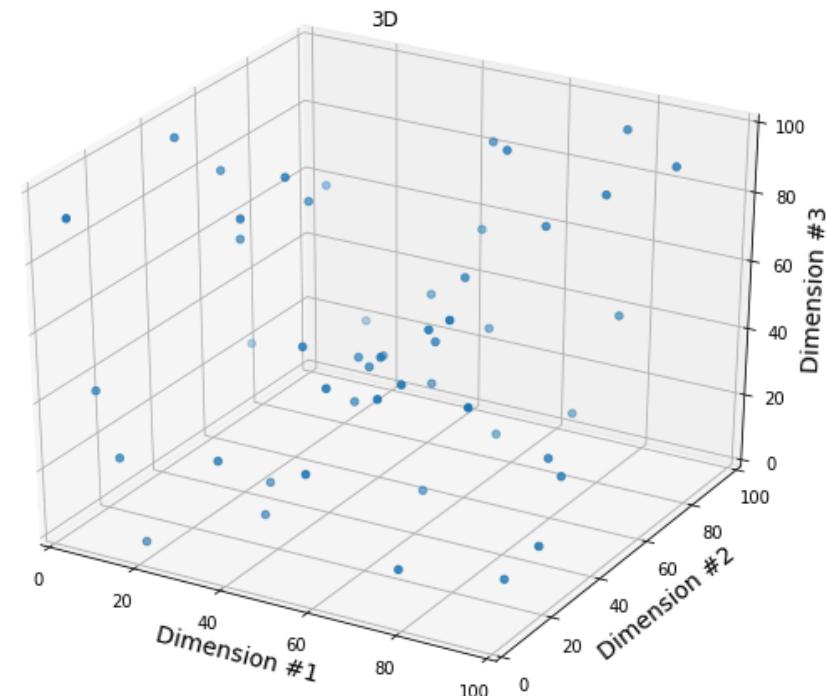
But there is a problem, the **curse
of dimensionality**

As d_z increases, points move
further and further apart

VAE Review and Coding

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

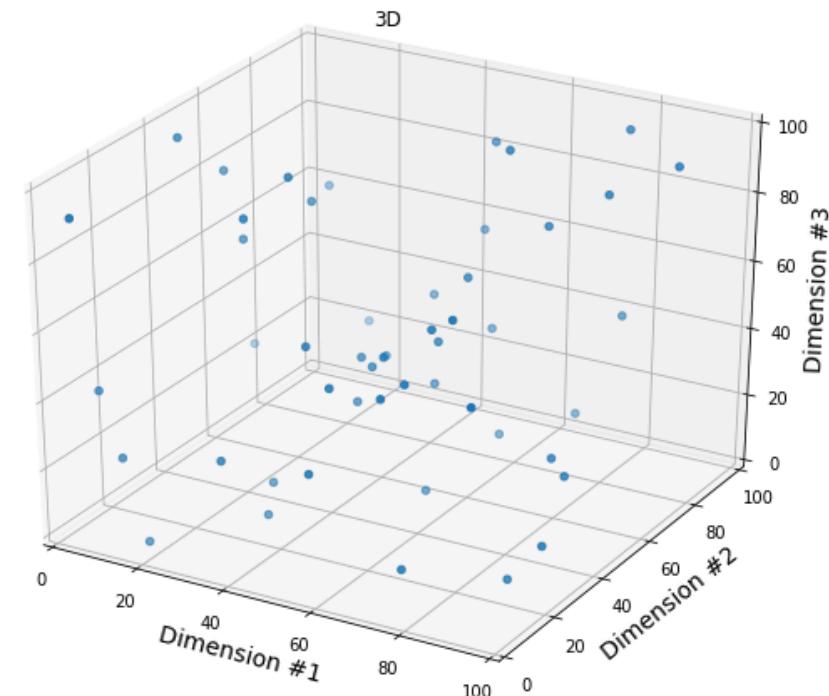
As d_z increases, points move further and further apart



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

VAE Review and Coding

Variational autoencoders (VAEs) make $z_{[1]}, \dots z_{[n]}$ normally distributed

This keeps the point close together

VAE Review and Coding

Variational autoencoders (VAEs) make $z_{[1]}, \dots z_{[n]}$ normally distributed

This keeps the point close together

We can sample new points $z_{\text{new}} \sim N(\mathbf{0}, \mathbf{1})$

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 θ 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 θ that best models the distribution of possible faces

Large $P(x; \theta)$



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



VAE Review and Coding

Given a prior distribution

$$P(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbf{1})$$

VAE Review and Coding

Given a prior distribution

$$P(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbf{1})$$

We want to approximate the dataset distribution

$$P(\mathbf{x})$$

VAE Review and Coding

Given a prior distribution

$$P(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbf{1})$$

We want to approximate the dataset distribution

$$P(\mathbf{x})$$

By approximating the marginal likelihood (from Bayes rule)

$$P(\mathbf{x}; \boldsymbol{\theta}) = \int_{\mathbf{z}} P(\mathbf{x} \mid \mathbf{z}; \boldsymbol{\theta}) P(\mathbf{z}) \, d\mathbf{z}$$

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

$x :$



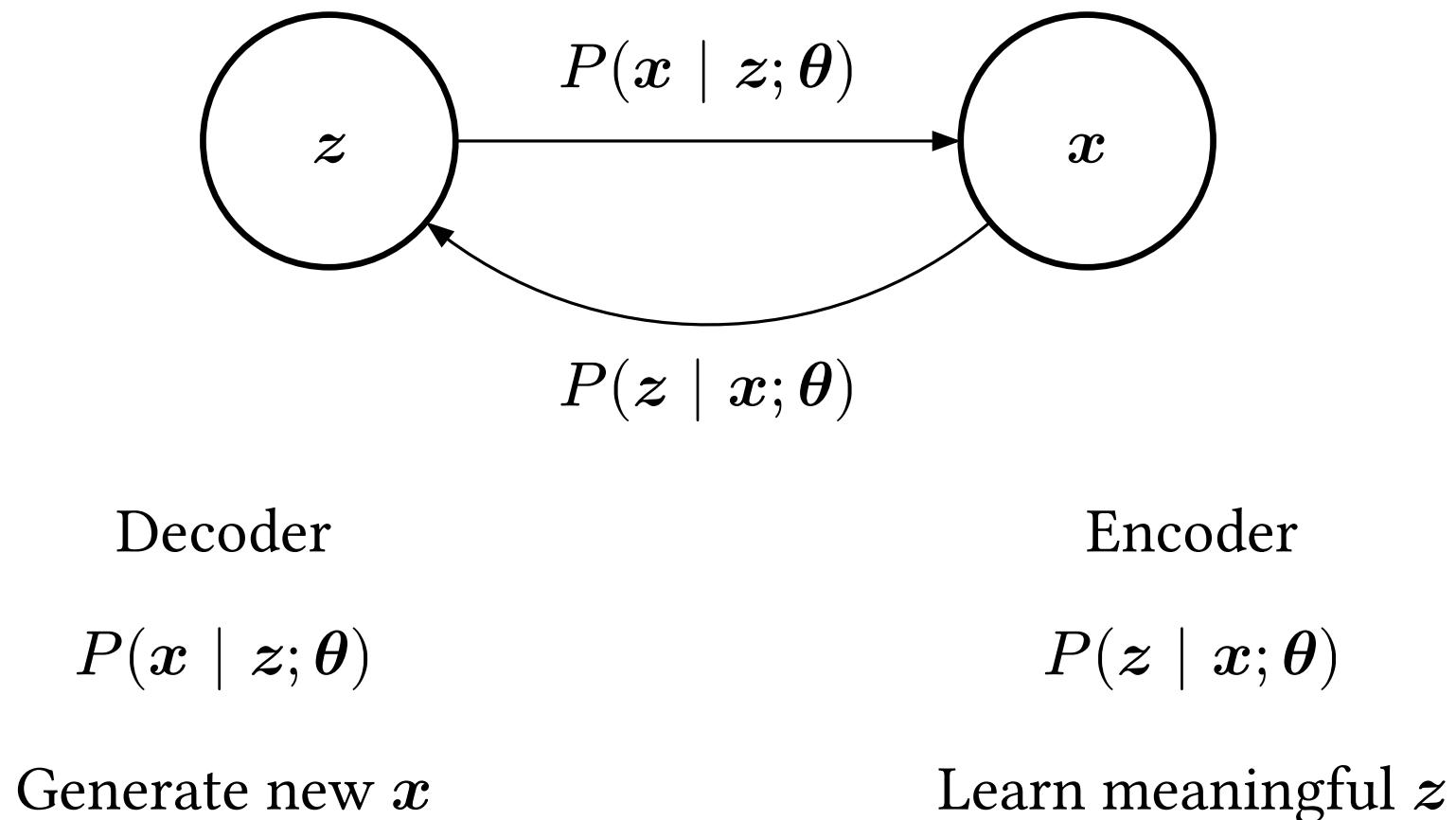
$z : [\text{woman brown hair (frown} \mid \text{smile)}]$

VAE Review and Coding

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

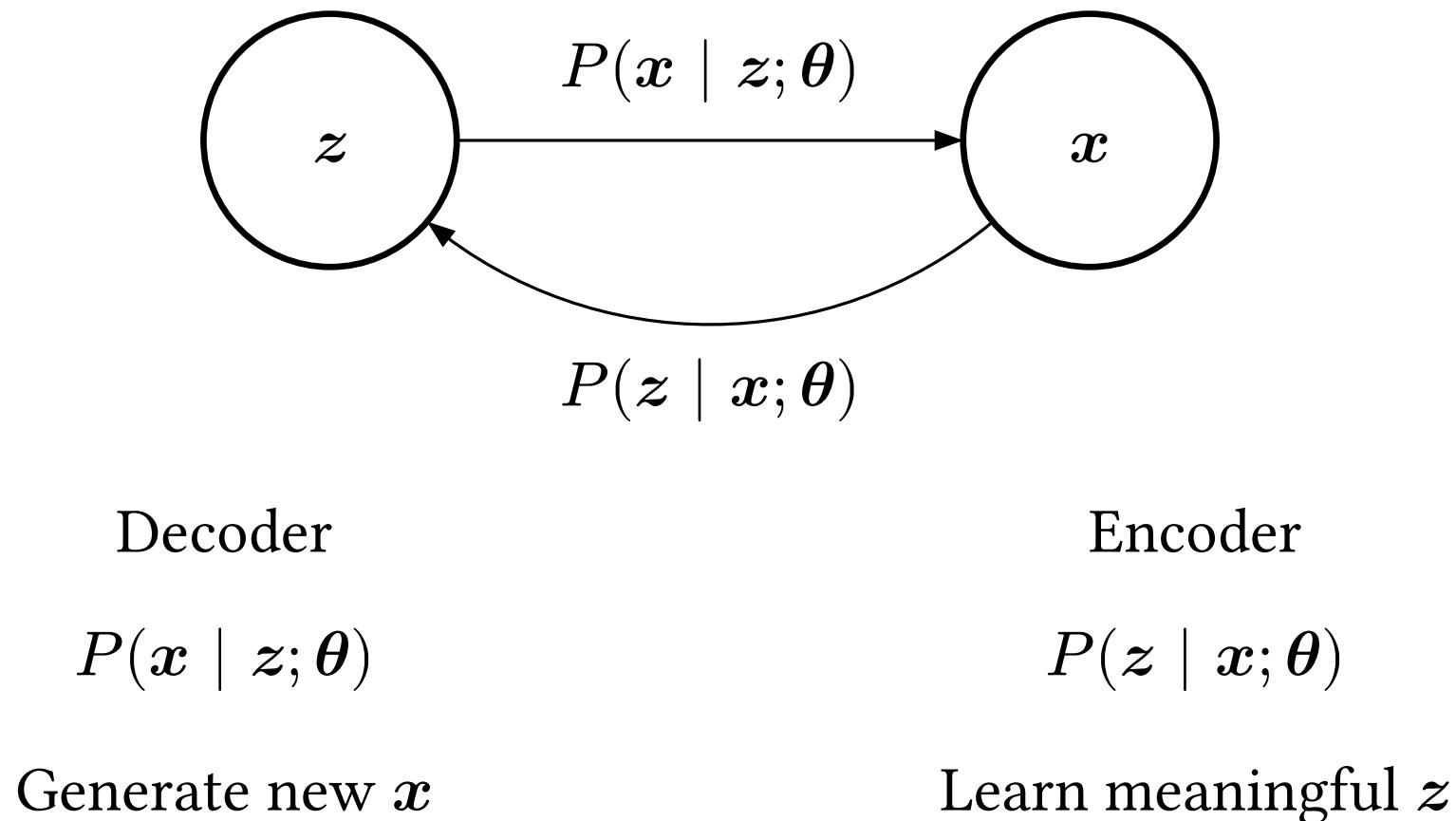
VAE Review and Coding

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



VAE Review and Coding

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



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(log_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

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

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 ΔZ 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 ΔZ but decoder input is Z

Solution: Sample a vector z from the distribution ΔZ

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 θ that best explains the dataset distribution

VAE Review and Coding

Now, all we must do is find θ 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 θ 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 θ 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, beta, 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 + beta * kl
```

VAE Review and Coding

https://colab.research.google.com/drive/1UyR_W6NDIujaJXYlHZh6O3NfaCAMscpH#scrollTo=nmyQ8aE2pSbb

VAE Review and Coding

https://colab.research.google.com/drive/1UyR_W6NDIujaJXYlHZh6O3NfaCAMscpH#scrollTo=nmyQ8aE2pSbb

<https://openai.com/index/glow/>

Agenda

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

Agenda

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

Attention

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

Attention

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

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

Attention

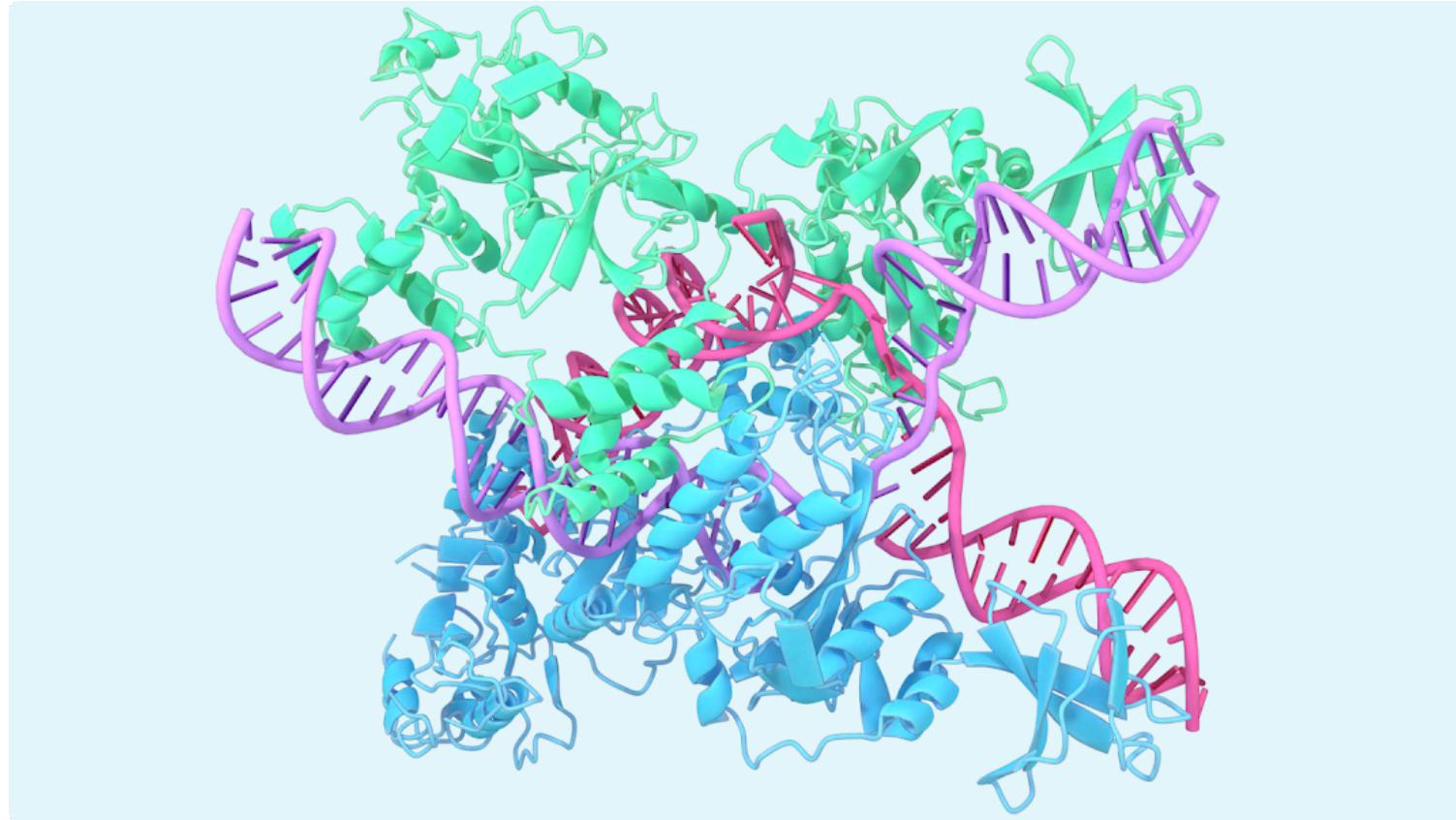
Attention and transformers are the “hottest” topic in deep learning

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

Let’s review some projects based on attention

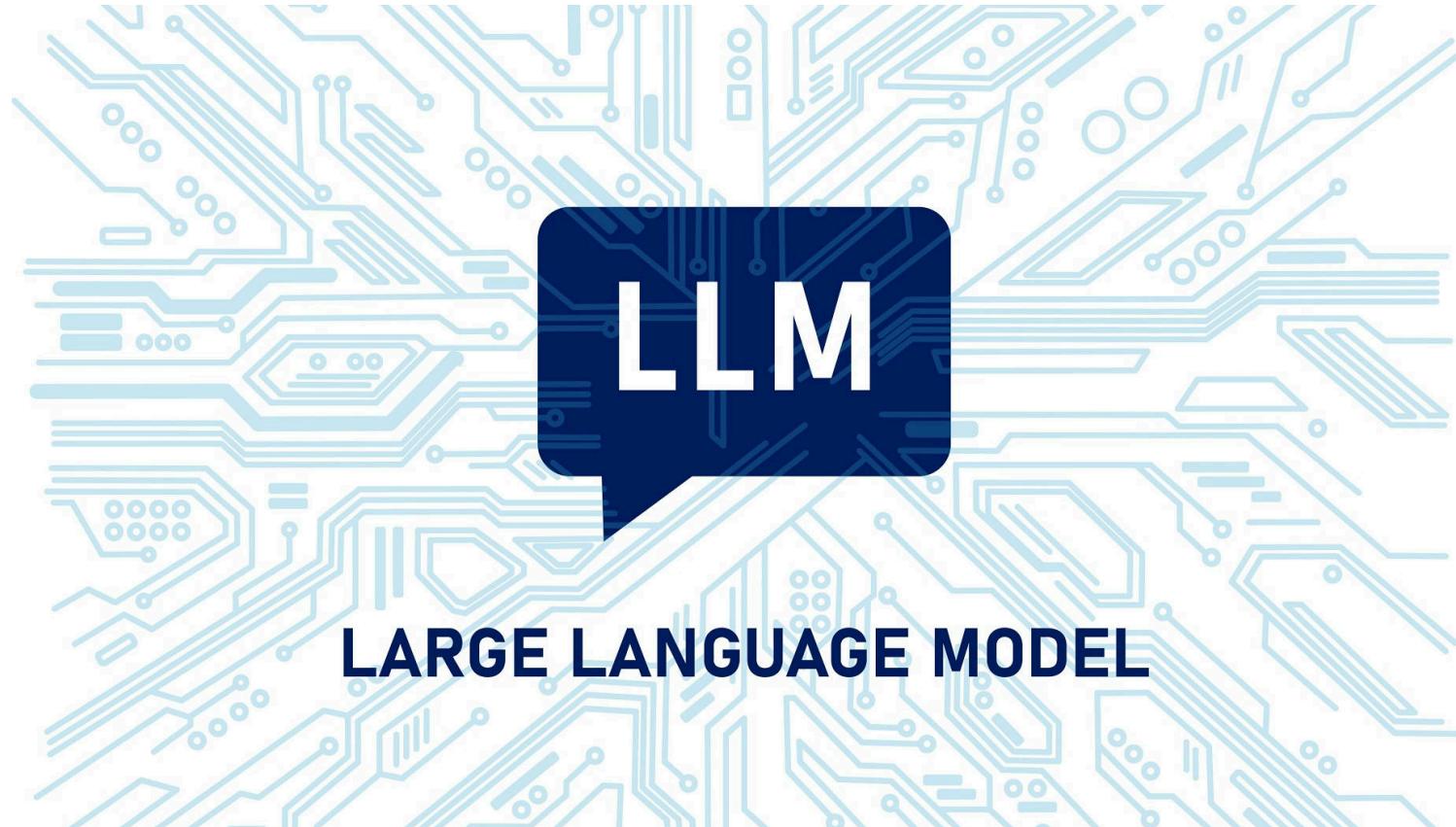
Attention

AlphaFold (Nobel prize)



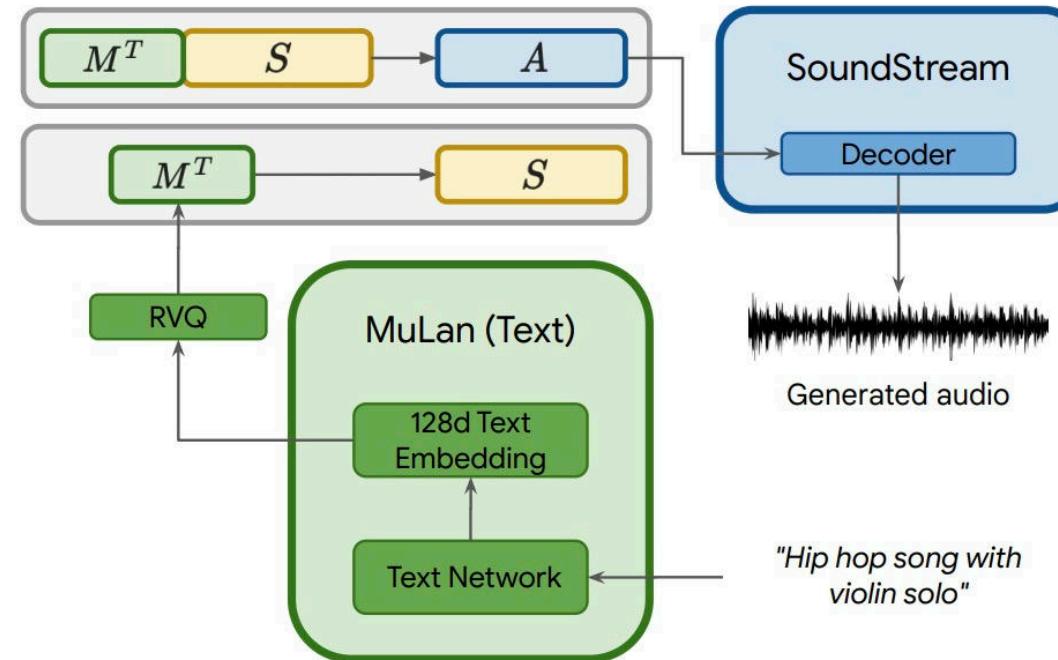
Attention

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



Attention

MusicTransformer, MuLan



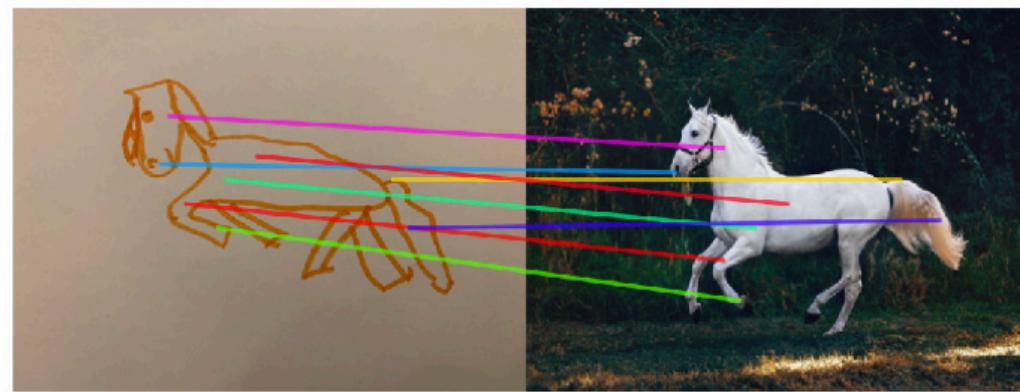
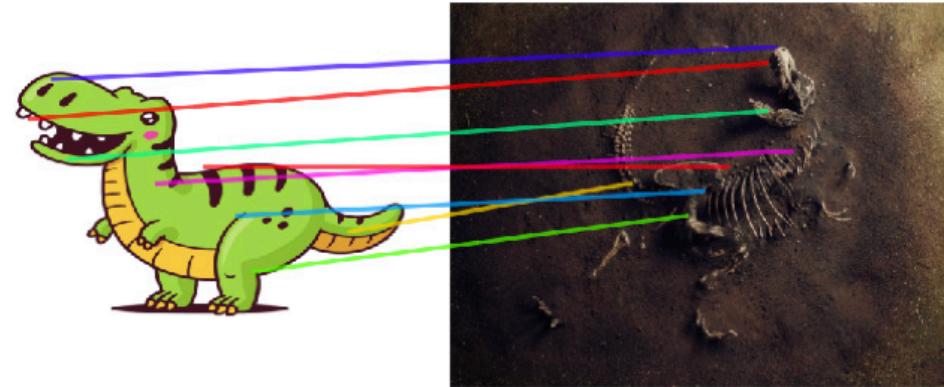
Attention

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

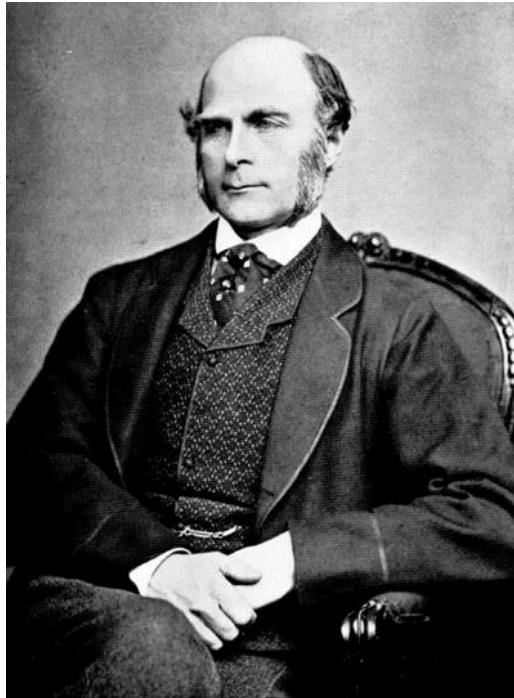
All these models are **transformers**

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

We can derive attention from composite memory

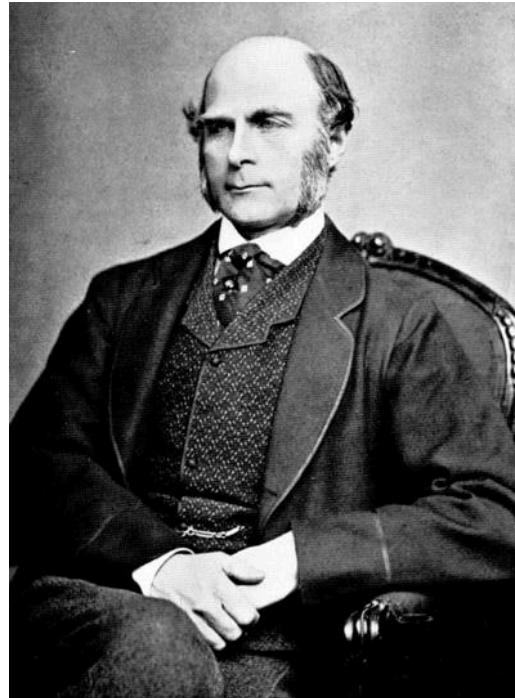
Attention

Francis Galton (1822-1911)
photo composite memory

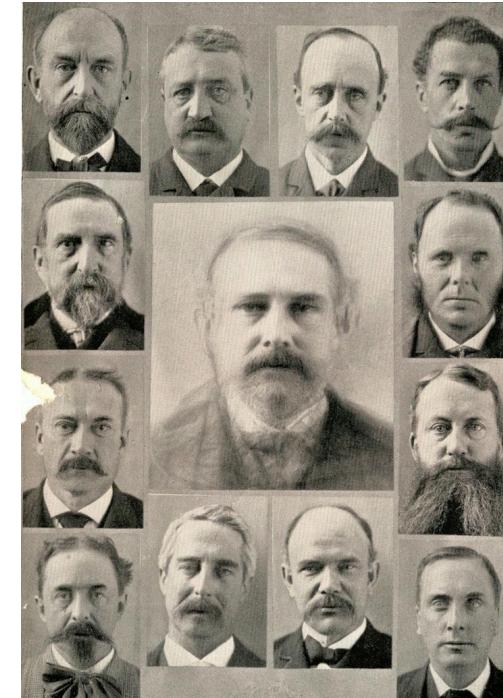


Attention

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

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

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 remember everything

Attention

Limited space, cannot remember everything

Introduced forgetting term $\gamma \in [0, 1]$

Attention

Limited space, cannot remember everything

Introduced forgetting term $\gamma \in [0, 1]$

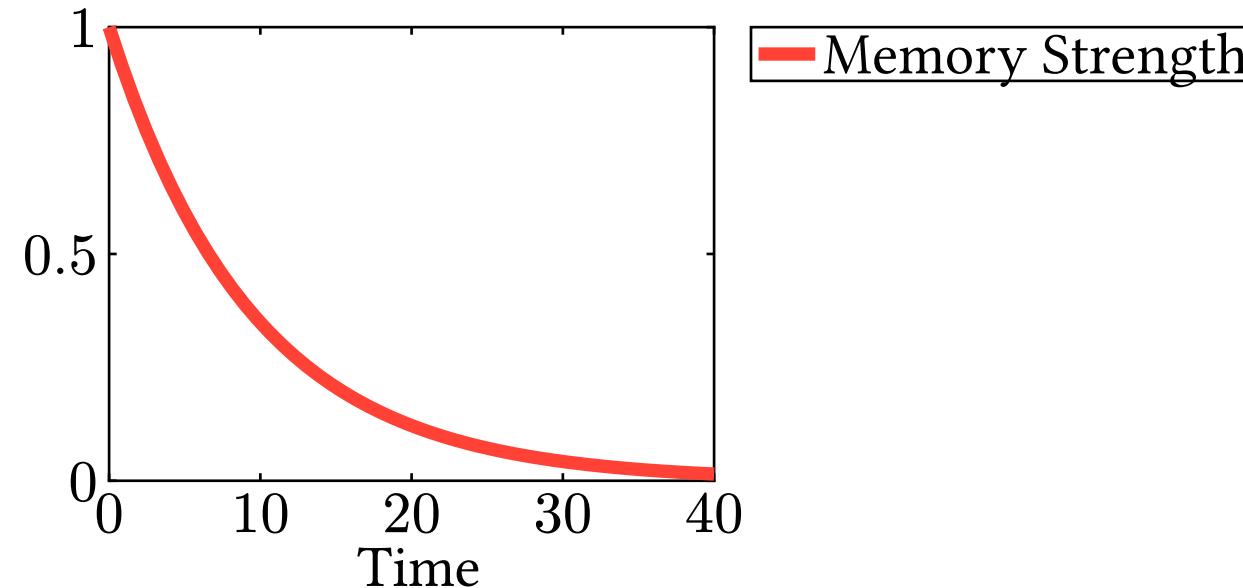
$$f(\mathbf{x}, \boldsymbol{\theta}) = \sum_{i=1}^T \gamma^{T-i} \cdot \boldsymbol{\theta}^\top \bar{\mathbf{x}}_i$$

Attention

Limited space, cannot remember everything

Introduced forgetting term $\gamma \in [0, 1]$

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

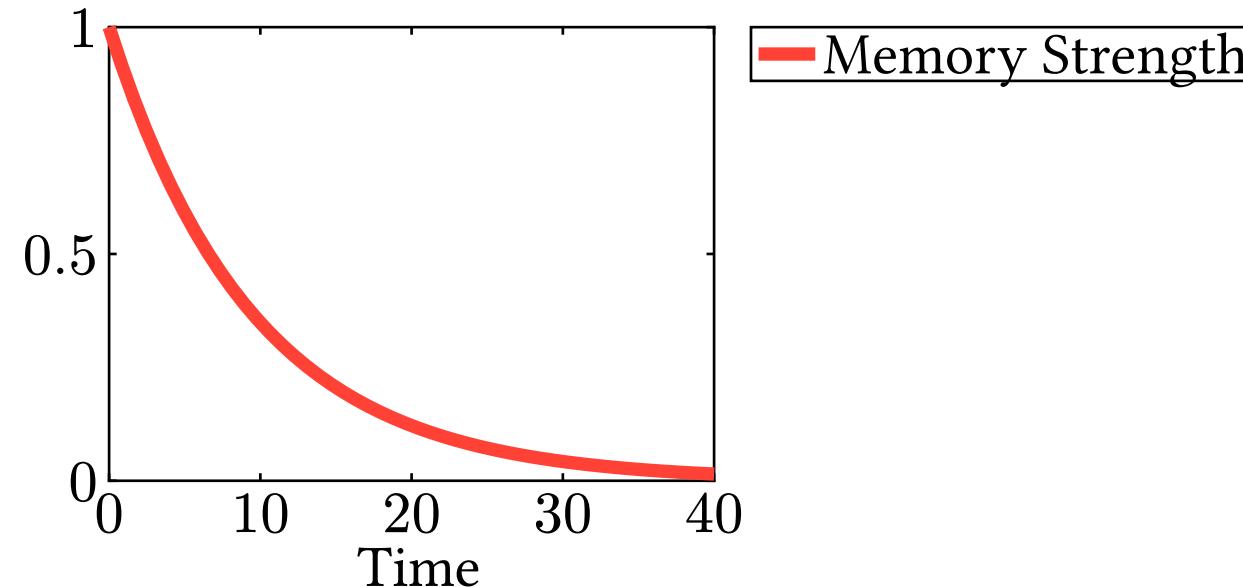


Attention

Limited space, cannot remember everything

Introduced forgetting term $\gamma \in [0, 1]$

$$f(x, \theta) = \sum_{i=1}^T \gamma^{T-i} \cdot \theta^\top \bar{x}_i$$



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

Attention

Example: We attend a party in 1850s

Attention

Example: We attend a party in 1850s

We talk with many people at this party

Attention

Example: We attend a party in 1850s

We talk with many people at this party



Attention

Example: We attend a party in 1850s

We talk with many people at this party



10 PM

Attention

Example: We attend a party in 1850s

We talk with many people at this party



10 PM

11 PM

Attention

Example: We attend a party in 1850s

We talk with many people at this party



10 PM

11 PM

12 AM

Attention

Example: We attend a party in 1850s

We talk with many people at this party



10 PM

11 PM

12 AM

1 AM

Attention

According to forgetting, the memories should fade with time

Attention

According to forgetting, the memories should fade with time



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

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

Any questions before moving on?

Attention

Consider another party, with one more guest

Attention

Consider another party, with one more guest



Attention

Consider another party, with one more guest



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

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

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

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

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

Attention

Consider another party, with one more guest



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

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

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

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

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

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!

Question: Would you forget meeting Taylor Swift?

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

Attention

Our model of memory is incomplete

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

Important memories persist longer than unimportant memories

Attention

Our model of memory is incomplete

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

Important memories persist longer than unimportant memories

We will **pay more attention** to certain memories

What does human memory actually look like?



What does human memory actually look like?



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

What does human memory actually look like?



$$1.0 \cdot \theta^\top \bar{x}_1 \quad 0.1 \cdot \theta^\top \bar{x}_2$$

What does human memory actually look like?



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

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

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

What does human memory actually look like?



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

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

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

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

What does human memory actually look like?



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

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

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

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

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

What does human memory actually look like?



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

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

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

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

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

Question: How can we achieve this forgetting?

Attention

In our composite model, forgetting is a function of time

Attention

In our composite model, forgetting is a function of time

Question: Any forgetting mechanism that is not a function of time?

Attention

In our composite model, forgetting is a function of time

Question: Any forgetting mechanism that is not a function of time?

Answer: Forgetting in recurrent neural network is function of input!

Attention

In our composite model, forgetting is a function of time

Question: Any forgetting mechanism that is not a function of time?

Answer: Forgetting in recurrent neural network is function of input!

$$f_{\text{forget}}(x, \theta) = \sigma(\theta_\lambda^\top \bar{x})$$

Attention

In our composite model, forgetting is a function of time

Question: Any forgetting mechanism that is not a function of time?

Answer: Forgetting in recurrent neural network is function of input!

$$f_{\text{forget}}(x, \theta) = \sigma(\theta_\lambda^\top \bar{x})$$

$$f(h, x, \theta) = f_{\text{forget}}(x, \theta) \odot h + \theta_x^\top \bar{x}$$

Attention

First, write our forgetting function with slightly different notation

Attention

First, write our forgetting function with slightly different notation

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

Attention

First, write our forgetting function with slightly different notation

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

Question: Shape of $\lambda(x, \theta_\lambda)$?

Attention

First, write our forgetting function with slightly different notation

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

Question: Shape of $\lambda(x, \theta_\lambda)$?

Answer: Scalar!

Attention

First, write our forgetting function with slightly different notation

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

Question: Shape of $\lambda(x, \theta_\lambda)$?

Answer: Scalar!

Then, write our composite memory model with forgetting

Attention

First, write our forgetting function with slightly different notation

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

Question: Shape of $\lambda(x, \theta_\lambda)$?

Answer: Scalar!

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 with slightly different notation

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

Question: Shape of $\lambda(x, \theta_\lambda)$?

Answer: Scalar!

Then, write our composite memory model with forgetting

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

Only pay attention to important inputs

Attention

First, write our forgetting function with slightly different notation

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

Question: Shape of $\lambda(x, \theta_\lambda)$?

Answer: Scalar!

Then, write our composite memory model with forgetting

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

Only pay attention to important inputs

Attention

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

Attention

We can use this simple form of attention to remember 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

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

Question: What do the images look like now?



Attention

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 finite (human) attention span

Attention

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

For example, normalize attention to sum to one

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

Attention

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

For example, normalize attention to sum to one

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

Now the model must choose who to remember!

Attention

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

For example, normalize attention to sum to one

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

Now the model must choose who to remember!

Question: How can we ensure that the attention sums to one?

Attention

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

For example, normalize attention to sum to one

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

Now the model must choose who to remember!

Question: How can we ensure that the attention sums to one?

Answer: Softmax!

Attention

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

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

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

Attention



Attention



$$\lambda \left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_5 \end{bmatrix}, \theta_\lambda \right)_1 \cdot \theta^\top \overline{\mathbf{x}}_1$$

Attention



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

Attention



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

Attention



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

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



Attention



$$0.70 \cdot \theta^\top \bar{x}_1$$

Attention



$$0.70 \cdot \theta^\top \bar{x}_1 - 0.04 \cdot \theta^\top \bar{x}_2$$

Attention



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

Attention



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

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

$$\lambda \left(\begin{bmatrix} x_1 \\ \vdots \\ x_T \end{bmatrix}, \theta_\lambda \right)_i = \frac{\exp(\theta_\lambda^\top \bar{x}_i)}{\sum_{j=1}^T \exp(\theta_\lambda^\top \bar{x}_j)}$$

Attention

$$\lambda \left(\begin{bmatrix} x_1 \\ \vdots \\ x_T \end{bmatrix}, \theta_\lambda \right)_i = \frac{\exp(\theta_\lambda^\top \bar{x}_i)}{\sum_{j=1}^T \exp(\theta_\lambda^\top \bar{x}_j)}$$

Compute attention for all inputs at once

Attention

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

Compute attention for all inputs at once

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

Attention

This is a simple form of attention

Attention

This is a simple form of attention

Next, we will investigate the attention used in transformers

Agenda

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

Agenda

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

Keys and Queries

The modern form of attention behaves like a database

Keys and Queries

The modern form of attention behaves like a database

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

Keys and Queries

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

Keys and Queries

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



Keys and Queries

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

Keys and Queries

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

Keys and Queries

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

Keys and Queries

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

Keys and Queries

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

Keys and Queries

We can search through our keys using a **query**

Keys and Queries

We can search through our keys using a **query**

Query: Which person will help me on my exam?

Keys and Queries

We can search through our keys using a **query**

Query: Which person will help me on my exam?



Keys and Queries

We can search through our keys using a **query**

Query: Which person will help me on my exam?



Musician

Lawyer

Shopkeeper

Chef

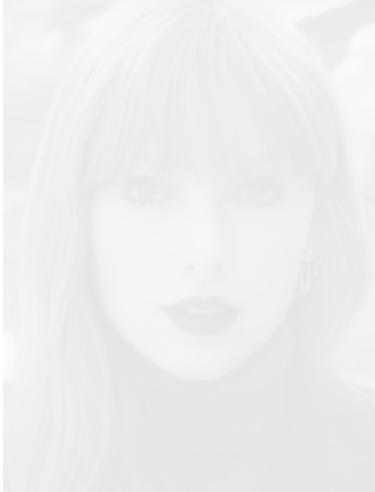
Scientist

Keys and Queries

We can search through our keys using a **query**

Query: Which person will help me on my exam?

Musician



Lawyer



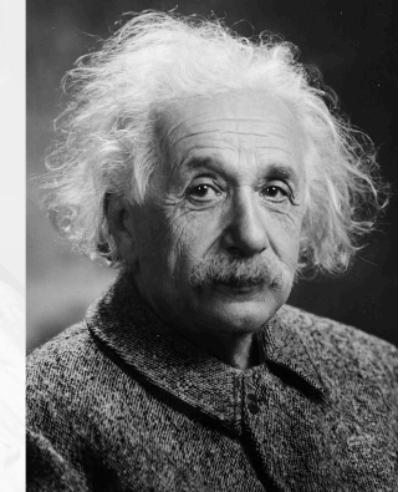
Shopkeeper



Chef



Scientist



Keys and Queries

Query: I want to have fun

Keys and Queries

Query: I want to have fun



Keys and Queries

Query: I want to have fun



Musician

Lawyer

Shopkeeper

Chef

Scientist

Keys and Queries

Query: I want to have fun

Musician



Lawyer



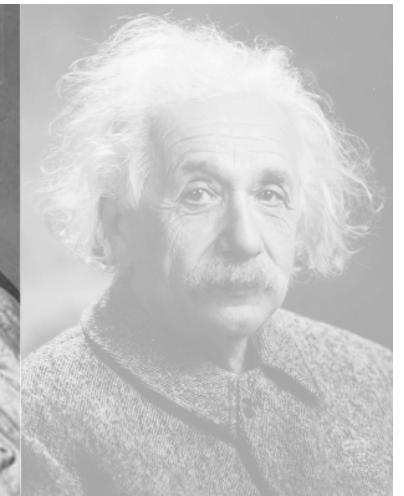
Shopkeeper



Chef



Scientist



Keys and Queries

Query: I want to have fun

Musician



Lawyer



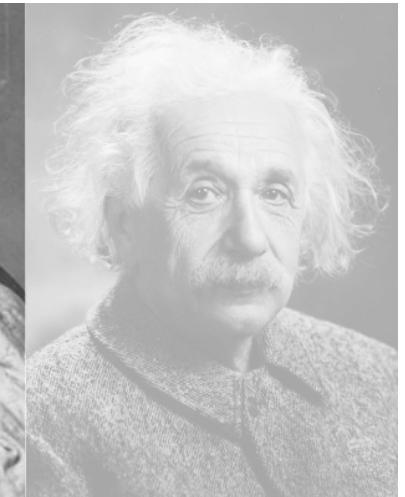
Shopkeeper



Chef



Scientist



How do we represent keys and queries mathematically?

Keys and Queries

For each input, we create a key k

Keys and Queries

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

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

Keys and Queries

For each input, we create a key k

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

Create keys for all inputs

Keys and Queries

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

$$\mathbf{k}_j = \boldsymbol{\theta}_K^\top \mathbf{x}_j, \quad \boldsymbol{\theta}_K \in \mathbb{R}^{d_x \times d_h}, \quad \mathbf{k}_j \in \mathbb{R}^{d_h}$$

Create keys for all inputs

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

Keys and Queries

Now, create a query from some x_q

Keys and Queries

Now, create a query from some x_q

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

Keys and Queries

Now, create a query from some x_q

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

To determine if a key and query match, we will take the dot product

Keys and Queries

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

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

To determine if a key and query match, we will take the dot product

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

Keys and Queries

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

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

To determine if a key and query match, we will take 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 $\mathbf{q}^\top \mathbf{k}_i$?

Keys and Queries

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

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

To determine if a key and query match, we will take 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

Keys and Queries

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

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

To determine if a key and query match, we will take 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

Large dot product \Rightarrow match! Small dot product \Rightarrow no match.

Keys and Queries

Example:

Keys and Queries

Example:

$$k_i = \theta_K^\top$$



Keys and Queries

Example:

$$k_i = \theta_K^\top$$



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

Keys and Queries

Example:

$$k_i = \theta_K^\top$$



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

$$q^\top k_i = (\theta_Q^\top \text{ Musician})^\top \left(\theta_K^\top \begin{array}{c} \\ \text{Musician} \\ \end{array} \right) = 100$$

Keys and Queries

Example:

$$k_i = \theta_K^\top$$



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

$$q^\top k_i = (\theta_Q^\top \text{ Musician})^\top \left(\theta_K^\top \begin{array}{c} \text{Taylor Swift portrait} \\ \vdots \end{array} \right) = 100$$

Large attention!

Keys and Queries

Example:

$$k_i = \theta_K^\top$$



Keys and Queries

Example:

$$k_i = \theta_K^\top$$



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

Keys and Queries

Example:

$$k_i = \theta_K^\top$$



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

$$q^\top k_i = (\theta_Q^\top \text{ Mathematician})^\top \left(\theta_K^\top \begin{array}{c} \text{Taylor Swift portrait} \\ \vdots \end{array} \right) = -50$$

Keys and Queries

Example:

$$k_i = \theta_K^\top$$



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

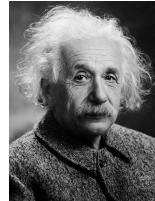
$$q^\top k_i = (\theta_Q^\top \text{ Mathematician})^\top \left(\theta_K^\top \begin{array}{c} \text{Taylor Swift portrait} \\ \vdots \end{array} \right) = -50$$

Small attention!

Keys and Queries

Example:

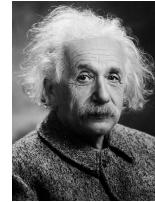
$$k_i = \theta_K^\top$$



Keys and Queries

Example:

$$k_i = \theta_K^\top$$

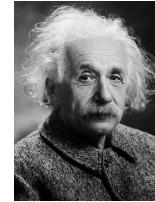


$$q = \theta_Q^\top \text{ Mathematician}$$

Keys and Queries

Example:

$$k_i = \theta_K^\top$$



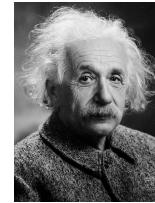
$$q = \theta_Q^\top \text{ Mathematician}$$

$$q^\top k_i = (\theta_Q^\top \text{ Mathematician})^\top \left(\theta_K^\top \begin{array}{c} \text{Mathematician} \\ \text{Albert Einstein} \end{array} \right) = 90$$

Keys and Queries

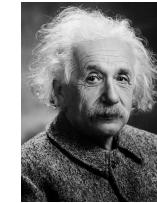
Example:

$$k_i = \theta_K^\top$$



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

$$q^\top k_i = (\theta_Q^\top \text{ Mathematician})^\top \left(\theta_K^\top \right) = 90$$

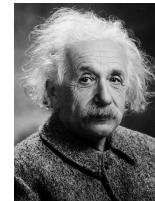


Large attention!

Keys and Queries

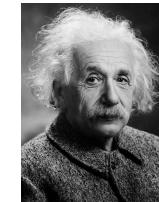
Example:

$$k_i = \theta_K^\top$$



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

$$q^\top k_i = (\theta_Q^\top \text{ Mathematician})^\top \left(\theta_K^\top \right) = 90$$



Large attention!

Remember, there are multiple inputs to pay attention to

Keys and Queries

We compute attention for each input

Keys and Queries

We compute attention for each input

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

Keys and Queries

We compute attention for each input

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

Question: Anything missing from before?

Keys and Queries

We compute attention for each input

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

Question: Anything missing from before?

Answer: Normalize attention to sum to one!

Keys and Queries

Normalize, only pay attention to important matches

Keys and Queries

Normalize, only pay attention to important matches

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

Keys and Queries

Normalize, 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**

Keys and Queries

Query: Which person will help me on my exam?

Keys and Queries

Query: Which person will help me on my exam?



$$q^\top k_1$$

$$q^\top k_2$$

$$q^\top k_3$$

$$q^\top k_4$$

$$q^\top k_5$$

Keys and Queries

Query: Which person will help me on my exam?



$$q^\top k_1$$

$$q^\top k_2$$

$$q^\top k_3$$

$$q^\top k_4$$

$$q^\top k_5$$

−1.71

0.60

−1.01

−0.61

2.73

softmax

Keys and Queries

Query: Which person will help me on my exam?



$$q^\top k_1$$

$$q^\top k_2$$

$$q^\top k_3$$

$$q^\top k_4$$

$$q^\top k_5$$

$$-1.71$$

$$0.60$$

$$-1.01$$

$$-0.61$$

$$2.73$$

softmax

$$0.01$$

$$0.10$$

$$0.02$$

$$0.03$$

$$0.84$$

Keys and Queries

Query: Which person will help me on my exam?



$$q^\top k_1$$

$$q^\top k_2$$

$$q^\top k_3$$

$$q^\top k_4$$

$$q^\top k_5$$

$$-1.71$$

$$0.60$$

$$-1.01$$

$$-0.61$$

$$2.73$$

softmax

$$0.01$$

$$0.10$$

$$0.02$$

$$0.03$$

$$0.84$$

Keys and Queries

Query: Which person will help me on my exam?



$$q^\top k_1$$

$$q^\top k_2$$

$$q^\top k_3$$

$$q^\top k_4$$

$$q^\top k_5$$

$$-1.71$$

$$0.60$$

$$-1.01$$

$$-0.61$$

$$2.73$$

softmax

$$0.01$$

$$0.10$$

$$0.02$$

$$0.03$$

$$0.84$$

Keys and Queries

Put dot product attention into our composite model

Keys and Queries

Put dot product attention into our composite model

$$\lambda \left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \boldsymbol{\theta}_\lambda \right)_i = \text{softmax} \left(\mathbf{q}^\top \begin{bmatrix} \mathbf{k}_1 \\ \mathbf{k}_2 \\ \vdots \\ \mathbf{k}_T \end{bmatrix} \right)_i$$

Keys and Queries

Put dot product attention into our composite model

$$\lambda \left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \boldsymbol{\theta}_\lambda \right)_i = \text{softmax} \left(\mathbf{q}^\top \begin{bmatrix} \mathbf{k}_1 \\ \mathbf{k}_2 \\ \vdots \\ \mathbf{k}_T \end{bmatrix} \right)_i$$

Then, write our composite memory model with forgetting

Keys and Queries

Put dot product attention into our composite model

$$\lambda \left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \boldsymbol{\theta}_{\lambda} \right)_i = \text{softmax} \left(\mathbf{q}^{\top} \begin{bmatrix} \mathbf{k}_1 \\ \mathbf{k}_2 \\ \vdots \\ \mathbf{k}_T \end{bmatrix} \right)_i$$

Then, write our composite memory model with forgetting

$$f \left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \boldsymbol{\theta} \right) = \sum_{i=1}^T \boldsymbol{\theta}^{\top} \mathbf{x}_i \cdot \lambda \left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \boldsymbol{\theta}_{\lambda} \right)_i$$

Keys and Queries

Put dot product attention into our composite model

$$\lambda \left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \boldsymbol{\theta}_{\lambda} \right)_i = \text{softmax} \left(\mathbf{q}^{\top} \begin{bmatrix} \mathbf{k}_1 \\ \mathbf{k}_2 \\ \vdots \\ \mathbf{k}_T \end{bmatrix} \right)_i$$

Then, write our composite memory model with forgetting

$$f \left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \boldsymbol{\theta} \right) = \sum_{i=1}^T \boldsymbol{\theta}^{\top} \mathbf{x}_i \cdot \lambda \left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \boldsymbol{\theta}_{\lambda} \right)_i$$

Keys and Queries

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

Keys and Queries

$$f\left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \theta\right) = \sum_{i=1}^T \theta^\top \mathbf{x}_i \cdot \lambda\left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \theta_\lambda\right)_i$$

$$f\left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \theta\right) = \sum_{i=1}^T \theta_V^\top \mathbf{x}_i \cdot \lambda\left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \theta_\lambda\right)_i$$

Keys and Queries

$$f\left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \theta\right) = \sum_{i=1}^T \theta^\top \mathbf{x}_i \cdot \lambda\left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \theta_\lambda\right)_i$$

$$f\left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \theta\right) = \sum_{i=1}^T \theta_V^\top \mathbf{x}_i \cdot \lambda\left(\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}, \theta_\lambda\right)_i$$

In dot-product attention, we call $\theta_V^\top \mathbf{x}_i$ the **value**

Keys and Queries



$$q^\top k_1 \quad q^\top k_2 \quad q^\top k_3 \quad q^\top k_4 \quad q^\top k_5$$

$$\cdot \theta_V^\top x_1 \quad \cdot \theta_V^\top x_2 \quad \cdot \theta_V^\top x_3 \quad \cdot \theta_V^\top x_4 \quad \cdot \theta_V^\top x_5$$

Keys and Queries



$$q^\top k_1 \quad q^\top k_2 \quad q^\top k_3 \quad q^\top k_4 \quad q^\top k_5$$

$$\cdot \theta_V^\top x_1 \quad \cdot \theta_V^\top x_2 \quad \cdot \theta_V^\top x_3 \quad \cdot \theta_V^\top x_4 \quad \cdot \theta_V^\top x_5$$

Agenda

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

Agenda

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

Transformer

Previously, we chose our own $x_q = \text{Musician}$

Transformer

Previously, we chose our own $x_q = \text{Musician}$

We can also create queries from all the inputs

Transformer

Previously, we chose our own $x_q = \text{Musician}$

We can also create queries from all the inputs

$$Q = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_T \end{bmatrix} = \begin{bmatrix} \theta_Q^\top x_1 \\ \theta_Q^\top x_2 \\ \vdots \\ \theta_Q^\top x_T \end{bmatrix}, \quad Q \in \mathbb{R}^{T \times d_h}$$

Transformer

Previously, we chose our own $x_q = \text{Musician}$

We can also create queries from all the inputs

$$Q = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_T \end{bmatrix} = \begin{bmatrix} \theta_Q^\top x_1 \\ \theta_Q^\top x_2 \\ \vdots \\ \theta_Q^\top x_T \end{bmatrix}, \quad Q \in \mathbb{R}^{T \times d_h}$$

We call this dot-product **self** attention

Transformer

Writing dot-product self attention in matrix form is easier

Transformer

Writing dot-product self attention in matrix form is easier

$$Q = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_T \end{bmatrix} = \begin{bmatrix} \theta_Q^\top x_1 \\ \theta_Q^\top x_2 \\ \vdots \\ \theta_Q^\top x_T \end{bmatrix} \quad K = \begin{bmatrix} k_1 \\ k_2 \\ \vdots \\ k_T \end{bmatrix} = \begin{bmatrix} \theta_K^\top x_1 \\ \theta_K^\top x_2 \\ \vdots \\ \theta_K^\top x_T \end{bmatrix} \quad V = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_T \end{bmatrix} = \begin{bmatrix} \theta_V^\top x_1 \\ \theta_V^\top x_2 \\ \vdots \\ \theta_V^\top x_T \end{bmatrix}$$

Transformer

Writing dot-product self attention in matrix form is easier

$$Q = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_T \end{bmatrix} = \begin{bmatrix} \theta_Q^\top x_1 \\ \theta_Q^\top x_2 \\ \vdots \\ \theta_Q^\top x_T \end{bmatrix} \quad K = \begin{bmatrix} k_1 \\ k_2 \\ \vdots \\ k_T \end{bmatrix} = \begin{bmatrix} \theta_K^\top x_1 \\ \theta_K^\top x_2 \\ \vdots \\ \theta_K^\top x_T \end{bmatrix} \quad V = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_T \end{bmatrix} = \begin{bmatrix} \theta_V^\top x_1 \\ \theta_V^\top x_2 \\ \vdots \\ \theta_V^\top x_T \end{bmatrix}$$

$$\text{attn}\left(\begin{bmatrix} x_1 \\ \vdots \\ x_T \end{bmatrix}, \theta\right) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_h}}\right)V$$

Transformer

Writing dot-product self attention in matrix form is easier

$$Q = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_T \end{bmatrix} = \begin{bmatrix} \theta_Q^\top x_1 \\ \theta_Q^\top x_2 \\ \vdots \\ \theta_Q^\top x_T \end{bmatrix} \quad K = \begin{bmatrix} k_1 \\ k_2 \\ \vdots \\ k_T \end{bmatrix} = \begin{bmatrix} \theta_K^\top x_1 \\ \theta_K^\top x_2 \\ \vdots \\ \theta_K^\top x_T \end{bmatrix} \quad V = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_T \end{bmatrix} = \begin{bmatrix} \theta_V^\top x_1 \\ \theta_V^\top x_2 \\ \vdots \\ \theta_V^\top x_T \end{bmatrix}$$

$$\text{attn}\left(\begin{bmatrix} x_1 \\ \vdots \\ x_T \end{bmatrix}, \theta\right) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_h}}\right)V$$

This operation powers today's biggest models

Transformer

$$\mathbf{Q} \in \mathbb{R}^{T \times d_h} \quad \mathbf{K} \in \mathbb{R}^{T \times d_h} \quad \mathbf{V} \in \mathbb{R}^{T \times d_h}$$

$$\underbrace{\text{attn}\left(\underbrace{\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}}_{\mathbb{R}^{T \times d_h}}, \boldsymbol{\theta}\right)}_{\mathbb{R}^{T \times d_h}} = \text{softmax}\left(\overbrace{\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_h}}}^{\mathbb{R}^{T \times T}}, \underbrace{\mathbf{V}}_{\mathbb{R}^{T \times d_h}}\right)$$

Transformer

$$\mathbf{Q} \in \mathbb{R}^{T \times d_h} \quad \mathbf{K} \in \mathbb{R}^{T \times d_h} \quad \mathbf{V} \in \mathbb{R}^{T \times d_h}$$

$$\underbrace{\text{attn}\left(\underbrace{\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix}}_{\mathbb{R}^{T \times d_h}}, \boldsymbol{\theta}\right)}_{\mathbb{R}^{T \times d_h}} = \text{softmax}\left(\overbrace{\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_h}}}^{\mathbb{R}^{T \times T}}, \underbrace{\mathbf{V}}_{\mathbb{R}^{T \times d_h}}\right)$$

Transformer

```
class Attention(nn.Module):
    def __init__(self):
        self.theta_K = nn.Linear(d_x, d_h, bias=False)
        self.theta_Q = nn.Linear(d_x, d_h, bias=False)
        self.theta_V = nn.Linear(d_x, d_h, bias=False)

    def forward(self, x):
        A = softmax(self.theta_Q(x) @ self.theta_K(x) / d_h)
        return A @ self.theta_V(x)
```

Transformers

Transformer

If you understand attention, transformers are very simple

Transformer

If you understand attention, transformers are very simple

Each transformer consists of many “transformer blocks”

Transformer

If you understand attention, transformers are very simple

Each transformer consists of many “transformer blocks”

A transformer block is attention and an MLP

Transformer

```
class TransformerBlock(nn.Module):
    def __init__(self):
        self.attn = Attention()
        self.mlp = Sequential(
            Linear(d_h, d_h), LeakyReLU(), Linear(d_h, d_h))
        self.norm1 = nn.LayerNorm(d_h)
        self.norm2 = nn.LayerNorm(d_h)

    def forward(self, x):
        # Residual connection
        x = self.norm1(self.attn(x) + x)
        x = self.norm2(self.mlp(x) + x)
        return x
```

Transformer

```
class Transformer(nn.Module):
    def __init__(self):
        self.block1 = TransformerBlock()
        self.block2 = TransformerBlock()

        ...

    def forward(self, x):
        x = self.block1(x)
        x = self.block2(x)

        ...
        return x
```

Transformer

Agenda

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

Agenda

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

Coding

Question: Do we care about the order of inputs x_1, x_2, \dots in attention?

Coding

Question: Do we care about the order of inputs x_1, x_2, \dots in attention?

Let us see!

Coding

Question: Do we care about the order of inputs x_1, x_2, \dots in attention?

Let us see!

Define a permutation matrix $P \in \{0, 1\}^{T \times T}$

Coding

Question: Do we care about the order of inputs x_1, x_2, \dots in attention?

Let us see!

Define a permutation matrix $P \in \{0, 1\}^{T \times T}$

Example:

$$P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}; \quad a = \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix}$$

Coding

Coding

Question: Do we care about the order of inputs x_1, x_2, \dots in attention?

Let us see!

Define a permutation matrix $P \in \{0, 1\}^{T \times T}$

Example:

$$P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}; \quad a = \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix}$$

Coding

$$Pa = \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix}$$

Coding

Example:

$$P = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}; \quad a = \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix}$$

$$Pa = \begin{bmatrix} 4 \\ 3 \\ 5 \end{bmatrix}$$

Coding

Recall attention

$$Q = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_T \end{bmatrix} = \begin{bmatrix} \theta_Q^\top x_1 \\ \theta_Q^\top x_2 \\ \vdots \\ \theta_Q^\top x_T \end{bmatrix} \quad K = \begin{bmatrix} k_1 \\ k_2 \\ \vdots \\ k_T \end{bmatrix} = \begin{bmatrix} \theta_K^\top x_1 \\ \theta_K^\top x_2 \\ \vdots \\ \theta_K^\top x_T \end{bmatrix} \quad V = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_T \end{bmatrix} = \begin{bmatrix} \theta_V^\top x_1 \\ \theta_V^\top x_2 \\ \vdots \\ \theta_V^\top x_T \end{bmatrix}$$

Coding

Recall attention

$$Q = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_T \end{bmatrix} = \begin{bmatrix} \theta_Q^\top x_1 \\ \theta_Q^\top x_2 \\ \vdots \\ \theta_Q^\top x_T \end{bmatrix} \quad K = \begin{bmatrix} k_1 \\ k_2 \\ \vdots \\ k_T \end{bmatrix} = \begin{bmatrix} \theta_K^\top x_1 \\ \theta_K^\top x_2 \\ \vdots \\ \theta_K^\top x_T \end{bmatrix} \quad V = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_T \end{bmatrix} = \begin{bmatrix} \theta_V^\top x_1 \\ \theta_V^\top x_2 \\ \vdots \\ \theta_V^\top x_T \end{bmatrix}$$

$$\text{attn}\left(\begin{bmatrix} x_1 \\ \vdots \\ x_T \end{bmatrix}, \theta\right) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_h}}\right)V$$

Coding

What if we permute $[x_1 \dots x_T]^\top$ by P ?

Coding

What if we permute $[x_1 \dots x_T]^\top$ by P ?

$$\text{attn}\left(P \begin{bmatrix} x_1 \\ \vdots \\ x_T \end{bmatrix}, \theta\right) = \text{softmax}\left(\frac{PQPK^\top}{\sqrt{d_h}}\right)PV$$

$$PQ = P \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_T \end{bmatrix} \quad PK = P \begin{bmatrix} k_1 \\ k_2 \\ \vdots \\ k_T \end{bmatrix} \quad PV = P \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_T \end{bmatrix}$$

Coding

What if we permute $[x_1 \dots x_T]^\top$ by P ?

$$\text{attn}\left(P \begin{bmatrix} x_1 \\ \vdots \\ x_T \end{bmatrix}, \theta\right) = \text{softmax}\left(\frac{PQPK^\top}{\sqrt{d_h}}\right)PV$$

$$PQ = P \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_T \end{bmatrix} \quad PK = P \begin{bmatrix} k_1 \\ k_2 \\ \vdots \\ k_T \end{bmatrix} \quad PV = P \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_T \end{bmatrix}$$

Example:

Example:

$$\text{attn} \left(\begin{bmatrix} \mathbf{x}_T \\ \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_2 \end{bmatrix}, \theta \right) = \text{softmax} \left(\frac{\mathbf{Q}_P \mathbf{K}_P^\top}{\sqrt{d_h}} \right) \mathbf{V}_P$$

Coding

Example:

$$\text{attn} \left(\begin{bmatrix} \mathbf{x}_T \\ \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_2 \end{bmatrix}, \theta \right) = \text{softmax} \left(\frac{\mathbf{Q}_P \mathbf{K}_P^\top}{\sqrt{d_h}} \right) \mathbf{V}_P$$

$$\mathbf{Q}_P = \begin{bmatrix} \mathbf{q}_T \\ \mathbf{q}_1 \\ \vdots \\ \mathbf{q}_2 \end{bmatrix} \quad \mathbf{K}_P = \begin{bmatrix} \mathbf{k}_T \\ \mathbf{k}_1 \\ \vdots \\ \mathbf{k}_2 \end{bmatrix} \quad \mathbf{V}_P = \begin{bmatrix} \mathbf{v}_T \\ \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_2 \end{bmatrix}$$

Coding

$$P \text{ attn} \left(\begin{bmatrix} x_1 \\ \vdots \\ x_T \end{bmatrix}, \theta \right) = \text{attn} \left(P \begin{bmatrix} x_1 \\ \vdots \\ x_T \end{bmatrix}, \theta \right)$$

Attention is **permutation equivariant**

$$P \text{ attn} \left(\begin{bmatrix} x_1 \\ \vdots \\ x_T \end{bmatrix}, \theta \right) = \text{attn} \left(P \begin{bmatrix} x_1 \\ \vdots \\ x_T \end{bmatrix}, \theta \right)$$

Attention is **permutation equivariant**

Order of the inputs does not matter

Coding

This makes sense, in our party example with attention we never consider the order

Coding

This makes sense, in our party example with attention we never consider the order



Coding

This makes sense, in our party example with attention we never consider the order



Question: Any situations where input order matters?

What about language?

Coding

What about language?

$$\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \\ \mathbf{x}_5 \end{bmatrix} = \begin{bmatrix} \text{The} \\ \text{dog} \\ \text{licks} \\ \text{the} \\ \text{owner} \end{bmatrix}$$

Coding

What about language?

$$\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \\ \mathbf{x}_5 \end{bmatrix} = \begin{bmatrix} \text{The} \\ \text{dog} \\ \text{licks} \\ \text{the} \\ \text{owner} \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_5 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \text{The} \\ \text{owner} \\ \text{licks} \\ \text{the} \\ \text{dog} \end{bmatrix}$$

Coding

What about language?

$$\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \\ \mathbf{x}_5 \end{bmatrix} = \begin{bmatrix} \text{The} \\ \text{dog} \\ \text{licks} \\ \text{the} \\ \text{owner} \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_5 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \text{The} \\ \text{owner} \\ \text{licks} \\ \text{the} \\ \text{dog} \end{bmatrix}$$

Question: Do these have the same meaning?

Coding

What about language?

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} \text{The} \\ \text{dog} \\ \text{licks} \\ \text{the} \\ \text{owner} \end{bmatrix}$$

$$\begin{bmatrix} x_1 \\ \color{red}{x_5} \\ x_3 \\ x_4 \\ \color{red}{x_2} \end{bmatrix} = \begin{bmatrix} \text{The} \\ \text{owner} \\ \text{licks} \\ \text{the} \\ \text{dog} \end{bmatrix}$$

Question: Do these have the same meaning?

To attention, these have the same meaning!

Coding

What about language?

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} \text{The} \\ \text{dog} \\ \text{licks} \\ \text{the} \\ \text{owner} \end{bmatrix}$$

$$\begin{bmatrix} x_1 \\ \textcolor{red}{x_5} \\ x_3 \\ x_4 \\ \textcolor{red}{x_2} \end{bmatrix} = \begin{bmatrix} \text{The} \\ \text{owner} \\ \text{licks} \\ \text{the} \\ \text{dog} \end{bmatrix}$$

Question: Do these have the same meaning?

To attention, these have the same meaning!

We want to **break** the permutation equivariance for certain tasks

Question: What are some ways we can introduce ordering?

Coding

Question: What are some ways we can introduce ordering?

Answer 1: We can introduce forgetting

Coding

Question: What are some ways we can introduce ordering?

Answer 1: We can introduce forgetting

ALiBi: Press, Ofir, Noah Smith, and Mike Lewis. “*Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation.*” *International Conference on Learning Representations*.

Question: What are some ways we can introduce ordering?

Answer 1: We can introduce forgetting

ALiBi: Press, Ofir, Noah Smith, and Mike Lewis. “*Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation.*” *International Conference on Learning Representations*.

RoPE: Su, Jianlin, et al. “*Roformer: Enhanced transformer with rotary position embedding.*” *Neurocomputing*.

Coding

Question: What are some ways we can introduce ordering?

Answer 1: We can introduce forgetting

ALiBi: Press, Ofir, Noah Smith, and Mike Lewis. “*Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation.*” *International Conference on Learning Representations*.

RoPE: Su, Jianlin, et al. “*Roformer: Enhanced transformer with rotary position embedding.*” *Neurocomputing*.

Answer 2: We can modify the inputs based on their ordering

Coding

Question: What are some ways we can introduce ordering?

Answer 1: We can introduce forgetting

ALiBi: Press, Ofir, Noah Smith, and Mike Lewis. “*Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation.*” *International Conference on Learning Representations*.

RoPE: Su, Jianlin, et al. “*Roformer: Enhanced transformer with rotary position embedding.*” *Neurocomputing*.

Answer 2: We can modify the inputs based on their ordering

We will focus on answer 2 because it was used first

Approach 2: Modify inputs based on ordering

Approach 2: Modify inputs based on ordering

$$\text{attn} \left(\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_T \end{bmatrix}, \theta \right)$$

Approach 2: Modify inputs based on ordering

$$\text{attn} \left(\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_T \end{bmatrix}, \theta \right)$$

$$\text{attn} \left(\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_T \end{bmatrix} + \begin{bmatrix} f_{\text{pos}}(1) \\ f_{\text{pos}}(2) \\ \vdots \\ f_{\text{pos}}(3) \end{bmatrix}, \theta \right)$$

Coding

Approach 2: Modify inputs based on ordering

$$\text{attn} \left(\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_T \end{bmatrix}, \theta \right)$$

$$\text{attn} \left(\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_T \end{bmatrix} + \begin{bmatrix} f_{\text{pos}}(1) \\ f_{\text{pos}}(2) \\ \vdots \\ f_{\text{pos}}(3) \end{bmatrix}, \theta \right)$$

$$Q = \begin{bmatrix} \theta_Q^\top x_1 \\ \theta_Q^\top x_2 \\ \vdots \\ \theta_Q^\top x_T \end{bmatrix}$$

$$K = \begin{bmatrix} \theta_K^\top x_1 \\ \theta_K^\top x_2 \\ \vdots \\ \theta_K^\top x_T \end{bmatrix}$$

$$V = \begin{bmatrix} \theta_V^\top x_1 \\ \theta_V^\top x_2 \\ \vdots \\ \theta_V^\top x_T \end{bmatrix}$$

Coding

Approach 2: Modify inputs based on ordering

$$\text{attn} \left(\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_T \end{bmatrix}, \theta \right)$$

$$\text{attn} \left(\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_T \end{bmatrix} + \begin{bmatrix} f_{\text{pos}}(1) \\ f_{\text{pos}}(2) \\ \vdots \\ f_{\text{pos}}(3) \end{bmatrix}, \theta \right)$$

$$Q = \begin{bmatrix} \theta_Q^\top x_1 \\ \theta_Q^\top x_2 \\ \vdots \\ \theta_Q^\top x_T \end{bmatrix}$$

$$K = \begin{bmatrix} \theta_K^\top x_1 \\ \theta_K^\top x_2 \\ \vdots \\ \theta_K^\top x_T \end{bmatrix}$$

$$V = \begin{bmatrix} \theta_V^\top x_1 \\ \theta_V^\top x_2 \\ \vdots \\ \theta_V^\top x_T \end{bmatrix}$$

Coding

Now, keys and values

Now what happens if we permute the inputs?

Coding

Now, keys and values

Now what happens if we permute the inputs?

$$\text{attn} \left(\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_T \end{bmatrix} + \begin{bmatrix} f_{\text{pos}}(1) \\ f_{\text{pos}}(2) \\ \vdots \\ f_{\text{pos}}(3) \end{bmatrix}, \theta \right)$$

Coding

Now, keys and values

Now what happens if we permute the inputs?

$$\text{attn} \left(\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_T \end{bmatrix} + \begin{bmatrix} f_{\text{pos}}(1) \\ f_{\text{pos}}(2) \\ \vdots \\ f_{\text{pos}}(3) \end{bmatrix}, \theta \right)$$

Coding

```
class PositionalTransformer(nn.Module):
    def __init__(self):
        self.f_pos = nn.Embedding(1024, d_x)
        self.block1 = TransformerBlock()
        self.block2 = TransformerBlock()

    def forward(self, x):
        timesteps = torch.arange(x.shape[0])
        x = x + self.embedding(timesteps)
        x = self.block1(x)
        x = self.block2(x)
        return x
```

Coding

Homework

<https://colab.research.google.com/drive/18VBb7sz0u8ul5vsFEJnQaepn0pQy4cUa?usp=sharing>