# Lecture 13

# Performer, Variational Autoencoders

## Rethinking Attention with Performers

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, Adrian Weller

- "We introduce the first Transformer architectures, *Performers,* capable of **provably** accurate and practical estimations of regular (softmax) full rank attention, but of only linear space and time complexity and **not relying on any priors** such as sparsity or low-rankness. Performers use the *Fast Attention Via positive Orthogonal Random features* (FAVOR+) mechanism"

# Softmax attention with Performers

Each matrix has one column per token.

$$Q, K \in \mathbb{R}^{p \times n}, \qquad V \in \mathbb{R}^{d_v \times n}$$

For each query, softmax assigns weights across the keys.

$$Y = V \ \mathrm{softmax}_{\mathrm{col}}\left(\frac{K^\top Q}{\sqrt{p}}\right) \in \mathbb{R}^{d_v \times n}$$

# Column-wise softmax

Exponentiate the scaled scores entry by entry.

$$A = \exp\left(\frac{K^\top Q}{\sqrt{p}}\right)$$

Collect the column sums on the diagonal.

$$\mathbf{c} = A^\top \mathbf{1}_n, \qquad D = \mathrm{diag}(\mathbf{c})$$

$$\mathrm{softmax}_{\mathrm{col}}\left(\frac{K^\top Q}{\sqrt{p}}\right) = AD^{-1}, \qquad Y = VAD^{-1}$$

# One denominator per query

The first query column sums to two. The second sums to four.

$$A = \begin{bmatrix} 1 & 3 \\ 1 & 1 \end{bmatrix}, \qquad D = \begin{bmatrix} 2 & 0 \\ 0 & 4 \end{bmatrix}$$

Right multiplication divides each column by its own sum.

$$AD^{-1} = \begin{bmatrix} \frac{1}{2} & \frac{3}{4} \\ \frac{1}{2} & \frac{1}{4} \end{bmatrix}$$

# Random features for the softmax kernel

Apply the same feature map to each key and query.

$$K' = \Phi(p^{-1/4}K), \qquad Q' = \Phi(p^{-1/4}Q)$$

$$K', Q' \in \mathbb{R}^{r \times n}$$

Feature inner products approximate the exponentiated scores.

$$\widehat{A} = (K')^\top Q' \approx \exp\left(\frac{K^\top Q}{\sqrt{p}}\right)$$

# Random features for the Gaussian kernel

The Gaussian kernel measures similarity through squared distance.

$$k_{\mathrm{G}}(\mathbf{x}, \mathbf{y}) = \exp\left(-\tfrac{1}{2}\|\mathbf{x} - \mathbf{y}\|^2\right)$$

Draw independent Gaussian directions. Each supplies two coordinates.

$$\boldsymbol{\omega}_j \sim \mathcal{N}(\mathbf{0}, I_p), \qquad j = 1, \ldots, s, \qquad r = 2s$$

$$\boldsymbol{\phi}_{\mathrm{G}}(\mathbf{x}) = \frac{1}{\sqrt{s}} \left[\cos(\boldsymbol{\omega}_1^\top \mathbf{x}),\ \sin(\boldsymbol{\omega}_1^\top \mathbf{x}),\ \ldots,\ \cos(\boldsymbol{\omega}_s^\top \mathbf{x}),\ \sin(\boldsymbol{\omega}_s^\top \mathbf{x})\right]^\top$$

# One direction gives one cosine

Project both inputs onto the same direction.

$$a_j = \boldsymbol{\omega}_j^\top \mathbf{x}, \qquad b_j = \boldsymbol{\omega}_j^\top \mathbf{y}$$

The two matching coordinates contribute this dot product before scaling.

$$\begin{bmatrix}\cos a_j & \sin a_j\end{bmatrix} \begin{bmatrix}\cos b_j \\ \sin b_j\end{bmatrix} = \cos a_j \cos b_j + \sin a_j \sin b_j$$

$$\cos a_j \cos b_j + \sin a_j \sin b_j = \cos(a_j - b_j) = \cos\left(\boldsymbol{\omega}_j^\top (\mathbf{x} - \mathbf{y})\right)$$

# The feature dot product is an average

Add the contributions from all directions. Each map supplies a scale factor.

$$(1/\sqrt{s})(1/\sqrt{s}) = 1/s$$

$$\boldsymbol{\phi}_{\mathrm{G}}(\mathbf{x})^{\top}\boldsymbol{\phi}_{\mathrm{G}}(\mathbf{y}) = \frac{1}{s}\sum_{j=1}^{s}\left(\cos a_j \cos b_j + \sin a_j \sin b_j\right)$$

Replace each pair of terms by its cosine difference.

$$\boldsymbol{\phi}_{\mathrm{G}}(\mathbf{x})^{\top}\boldsymbol{\phi}_{\mathrm{G}}(\mathbf{y}) = \frac{1}{s}\sum_{j=1}^{s}\cos\left(\boldsymbol{\omega}_j^{\top}(\mathbf{x}-\mathbf{y})\right)$$

# The distribution of one projection

Fix the two inputs and draw one Gaussian direction.

$$\boldsymbol{\delta} = \mathbf{x} - \mathbf{y}, \qquad \boldsymbol{\omega} \sim \mathcal{N}(\mathbf{0}, I_p)$$

The projection is Gaussian. Its variance is the squared distance.

$$u = \boldsymbol{\omega}^\top \boldsymbol{\delta}, \qquad \mathrm{Var}(u) = \boldsymbol{\delta}^\top I_p \boldsymbol{\delta} = \|\boldsymbol{\delta}\|^2$$

$$u \overset{\mathrm{d}}{=} \|\boldsymbol{\delta}\| \, z, \qquad z \sim \mathcal{N}(0, 1)$$

# The mean cosine of a Gaussian

Let the frequency vary and differentiate the mean cosine.

$$z \sim \mathcal{N}(0,1), \qquad h(t) = \mathbb{E}[\cos(tz)], \qquad h(0) = 1$$

$$h^{'}(t) = -\mathbb{E}[z \sin(tz)]$$

Gaussian integration by parts gives the differential equation.

$$\mathbb{E}[z \sin(tz)] = t\, \mathbb{E}[\cos(tz)] = t\, h(t)$$

$$h^{'}(t) = -t\, h(t) \qquad \Longrightarrow \qquad h(t) = e^{-t^2/2}$$

# Why the average gives a Gaussian kernel

Set the frequency equal to the distance between the inputs.

$$\mathbb{E}_{\boldsymbol{\omega}}[\cos(\boldsymbol{\omega}^\top \boldsymbol{\delta})] = h(\|\boldsymbol{\delta}\|) = e^{-\|\boldsymbol{\delta}\|^2/2} = k_{\mathrm{G}}(\mathbf{x}, \mathbf{y})$$

The law of large numbers takes the sample average to this expectation.

$$\boldsymbol{\phi}_{\mathrm{G}}(\mathbf{x})^\top \boldsymbol{\phi}_{\mathrm{G}}(\mathbf{y}) = \frac{1}{s}\sum_{j=1}^{s} \cos(\boldsymbol{\omega}_j^\top \boldsymbol{\delta}) \;\longrightarrow\; k_{\mathrm{G}}(\mathbf{x}, \mathbf{y}) \quad (s \to \infty)$$

With finitely many directions, the dot product estimates the kernel.

# Expanding the Gaussian exponent

Expand the squared distance before applying the negative one-half.

$$\|\mathbf{x}-\mathbf{y}\|^2 = \|\mathbf{x}\|^2 + \|\mathbf{y}\|^2 - 2\mathbf{x}^\top\mathbf{y}$$

$$-\tfrac{1}{2}\|\mathbf{x}-\mathbf{y}\|^2 = -\tfrac{1}{2}\|\mathbf{x}\|^2 - \tfrac{1}{2}\|\mathbf{y}\|^2 + \mathbf{x}^\top\mathbf{y}$$

Exponentiating separates the three terms into three factors.

$$k_{\mathrm{G}}(\mathbf{x},\mathbf{y}) = e^{-\|\mathbf{x}\|^2/2}\, e^{\mathbf{x}^\top\mathbf{y}}\, e^{-\|\mathbf{y}\|^2/2}$$

# Isolating the softmax exponential

Undo the two negative norm factors by multiplying by their reciprocals.

$$e^{\mathbf{x}^\top \mathbf{y}} = e^{\|\mathbf{x}\|^2/2}\, k_{\mathrm{G}}(\mathbf{x}, \mathbf{y})\, e^{\|\mathbf{y}\|^2/2}$$

Replace the Gaussian kernel by its feature dot product.

$$e^{\mathbf{x}^\top \mathbf{y}} \approx \left(e^{\|\mathbf{x}\|^2/2} \boldsymbol{\phi}_{\mathrm{G}}(\mathbf{x})\right)^\top \left(e^{\|\mathbf{y}\|^2/2} \boldsymbol{\phi}_{\mathrm{G}}(\mathbf{y})\right)$$

$$\boldsymbol{\phi}_{\exp}(\mathbf{x}) = e^{\|\mathbf{x}\|^2/2} \boldsymbol{\phi}_{\mathrm{G}}(\mathbf{x})$$

# A positive estimate of the exponential

A finite cosine average can be negative. Performer uses positive features.

For a Gaussian direction, the exponential moment has a closed form.

$$\mathbb{E}_{\boldsymbol{\omega}}[e^{\boldsymbol{\omega}^\top \mathbf{t}}] = e^{\|\mathbf{t}\|^2/2}, \qquad \boldsymbol{\omega} \sim \mathcal{N}(\mathbf{0}, I_p)$$

Choose the sum of the inputs and expand its squared norm.

$$\mathbb{E}_{\boldsymbol{\omega}}[e^{\boldsymbol{\omega}^\top (\mathbf{x}+\mathbf{y})}] = e^{\|\mathbf{x}+\mathbf{y}\|^2/2} = e^{\|\mathbf{x}\|^2/2}\, e^{\mathbf{x}^\top \mathbf{y}}\, e^{\|\mathbf{y}\|^2/2}$$

# One positive feature per direction

Cancel the norm factors in the Gaussian exponential moment.

$$e^{\mathbf{x}^\top \mathbf{y}} = e^{-\|\mathbf{x}\|^2/2}\, \mathbb{E}_{\boldsymbol{\omega}}\left[e^{\boldsymbol{\omega}^\top \mathbf{x}} e^{\boldsymbol{\omega}^\top \mathbf{y}}\right] e^{-\|\mathbf{y}\|^2/2}$$

Group each input with its own norm correction inside the expectation.

$$g_{\boldsymbol{\omega}}(\mathbf{x}) = \exp\left(\boldsymbol{\omega}^\top \mathbf{x} - \tfrac{1}{2}\|\mathbf{x}\|^2\right) > 0$$

$$e^{\mathbf{x}^\top \mathbf{y}} = \mathbb{E}_{\boldsymbol{\omega}}\left[g_{\boldsymbol{\omega}}(\mathbf{x})\, g_{\boldsymbol{\omega}}(\mathbf{y})\right]$$

# FAVOR+ in Performer

FAVOR+ stands for Fast Attention Via positive Orthogonal Random features.

Stack the positive features and scale the vector.

$$\boldsymbol{\phi}_+(\mathbf{x}) = \frac{1}{\sqrt{r}}\left[g_{\boldsymbol{\omega}_1}(\mathbf{x}), \ldots, g_{\boldsymbol{\omega}_r}(\mathbf{x})\right]^\top \in \mathbb{R}^r$$

$$\boldsymbol{\phi}_+(\mathbf{x})^\top \boldsymbol{\phi}_+(\mathbf{y}) = \frac{1}{r}\sum_{j=1}^{r} g_{\boldsymbol{\omega}_j}(\mathbf{x}) g_{\boldsymbol{\omega}_j}(\mathbf{y}) \approx e^{\mathbf{x}^\top \mathbf{y}}$$

Performer chooses directions that are orthogonal within each block.

# Applying the map to keys and queries

Scale both inputs so their dot product has the attention scaling.

$$(p^{-1/4}K)^\top(p^{-1/4}Q) = \frac{K^\top Q}{\sqrt{p}}$$

Apply the positive feature map to each key and query column.

$$K' = \Phi(p^{-1/4}K), \qquad Q' = \Phi(p^{-1/4}Q), \qquad K', Q' \in \mathbb{R}^{r\times n}$$

$$\widehat{A} = (K')^\top Q' \approx \exp\left(\frac{K^\top Q}{\sqrt{p}}\right)$$

# Normalizing the approximate kernel

The approximate kernel has its own column sums.

$$\hat{\mathbf{c}} = \widehat{A}^\top \mathbf{1}_n, \qquad \widehat{D} = \mathrm{diag}(\hat{\mathbf{c}})$$

Use those sums to normalize the approximate output.

$$\widehat{Y} = V\widehat{A}\widehat{D}^{-1} = V\big((K')^\top Q'\big)\widehat{D}^{-1}$$

Forming the middle product would still compute every pairwise score.

# The normalization denominators

Transpose the feature product to expose a shared sum over keys.

$$\hat{\mathbf{c}} = \left((K')^\top Q'\right)^\top \mathbf{1}_n = (Q')^\top \left(K' \mathbf{1}_n\right)$$

Sum the key columns once, then evaluate all query denominators.

$$\mathbf{z} = K' \mathbf{1}_n \in \mathbb{R}^r, \qquad \hat{\mathbf{c}} = (Q')^\top \mathbf{z} \in \mathbb{R}^n$$

# The attention numerator

Group the value matrix and the key features first.

$$V((K')^\top Q') = (V(K')^\top)Q'$$

The shared summary has one column per random feature.

$$S = V(K')^\top \in \mathbb{R}^{d_v \times r}$$

# Normalized Performer attention

For bidirectional attention, compute both summaries once.

$$S = V(K')^\top, \qquad \mathbf{z} = K'\mathbf{1}_n$$

Divide each output column by its own denominator.

$$Y \approx \widehat{Y} = (SQ') \operatorname{diag}\big((Q')^\top \mathbf{z}\big)^{-1}$$

Causal attention uses running summaries of the prefix through each query.

# Computational cost

Direct softmax attention computes a score for every key and query pair.

$$\mathcal{O}(n^2 p + n^2 d_v)$$

Performer computes the features and reuses them for all queries.

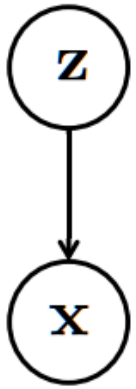
$$\underbrace{\mathcal{O}(npr)}_{\text{feature map}} + \underbrace{\mathcal{O}(nrd_v + nr)}_{\text{attention from features}}$$

With fixed dimensions, the total cost is linear in sequence length.

# Variational Auto encoder (VEA)

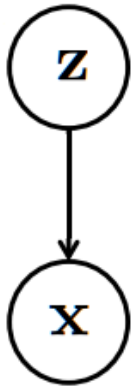
# Variational Inference

- Problem Definition
  - Observable Data: $x = \{x_1, x_2, ..., x_n\}$
  - Hidden Variable: $z = \{z_1, z_2, ..., z_n\}$



# Variational Inference

- Problem Definition
  - Observable Data: $x = \{x_1, x_2, ..., x_n\}$
  - Hidden Variable: $z = \{z_1, z_2, ..., z_n\}$



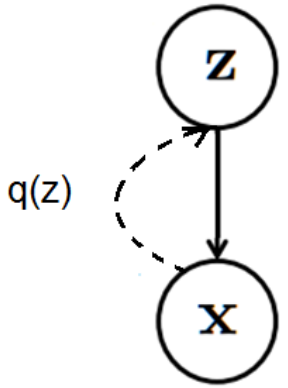

$$p(z|x) = \frac{p(z,x)}{p(x)} = \frac{p(x|z)p(z)}{\int p(x|z)p(z)dz}$$

# Variational Inference

- Solutions
  - Monte Carlo Sampling
    - Metropolis Hasting
    - Gibbs Sampling
  - Variational Inference

# Variational Inference

- Approximate $p(z|x)$ by $q(z)$




- Minimize the KL Divergence:

$$D_{KL}\Big[q(z)||p(z|x)\Big] = -\int q(z) log \frac{p(z|x)}{q(z)} dz$$

# Variational Lower Bound

$$D_{KL}\Big[q(z)||p(z|x)\Big] = -\int q(z) log \frac{p(z|x)}{q(z)} dz$$

# Variational Lower Bound

$$D_{KL}\Big[q(z)||p(z|x)\Big] = -\int q(z) log \frac{p(z|x)}{q(z)} dz$$

$$= -\int q(z) log \frac{p(z,x)}{q(z)p(x)} dz$$

# Variational Lower Bound

$$D_{KL}\Big[q(z)||p(z|x)\Big] = -\int q(z) log \frac{p(z|x)}{q(z)} dz$$

$$= -\int q(z) log \frac{p(z,x)}{q(z)p(x)} dz$$

$$= -\int q(z) log \frac{p(z,x)}{q(z)} dz + \int q(z) log\big(p(x)\big) dz$$

# Variational Lower Bound

$$D_{KL}\Big[q(z)||p(z|x)\Big] = -\int q(z) log \frac{p(z|x)}{q(z)} dz$$

$$= -\int q(z) log \frac{p(z,x)}{q(z)p(x)} dz$$

$$= -\int q(z) log \frac{p(z,x)}{q(z)} dz + \int q(z) log\big(p(x)\big) dz$$

$$= -\int q(z) \Big( log\big(p(z,x)\big) - log\big(q(z)\big) \Big) dz + log\big(p(x)\big)$$

# Variational Lower Bound

$$D_{KL}\Big[q(z)||p(z|x)\Big] = -\int q(z) log \frac{p(z|x)}{q(z)} dz$$

$$= -\int q(z) log \frac{p(z,x)}{q(z)p(x)} dz$$

$$= -\int q(z) log \frac{p(z,x)}{q(z)} dz + \int q(z) log\big(p(x)\big) dz$$

$$= -\int q(z) \Big( log\big(p(z,x)\big) - log\big(q(z)\big) \Big) dz + log\big(p(x)\big)$$

$$= -\underbrace{\Big(E_{q(z)}\Big[log\big(p(z,x)\big)\Big] - E_{q(z)}\Big[log\big(q(z)\big)\Big]\Big)}_{\text{Evidence Lower Bound (ELBO)}} + log\big(p(x)\big)$$

# Variational Lower Bound

$$D_{KL}\Big[q(z)||p(z|x)\Big] = -\underbrace{(E_{q(z)}\Big[log\big(p(z,x)\big)\Big] - E_{q(z)}\Big[log\big(q(z)\big)\Big])}_{\text{Evidence Lower Bound (ELBO)}} + log\big(p(x)\big)$$

# Variational Lower Bound

$$D_{KL}\Big[q(z)||p(z|x)\Big] = -\underbrace{(E_{q(z)}\Big[log\big(p(z,x)\big)\Big] - E_{q(z)}\Big[log\big(q(z)\big)\Big])}_{\text{Evidence Lower Bound (ELBO)}} + log\big(p(x)\big)$$

$$D_{KL}\Big[q(z)||p(z|x)\Big] = -L\Big[q(z)\Big] + log\Big(p(x)\Big)$$

# Variational Lower Bound

$$D_{KL}\Big[q(z)||p(z|x)\Big] = -\underbrace{(E_{q(z)}\Big[log\big(p(z,x)\big)\Big] - E_{q(z)}\Big[log\big(q(z)\big)\Big])}_{\text{Evidence Lower Bound (ELBO)}} + log\big(p(x)\big)$$

$$D_{KL}\Big[q(z)||p(z|x)\Big] = -L\Big[q(z)\Big] + log\Big(p(x)\Big)$$

$$log\Big(p(x)\Big) = D_{KL}\Big[q(z)||p(z|x)\Big] + L\Big[q(z)\Big]$$

# Variational Lower Bound

$$D_{KL}\Big[q(z)||p(z|x)\Big] = -\underbrace{(E_{q(z)}\Big[log\big(p(z,x)\big)\Big] - E_{q(z)}\Big[log\big(q(z)\big)\Big])}_{\text{Evidence Lower Bound (ELBO)}} + log\big(p(x)\big)$$

$$D_{KL}\Big[q(z)||p(z|x)\Big] = -L\Big[q(z)\Big] + log\Big(p(x)\Big)$$

$$log\Big(p(x)\Big) = D_{KL}\Big[q(z)||p(z|x)\Big] + L\Big[q(z)\Big]$$

Minimizing $D_{KL}\Big[q(z)||p(z|x)\Big]$

is equal to Maximizing $L\Big[q(z)\Big]$

VAE

