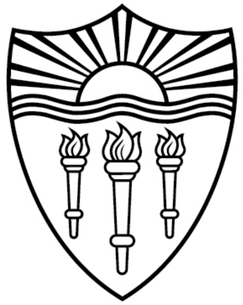
CSCI 544: Applied Natural Language Processing

Advances in Transformer II & Deep Generative Models

Xuezhe Ma (Max)



USC University of Southern California

Recap: Advances In NMT

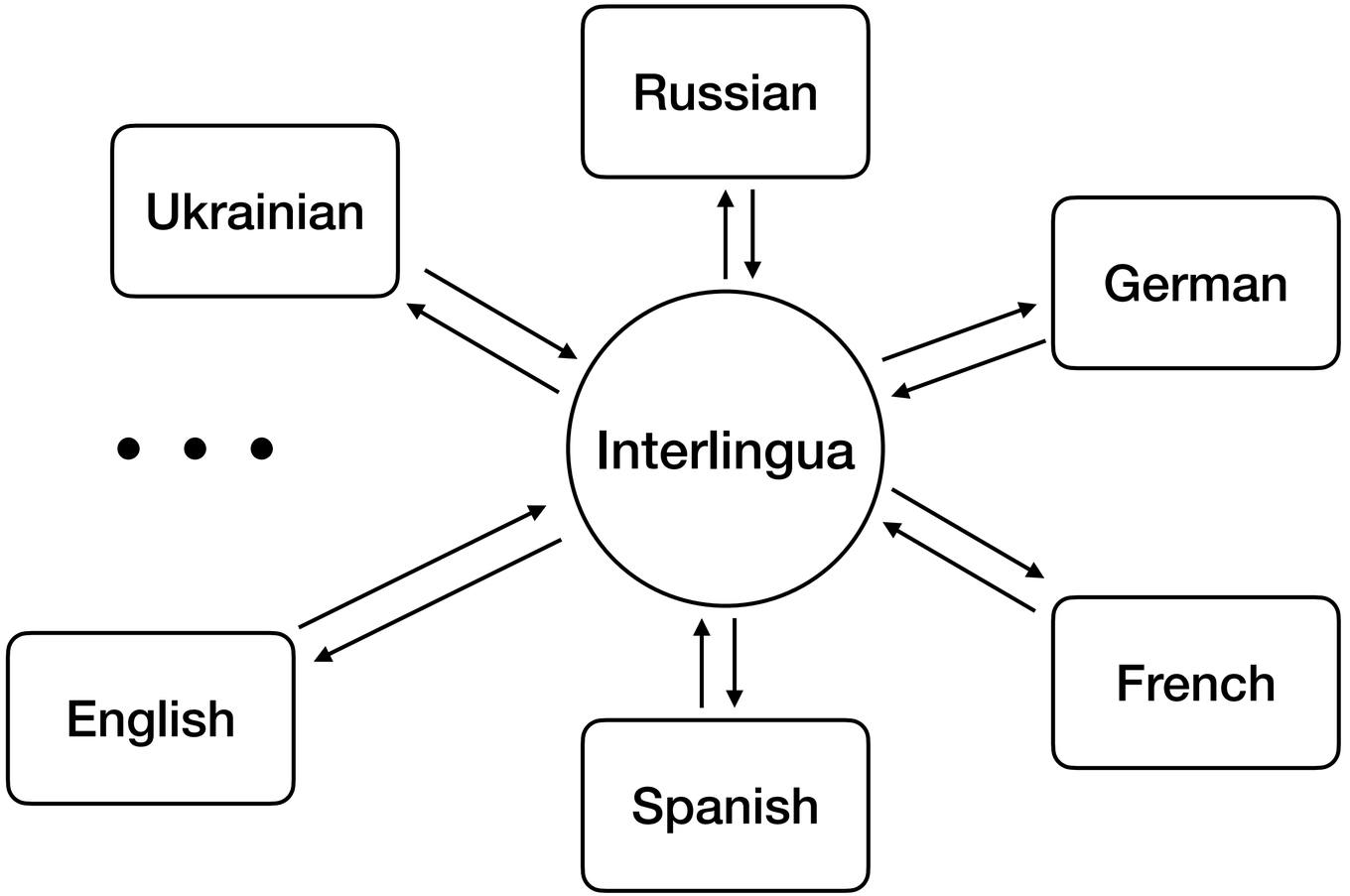
- Semi-Supervised NMT
- Multilingual NMT
- Evaluation beyond BLEU

Recap: Semi-Supervised NMT

- **Motivation**
  - Leveraging large-scale monolingual data to improve MT models
  - Monolingual target sentences: back-translation
  - Monolingual source sentences: self-learning
- **Empirical Evidences**
  - Genuine sentences helps decoder: better language model
  - Noisy sentences helps encoder: robust against noise

Multilingual NMT

- **Interlingual representation for NMT**

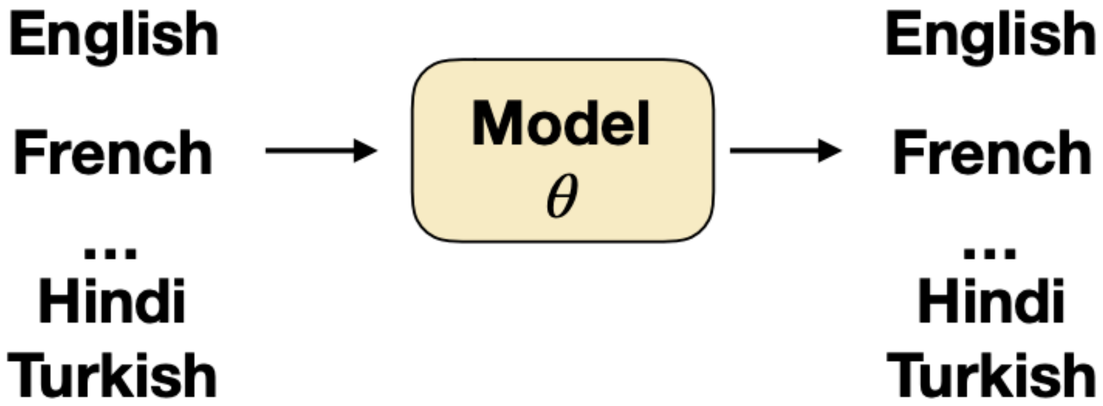


Small languages benefit from big ones that are in the same language family

Byte Pair Encoding: Pros and Cons

- **Pros**
  - Trade-off between char/byte -level tokens and original words
  - Capturing shared morphemes/sub-words across similar languages
  - Usually no *unknown* words, unless meeting special/uncommon characters
  - Not only for multilingual tasks, but also for monolingual ones
- **Cons**
  - Shallow similarity, working well only on similar languages
  - Over-segment low-resource or morphologically rich languages

Recap: Many-to-Many NMT



<2fr> How are you? Comment ça va? French

<2es> How are you? cómo estás? Spanish

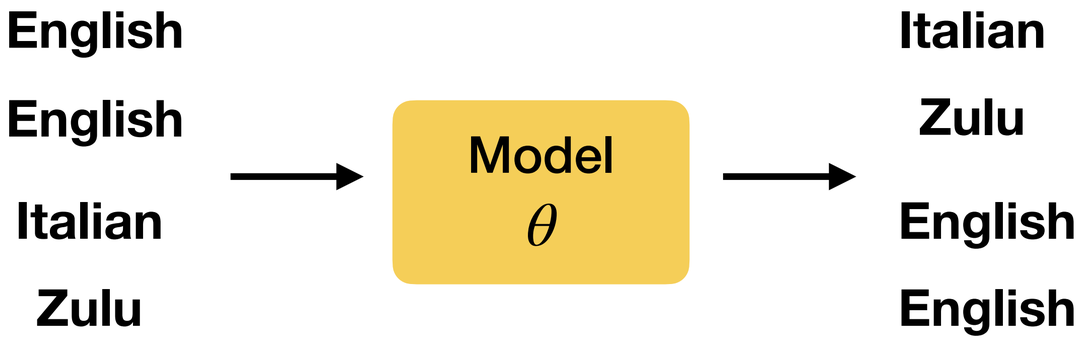
<2tr> How are you? nasılsın? Turkish

slides credit: Xinyi Wang

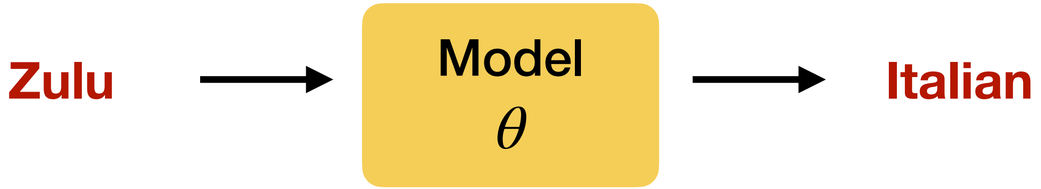
Recap: Zero-shot Transfer

- **Training on {English-Zulu, Zulu-English, English-Italian, Italian-English}**
- **Zero-shot transfer: the model can translate directly between Zulu and Italian**

Training:



Testing:



Google's multilingual neural machine translation system. (Johnson et al., 2016)

Recap: MT Evaluation

- **Drawbacks of n-gram based metrics:**
  - Penalize semantically-correct paraphases due to string matching (Banerjee & Lavie, 2005)
    - e.g. “No worries!” and “Don’t worry!”
  - Fail to capture distant dependencies and penalize semantically-critical ordering changes or word drops(Isozaki et al., 2010)
    - e.g. “A because B” and “B because A”
    - e.g. “A do not like B” and “A likes B”
- **Recent Proposed Metrics: contextualized embedding based metrics**
  - BERTScores (Zhang et al., 2020)
    - Computes token similarity using contextual embedding between candidate and reference
  - BLEURT (Stellam et al., 2020)
    - A learned evaluation metric based on BERT

Outline

- **More Advances in Transformers**
  - Efficient Attention Mechanisms
- **Deep Generative Models**
  - Neural Language Models
  - Variational Auto-Encoders (VAEs)
  - Generative (Normalizing) Flows

Efficient Attention Mechanisms

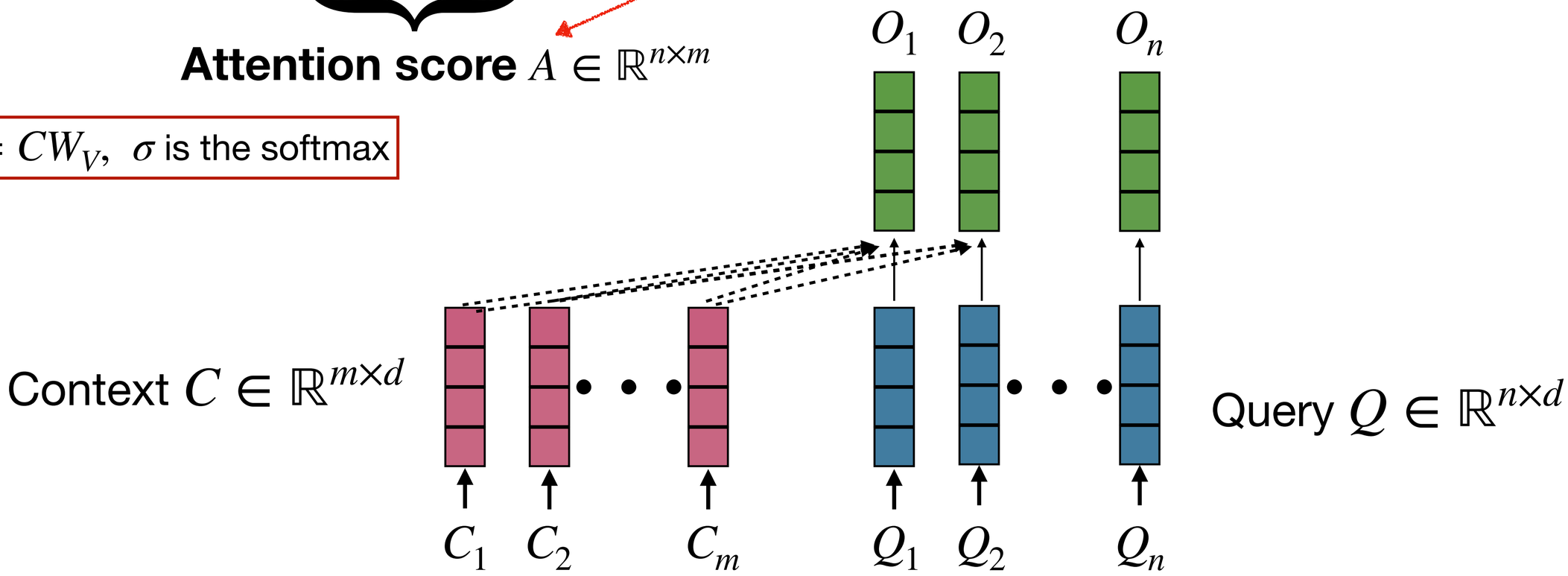
Attention Mechanism

Inputs: $Q \in \mathbb{R}^{n \times d},\ C \in \mathbb{R}^{m \times d}$

Outputs: $O = Attn(Q, C) = \underbrace{\sigma\left(\frac{QK^T}{\sqrt{d}}\right)}_{\text{Attention score } A \in \mathbb{R}^{n \times m}} V$

time and memory consuming

$K = CW_K,\ V = CW_V,\ \sigma$ is the softmax

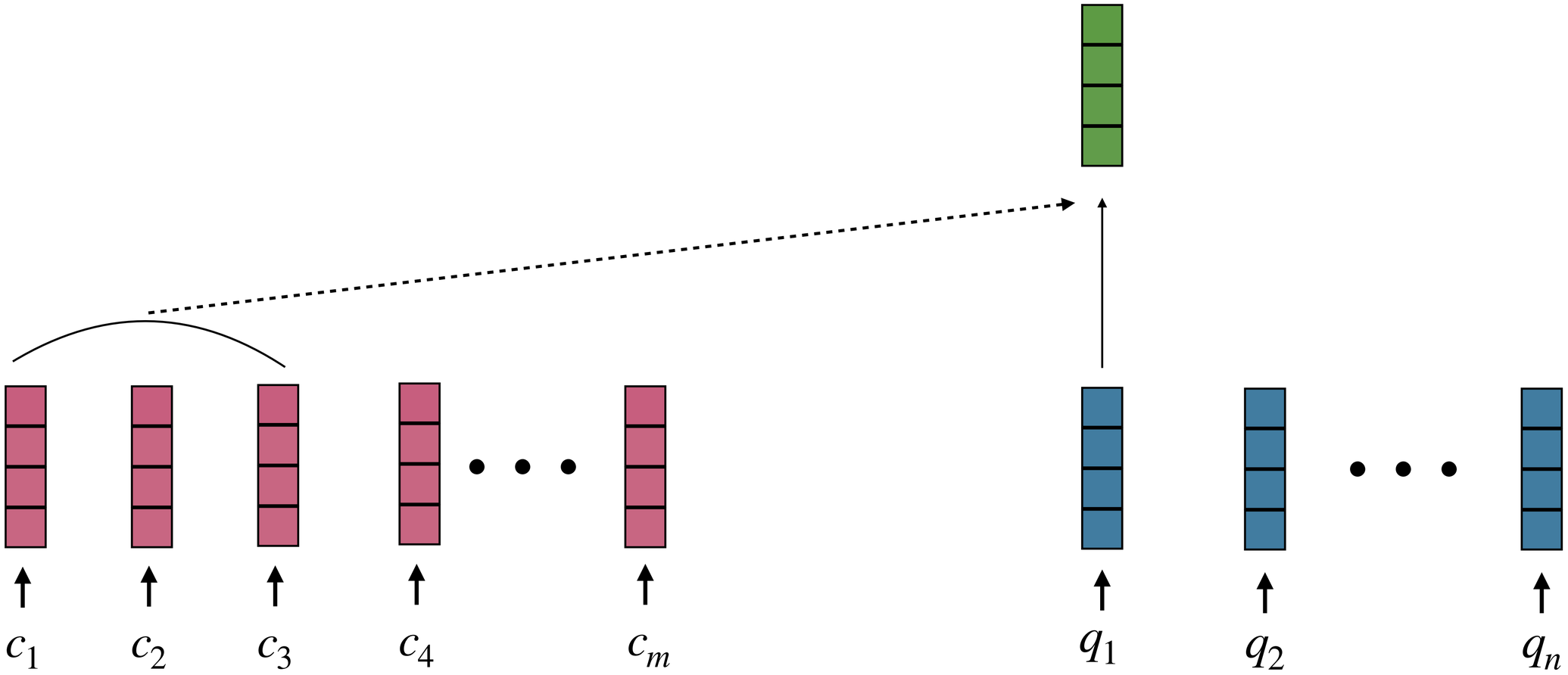


Previous Work: Efficient Attention

- **Sparse Attention**
  - Local attention
    - Image Transformer (Parmar et al., 2018)
  - Stride attention
    - Sparse Transformer (Child et al., 2019)
    - Longformer (Beltagy et al., 2020)
  - Attention with learnable patterns
    - Reformer (Kitaev et al., 2020)
    - Sinkhorn attention (Tay et al., 2020)
- **Decoupling Attention Matrix**
  - Lambda Network (Bello, et al., 2021)
  - Performer (Chor, et al., 2021)
  - Random Feature Attention (RFA) (Peng et al., 2021)
- **Low-rank Approximation**
  - Linformer (Wang et al., 2020)
  - Luna (Ma et al., 2021)

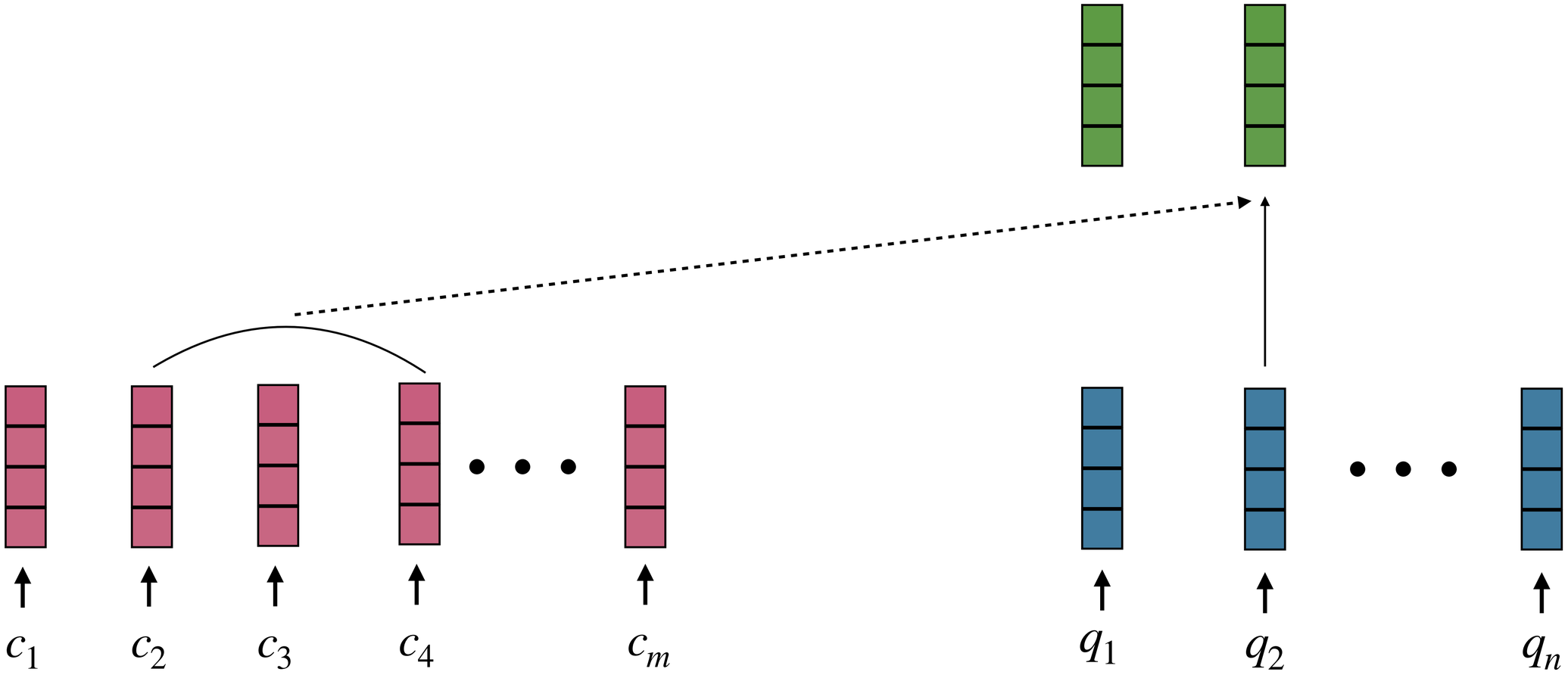
Sparse Attention

- **Sliding Window Attention**

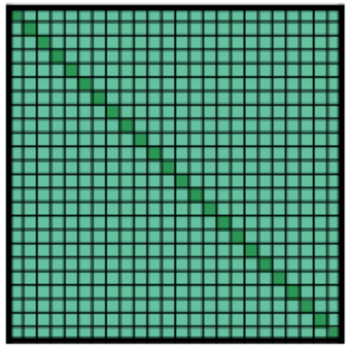


Sparse Attention

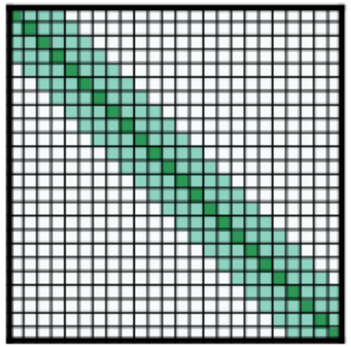
- **Sliding Window Attention**



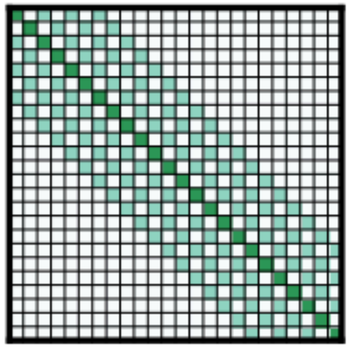
Sparse Attention



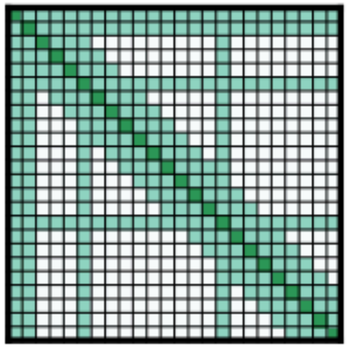
(a) Full n^2 attention



(b) Sliding window attention



(c) Dilated sliding window



(d) Global+sliding window

Longformer (Beltagy, et al., 2020)

Decoupling Attention Matrix

$$\sigma\left(\frac{QK^T}{\sqrt{d}}\right)V = \underbrace{A}_{n\times m} \times \underbrace{V}_{m\times d}$$

$$\approx \left(\underbrace{Q'}_{n\times k} \times \underbrace{(K')^T}_{k\times m}\right) \times \underbrace{V}_{m\times d}$$

$$\approx \underbrace{Q'}_{n\times k} \times \left(\underbrace{(K')^T}_{k\times m} \times \underbrace{V}_{m\times d}\right)$$

$$k \ll n, m$$

Decoupling Attention Matrix

- **Lambda Network** (Bello, 2021)

$$Q' = Q,\ (K')^T = \sigma\left(\frac{K^T}{\sqrt{d}}\right)$$

Decoupling Attention Matrix

- **Kernel Methods** (Chor, et al., 2021, Peng et al., 2021)

$$Q' = \phi(Q),\ (K')^T = \phi(K^T)$$

$$\phi(x) = \frac{1}{\sqrt{k}}\left[\sin(w_1^T x), \ldots, \sin(w_k^T x), \cos(w_1^T x), \ldots, \cos(w_k^T x)\right]$$

$$w_i \sim \mathcal{N}(0,\ \sigma^2 I)$$

$$\mathbb{E}_W\left[\phi(x)^T \phi(y)\right] = \exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right)$$

Low-Rank Approximation

- **Motivation: projecting context C into a shorter length**

Linear Attention (Linformer, Wang et al., 2020)

$$\sigma\left(\frac{Q(EC)^T}{\sqrt{d}}\right)(FC)$$

$n \times l$ $\quad$ $l \times d$

$$l \ll n, m$$

$$E,\ F \in \mathbb{R}^{l \times m}$$

learnable parameters

Problems:

- Cannot model sequences with various lengths
- E and F do NOT contain contextual information

Linear Nested Attention: Pack & Unpack

- **Motivation: first packing context C into a shorter length, then unpacking with query Q**

Linear Attention

$$\underbrace{\sigma\left(\frac{Q(EC)^T}{\sqrt{d}}\right)}_{n\times l}\underbrace{(FC)}_{l\times d}$$

A nested attention

Pack Attention

$$C' = \underbrace{\sigma\left(\frac{PC^T}{\sqrt{d}}\right)}_{l\times m} \underbrace{C}_{m\times d}$$

$P \in \mathbb{R}^{l\times d}$

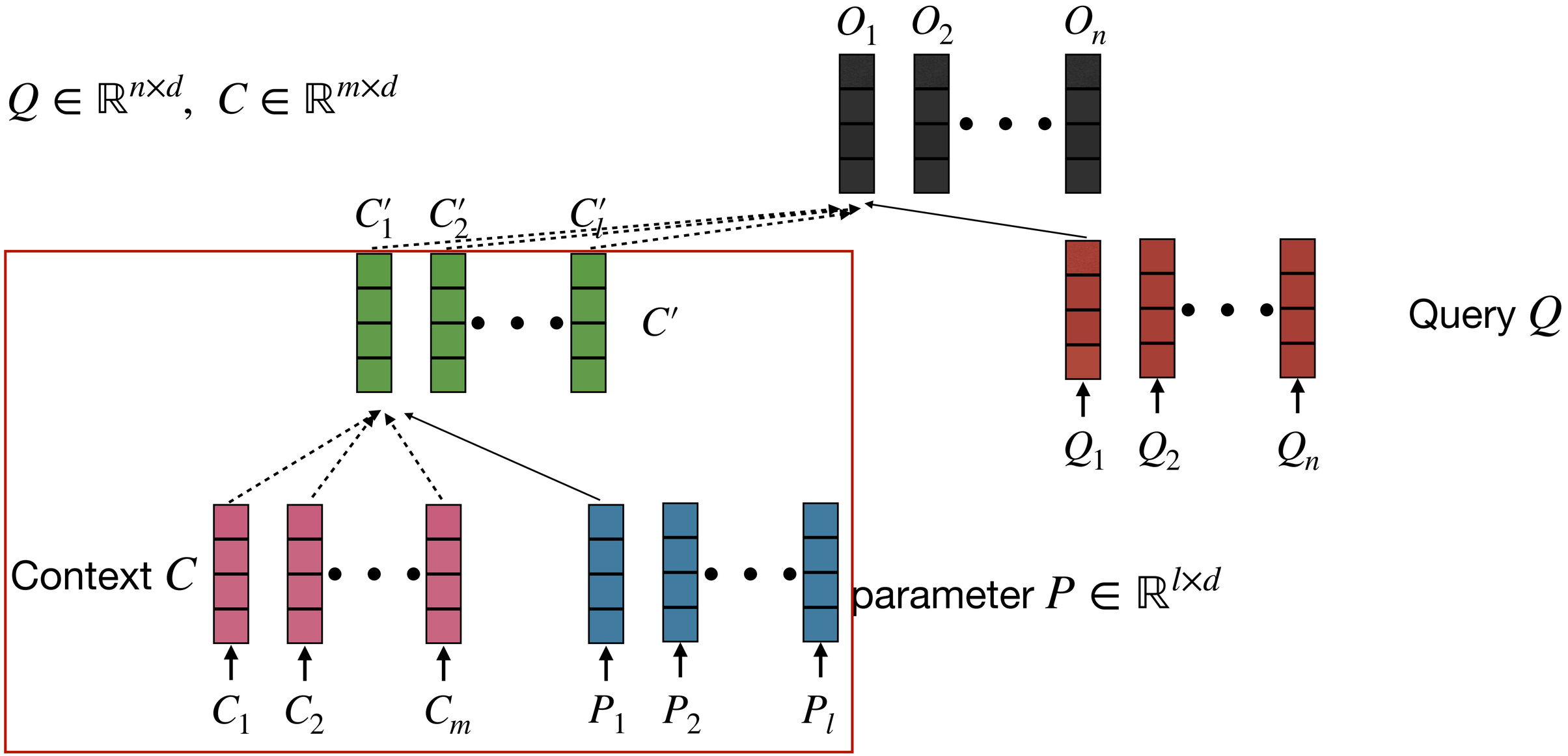
learnable parameters

Unpack Attention

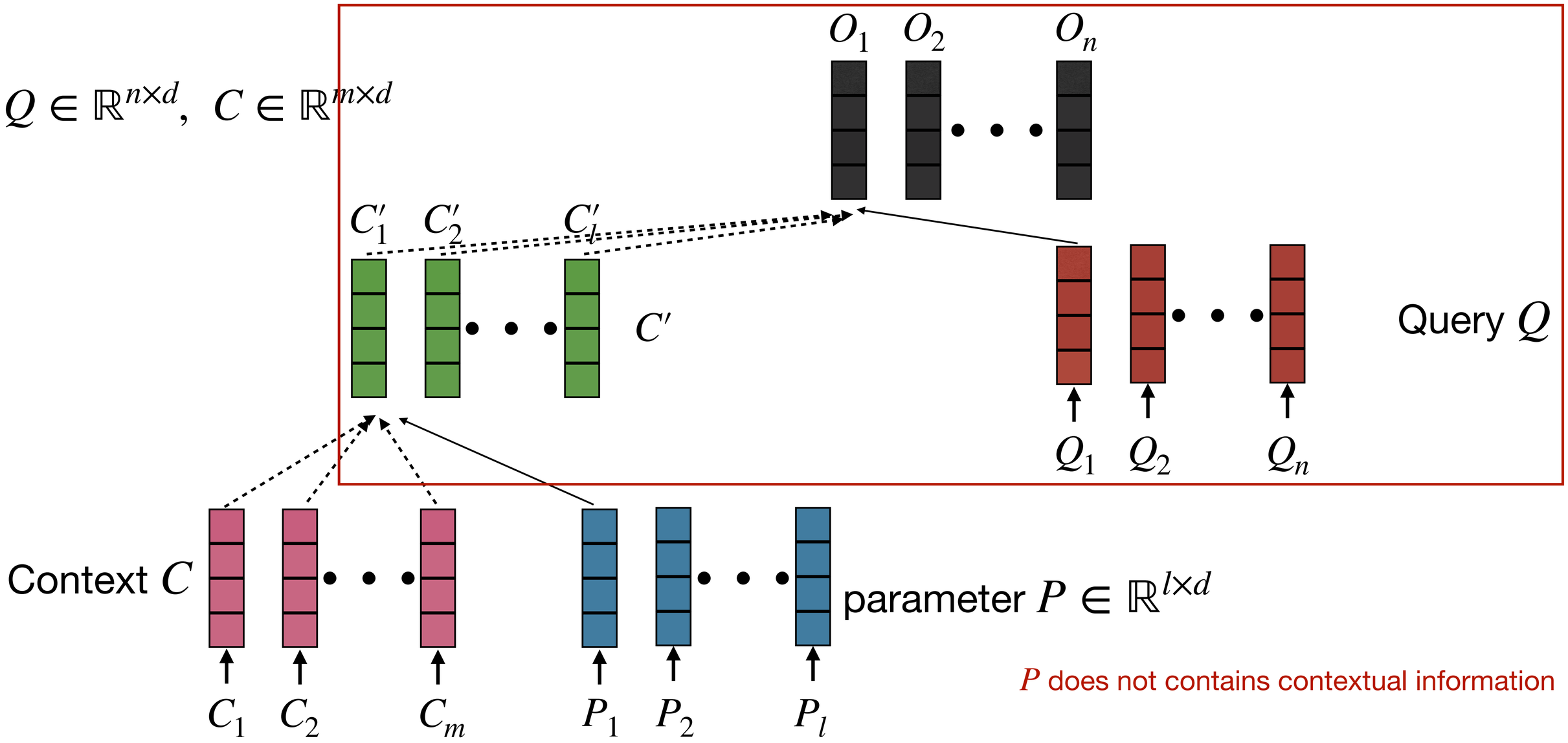
$$O = \underbrace{\sigma\left(\frac{Q(C')^T}{\sqrt{d}}\right)}_{n\times l} \underbrace{C'}_{l\times d}$$

Linear Nested Attention: Pack & Unpack

$Q \in \mathbb{R}^{n \times d},\ C \in \mathbb{R}^{m \times d}$



Linear Nested Attention: Pack & Unpack



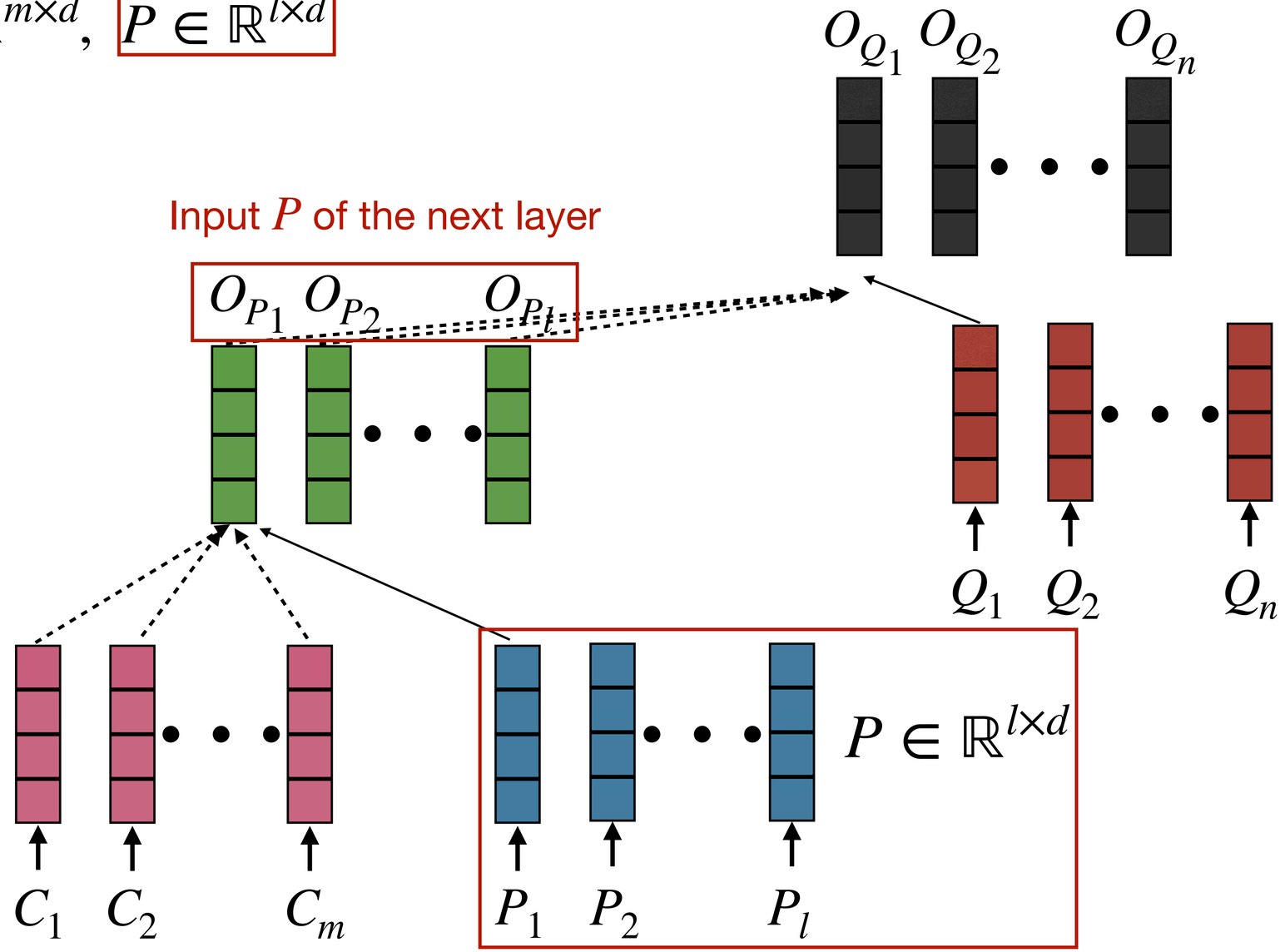
Luna: Linear Unified Nested Attention (Ma, et al., 2021)

Inputs: $Q \in \mathbb{R}^{n \times d}$, $C \in \mathbb{R}^{m \times d}$, $P \in \mathbb{R}^{l \times d}$

Outputs:

$$O_P = C' = \sigma\left(\frac{PC^T}{\sqrt{d}}\right)C$$

$$O_Q = \sigma\left(\frac{Q(C')^T}{\sqrt{d}}\right)C'$$

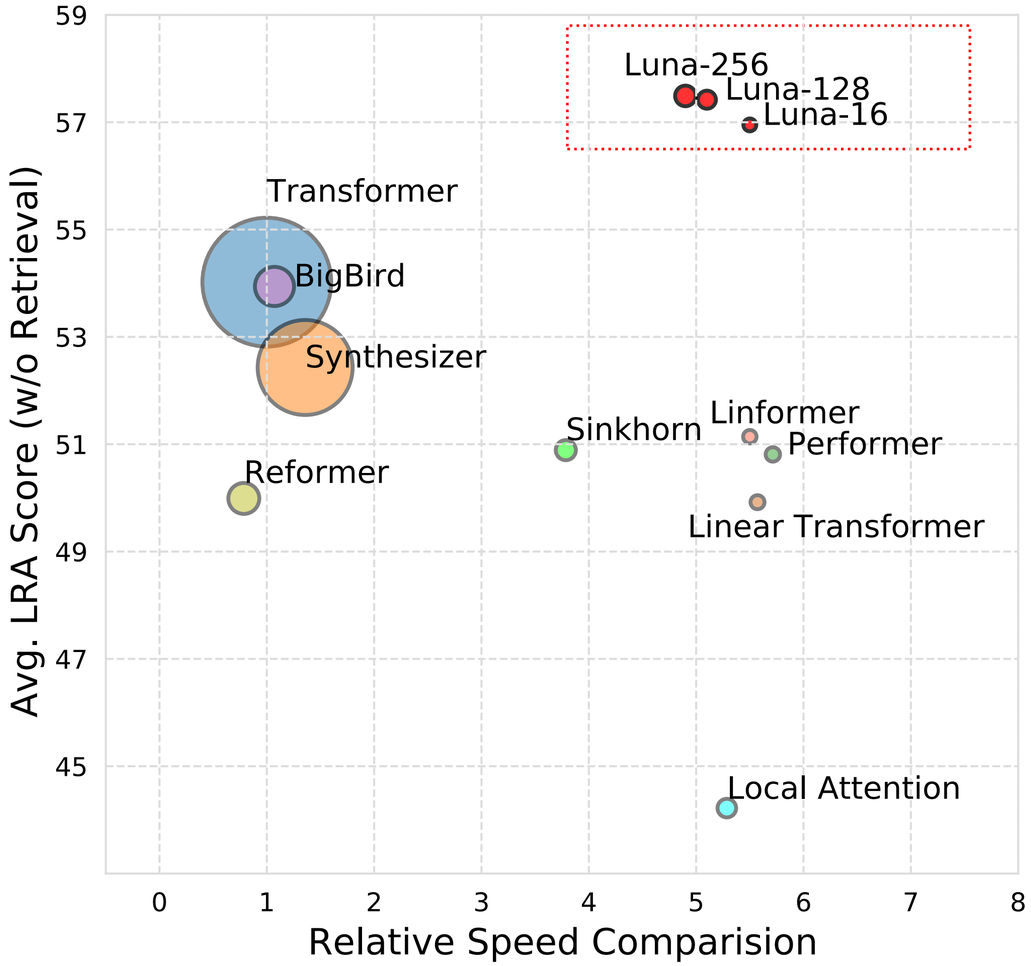


Complexity

- **Transformer**
  - Time: $O(n^2 d)$, Space: $O(n^2 + nd)$
- **Kernel Methods**
  - Time: $O(nkd)$, Space: $O(nk + kd + nd)$ (k is the number of kernel maps)
- **Luna**
  - Time: $O(nld)$, Space: $O(nl + ld + nd)$ ($l = |P|$ is the length of P)

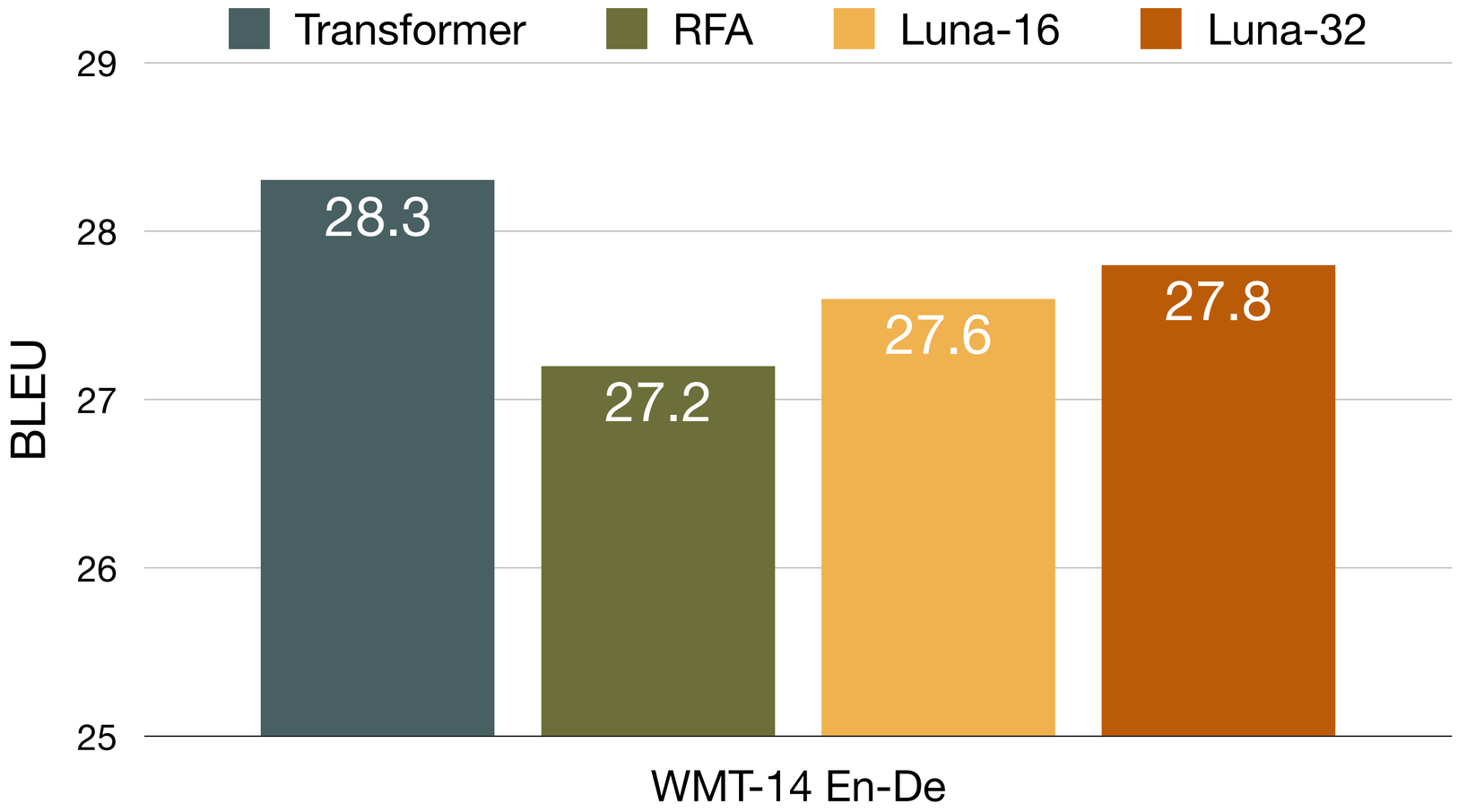
$|Q| = |C| = n$, d is model dimension

Performance on LRA



Trade-off between accuracy (y-axis), speed (x-axis) and memory (circle-radius)

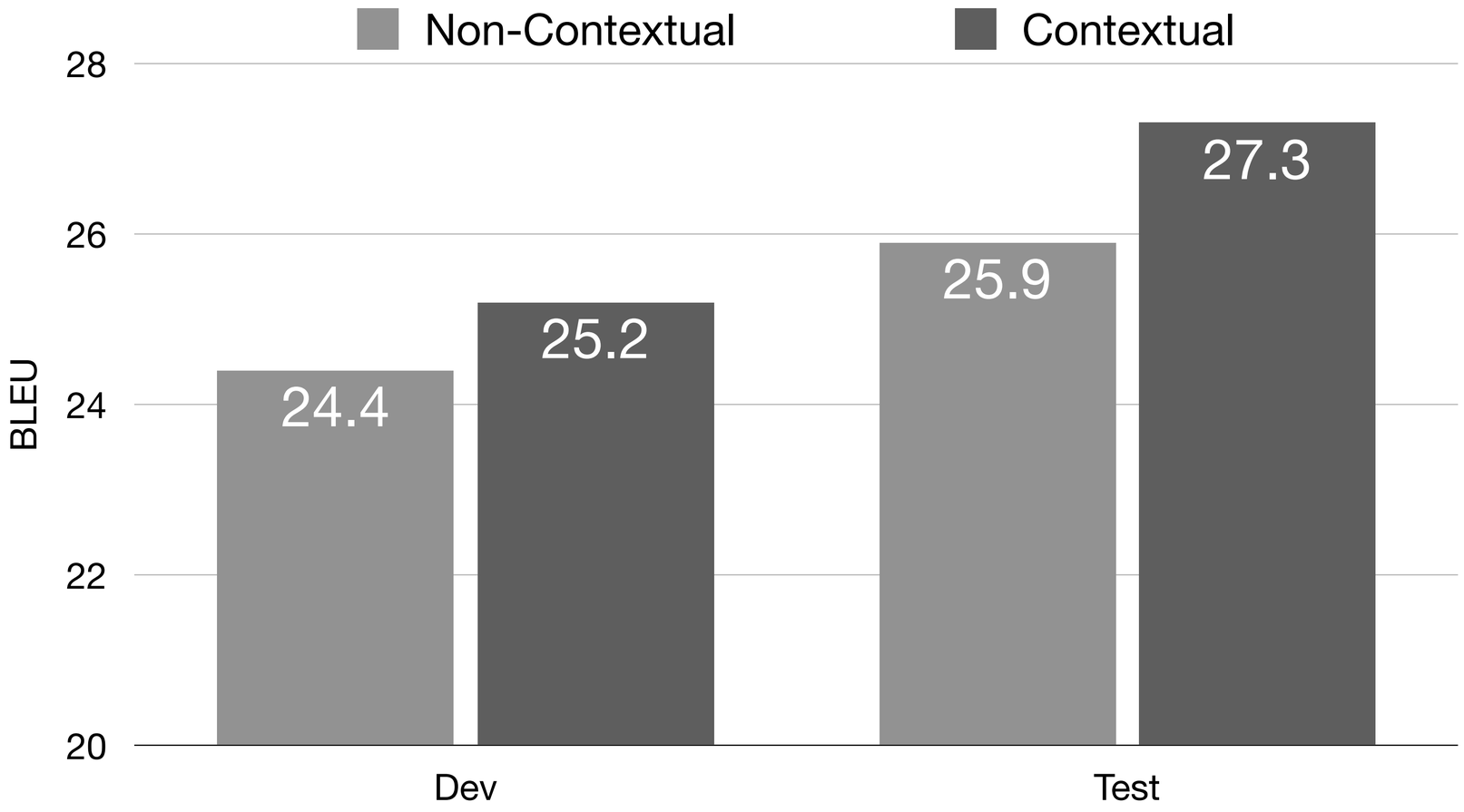
Results: Machine Translation



RFA: random feature attention (Peng et al., 2021)

All models were optimized with Apollo (Ma 2020)

Effect of Contextual P



All models were optimized with Apollo (Ma 2020)

Advances in Transformer & MT: Summary

- Semi-Supervised NMT
- Multilingual NMT
- Evaluation beyond BLEU
- Efficient Attention Mechanisms

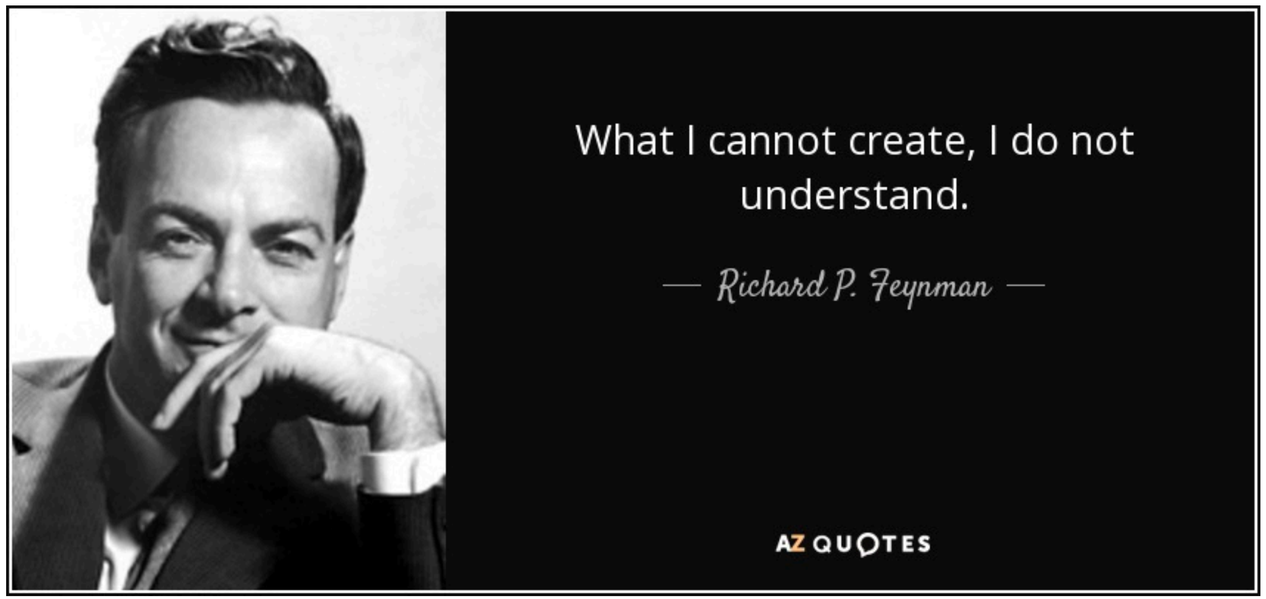
Deep Generative Models

Data Generation

- **What is Generation?**
  - Learning to generate new data from samples

Data Generation

- **Why Generation for Representation Learning?**
  - Requiring no labeled data
  - A good way to learn knowledge about data



Deep Generative Models

- **Distribution-based Generative Models**
  - Auto-regressive Neural Language Models
  - Generative (Normalizing) Flows
  - Variational Auto-Encoders (VAEs)
- **Non-distribution Generative Models**
  - Generative Adversarial Networks (GANs)

Deep Generative Models

- **Distribution-based Generative Models**

$$X \sim P(X)$$

$$P_\theta(X) \approx P(X)$$

Auto-regressive Generative Models

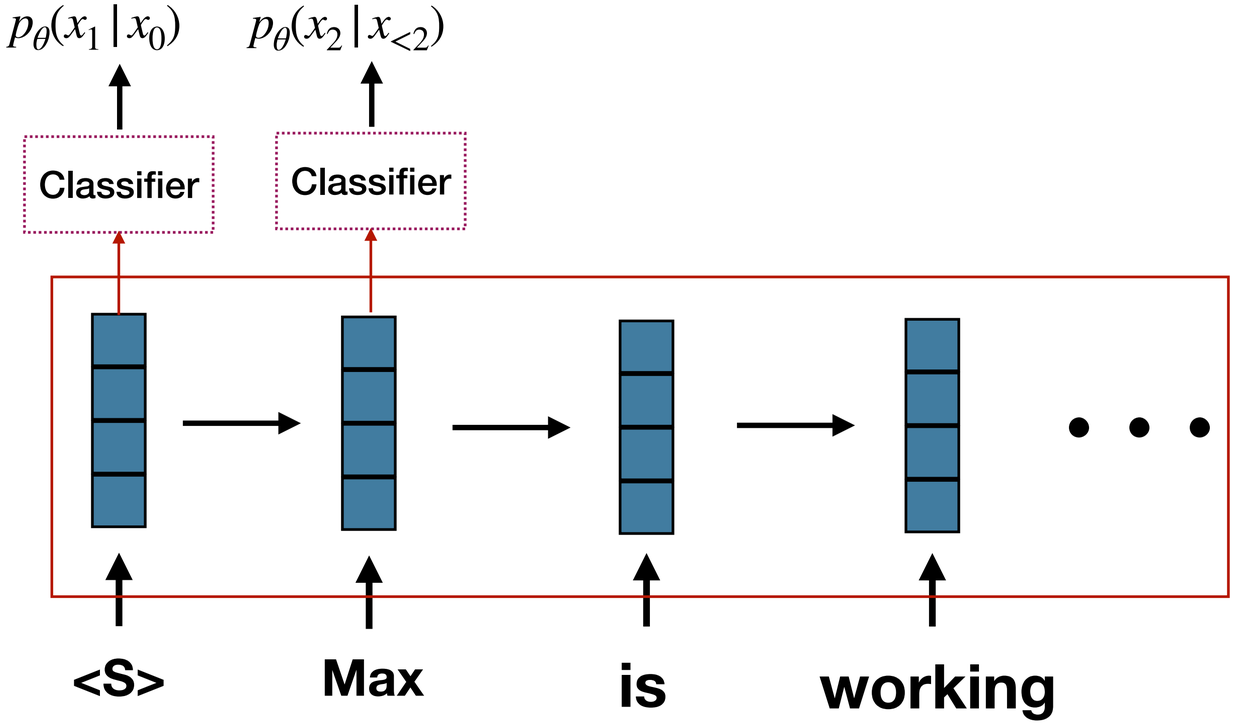
- **Auto-regressive Neural Language Models**

$$p_\theta(X) = \prod_{t=1}^{T} p_\theta(x_t \mid x_{<t})$$

Auto-regressive Generative Models

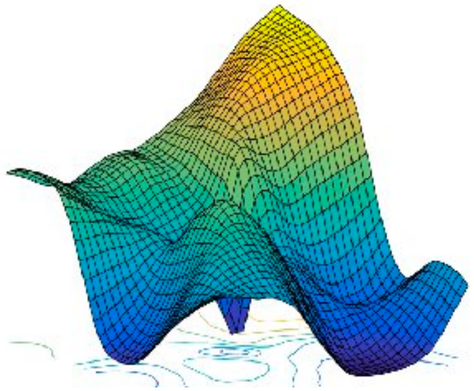
- **Auto-regressive Neural Language Models**

$$p_\theta(X) = \prod_{t=1}^{T} p_\theta(x_t \mid x_{<t})$$



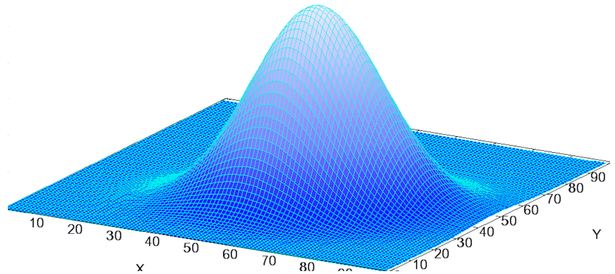
Transformer Decoder
Basic Architecture for GPT-2&3 (next lecture)

Generative (Normalizing) Flows



$$\Gamma = f_\theta(X)$$

$$X = f_\theta^{-1}(\Gamma)$$



$X \sim p_\theta(X)$

$\Gamma \sim \text{Normal}(0, I)$

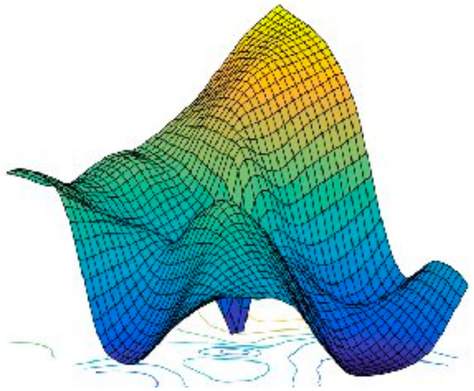
Change of Variable formula:

$$p_\theta(x) = p_\Gamma\left(f_\theta(x)\right) \left| \det \left(\frac{\partial f_\theta(x)}{\partial x} \right) \right|$$

Normal

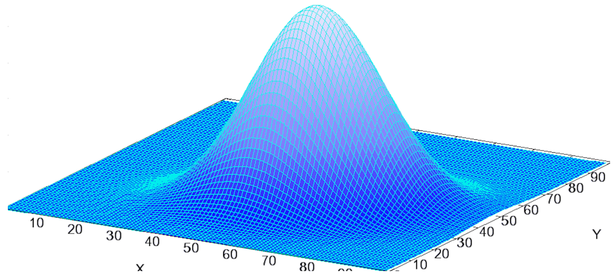
Jacobian Matrix

Generative (Normalizing) Flows



$$\Gamma = f_\theta(X)$$

$$X = f_\theta^{-1}(\Gamma)$$



$X \sim p_\theta(X)$

$\Gamma \sim \text{Normal}(0, I)$

Change of Variable formula:

$$p_\theta(x) = p_\Gamma\left(f_\theta(x)\right)\left|\det\left(\frac{\partial f_\theta(x)}{\partial x}\right)\right|$$

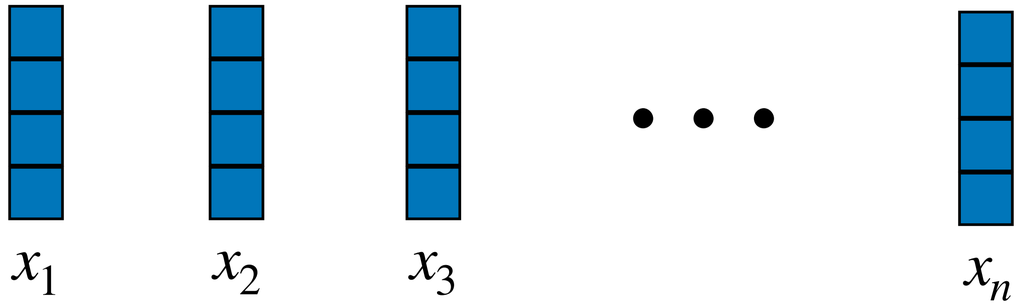
Generative Flow: A series of such f

$$X \underset{g_1}{\overset{f_1}{\longleftrightarrow}} H_1 \underset{g_2}{\overset{f_2}{\longleftrightarrow}} H2 \underset{g_3}{\overset{f_3}{\longleftrightarrow}} \cdots \underset{g_K}{\overset{f_K}{\longleftrightarrow}} \Gamma$$

Generative (Normalizing) Flows

- **ActNorm**

$$y_i = s \odot x_i + b$$



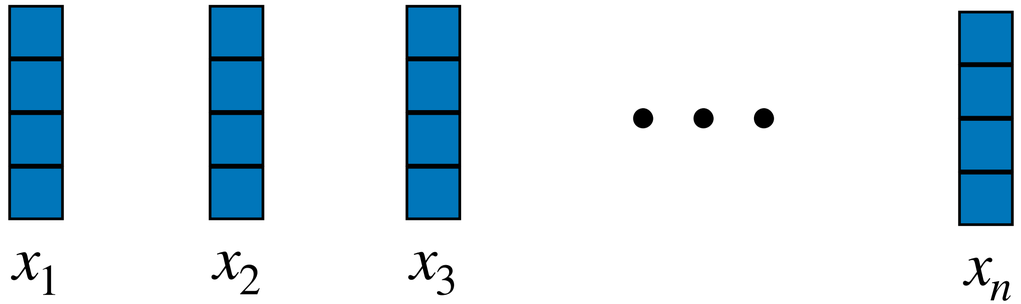
Generative (Normalizing) Flows

- **ActNorm**

$$y_i = s \odot x_i + b$$

- **Invertible Linear**

$$y_i = W x_i$$



Generative (Normalizing) Flows

- **ActNorm**

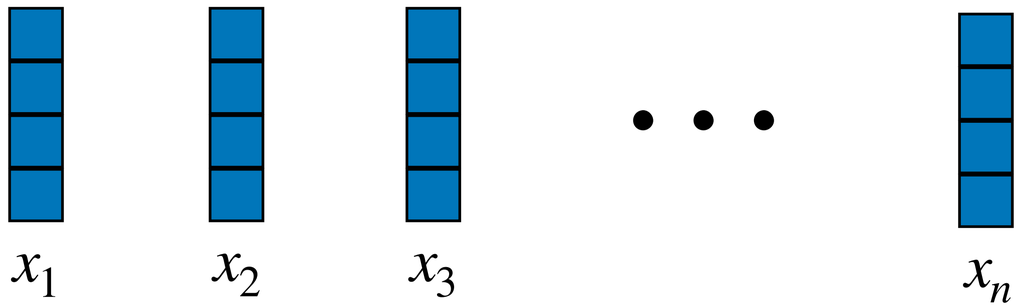
$$y_i = s \odot x_i + b$$

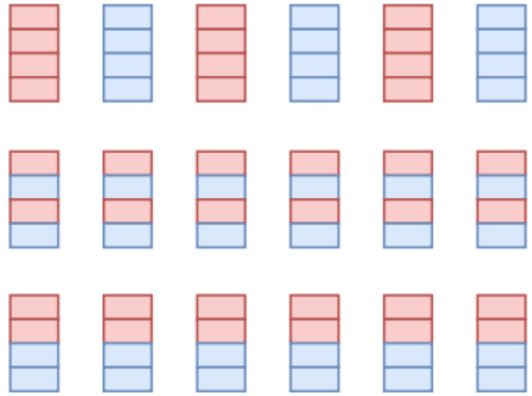
- **Invertible Linear**

$$y_i = W x_i$$

- **Affine Coupling**

$$\begin{aligned} x_a, x_b &= \text{split}(x) \\ y_a &= x_a \\ y_b &= \text{s}(x_a) \odot x_b + \text{b}(x_a) \\ y &= \text{concat}(y_a, y_b), \end{aligned}$$





Generative (Normalizing) Flows: Pros and Cons

- Modeling the exact distribution $P_\theta(X)$
- No auto-regressive factorization

- A large number of layers: invertible function f_i is very weak
- Determinant calculation is expensive

Variational Auto-Encoders (VAEs)

- **Learning Latent Representations**

X

Z

low-dimensional

Classification

Clustering

Retrieval

……

Variational Auto-Encoders (VAEs)

- **Latent random variable Z**
- **Marginal Likelihood:**

$$p_\theta(X) = \boxed{\int_Z p_\theta(X \mid Z) p_\theta(Z) dz,}$$

- How to compute the integral?
  - Variational Inference

Variational Inference

$$\underbrace{\log p_\theta(X)}_{\textbf{LL}} = \log \int_Z p_\theta(X|Z) p_\theta(Z) dz$$

Evidence Lower Bound (ELBO)

$$\geq \underbrace{\mathrm{E}_{q_\phi(Z|X)}[\log p_\theta(X|Z)] - \mathrm{KL}(q_\phi(Z|X) \,||\, p_\theta(Z))}_{\textbf{ELBO}}$$

Posterior **Generator** **Prior**

Variational Inference

$$\underbrace{\log p_\theta(X)}_{\textbf{LL}} = \log \int_Z p_\theta(X|Z) p_\theta(Z) dz$$

Evidence Lower Bound (ELBO)

$$\geq \underbrace{\mathrm{E}_{q_\phi(Z|X)}[\log p_\theta(X|Z)] - D_{\mathrm{KL}}(q_\phi(Z|X)||p_\theta(Z))}_{\textbf{ELBO}}$$

$$\underbrace{\mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})]}_{\text{Reconstruction Loss}} - \underbrace{D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))}_{\text{KL Regularizer}}$$

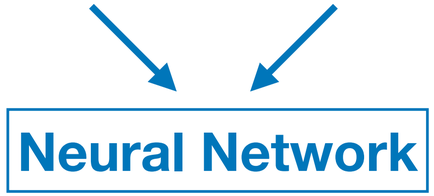
Variational Auto-Encoders (VAEs)

- **Prior**

$$P(Z) \sim \mathcal{N}(0, I)$$

- **Posterior**

$$q_\phi(Z|X) \sim \mathcal{N}(\mu(X), \sigma^2(X))$$



- **Generater**

$$P_\theta(X|Z)$$

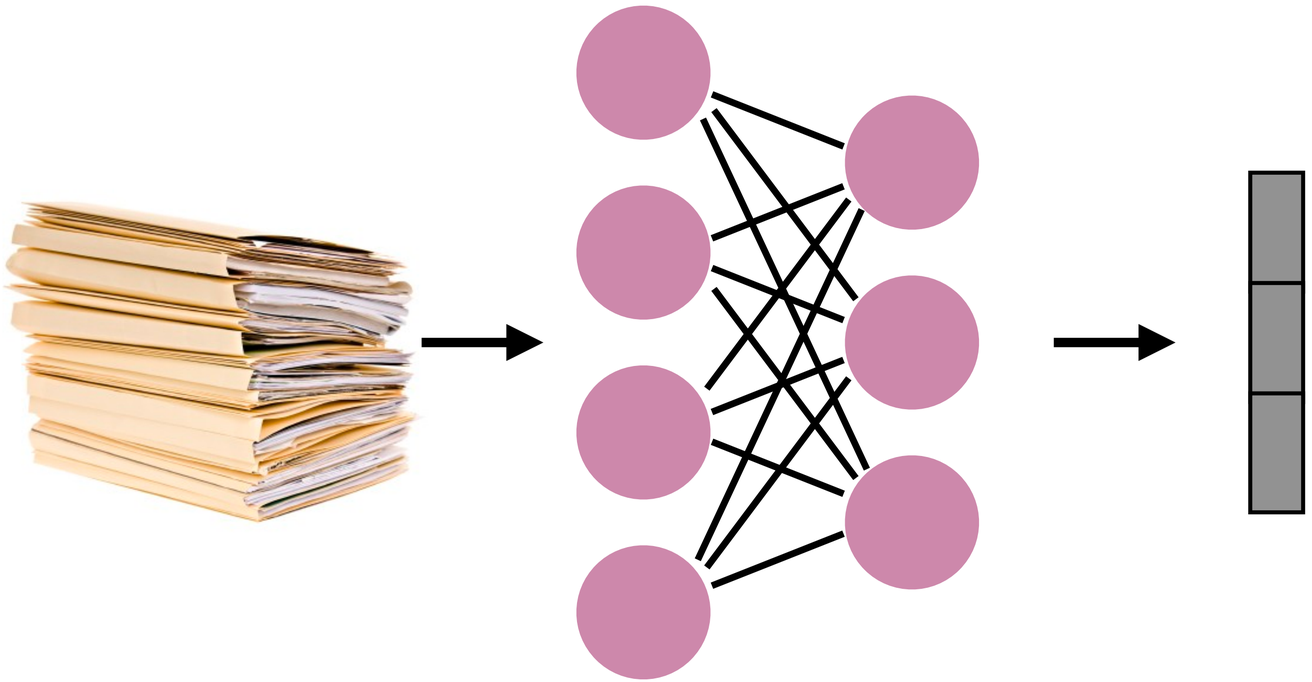
Depends on tasks:
Auto-regressive model for sequences
Generative flows
…

Variational Inference

Evidence Lower Bound (ELBO)

$$\underbrace{\log p_\theta(X)}_{\textbf{LL}} \geq \underbrace{\mathrm{E}_{q_\phi(Z|X)}[\log p_\theta(X|Z)] - \mathrm{KL}(q_\phi(Z|X)||p_\theta(Z))}_{\textbf{ELBO}}$$

Posterior



X $q_\phi(Z|X)$ Z

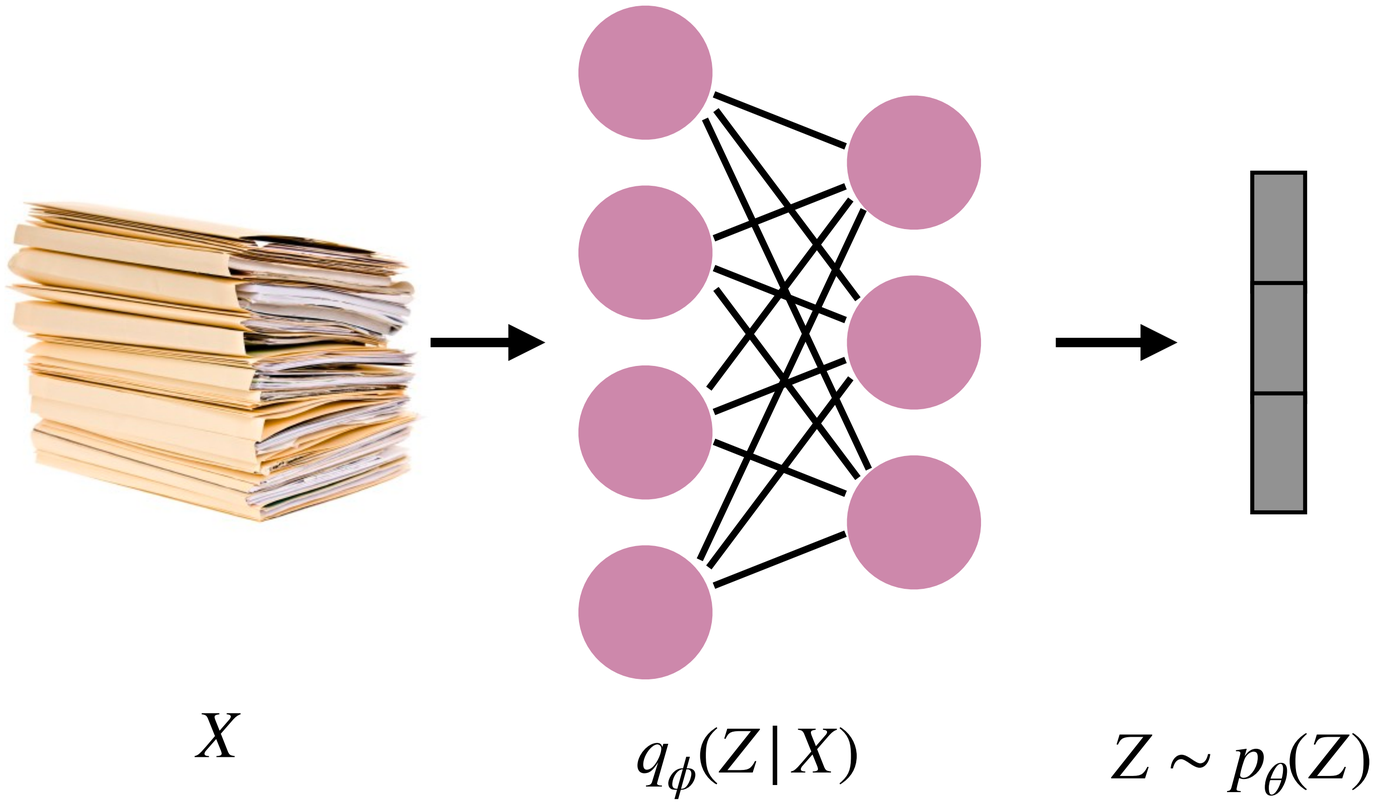
Variational Inference

Evidence Lower Bound (ELBO)

$$\underbrace{\log p_\theta(X)}_{\textbf{LL}} \geq \underbrace{\mathrm{E}_{q_\phi(Z|X)}[\log p_\theta(X|Z)] - \mathrm{KL}(q_\phi(Z|X)\,||\,p_\theta(Z))}_{\textbf{ELBO}}$$

Posterior

Prior



X

$q_\phi(Z|X)$

$Z \sim p_\theta(Z)$

Variational Inference

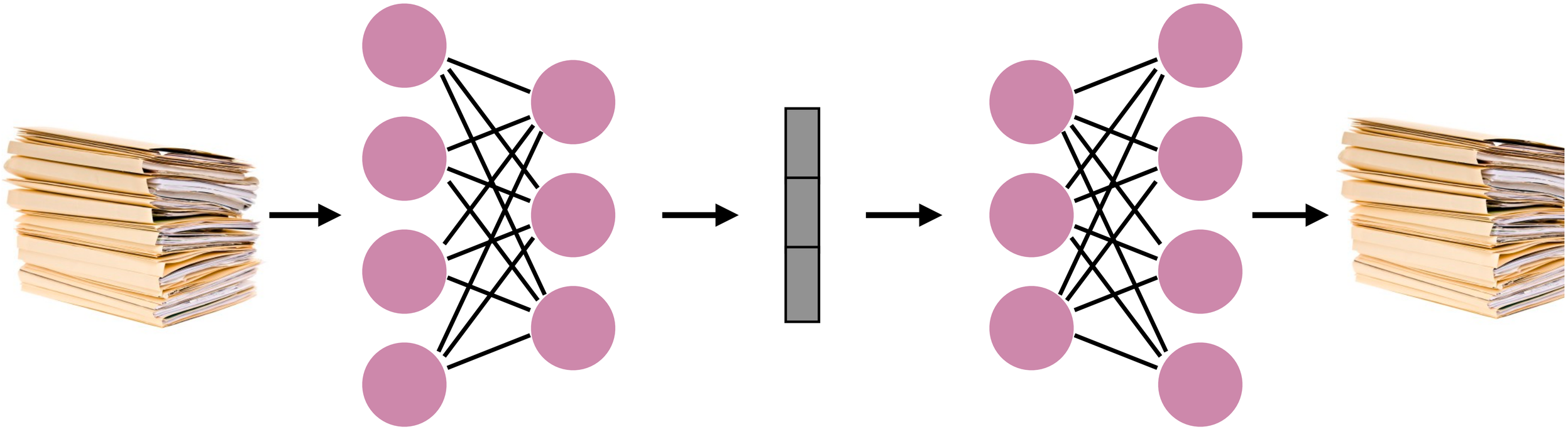
Evidence Lower Bound (ELBO)

$$\underbrace{\log p_\theta(X)}_{\textbf{LL}} \geq \underbrace{\mathrm{E}_{q_\phi(Z|X)}[\log p_\theta(X|Z)] - \mathrm{KL}(q_\phi(Z|X)||p_\theta(Z))}_{\textbf{ELBO}}$$

Posterior

Prior

Generator



X

$q_\phi(Z|X)$

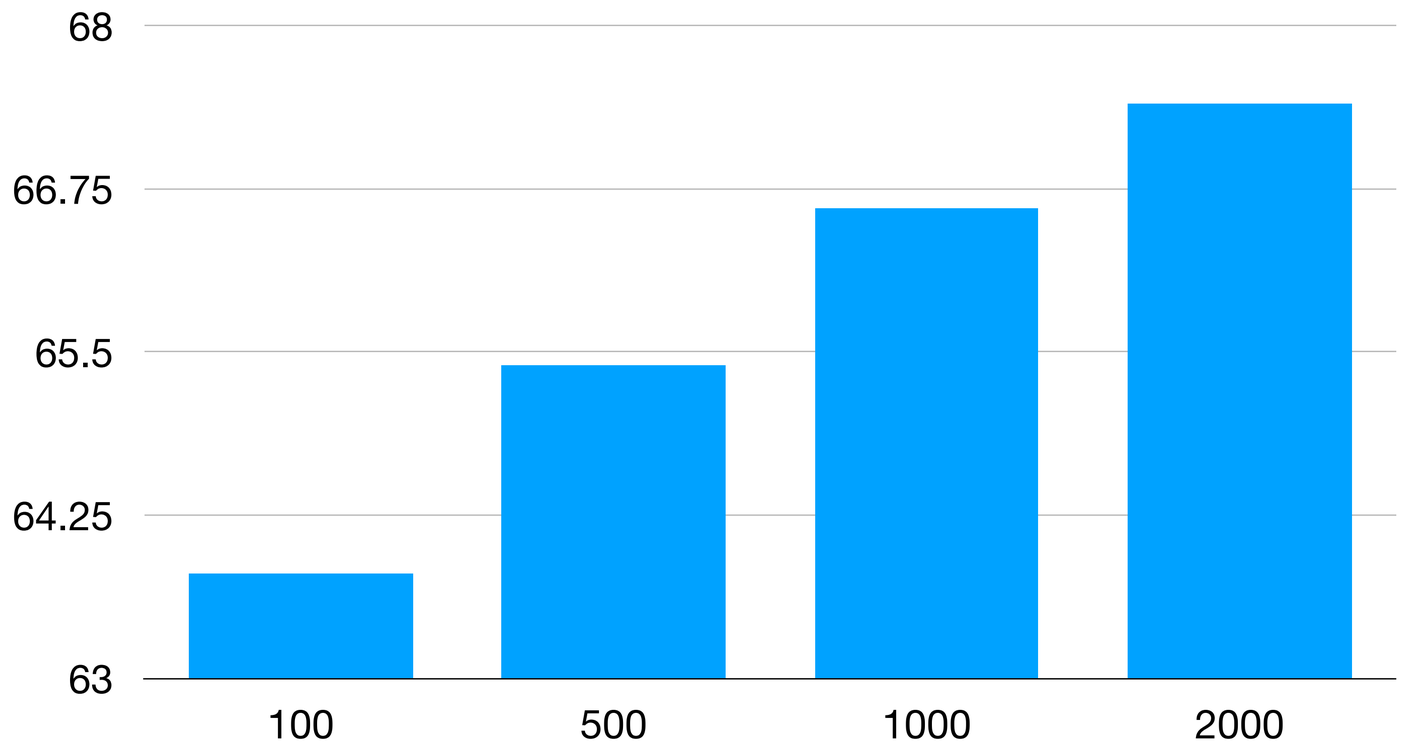
$Z \sim p_\theta(Z)$

$p_\theta(X|Z)$

$\tilde{X}$

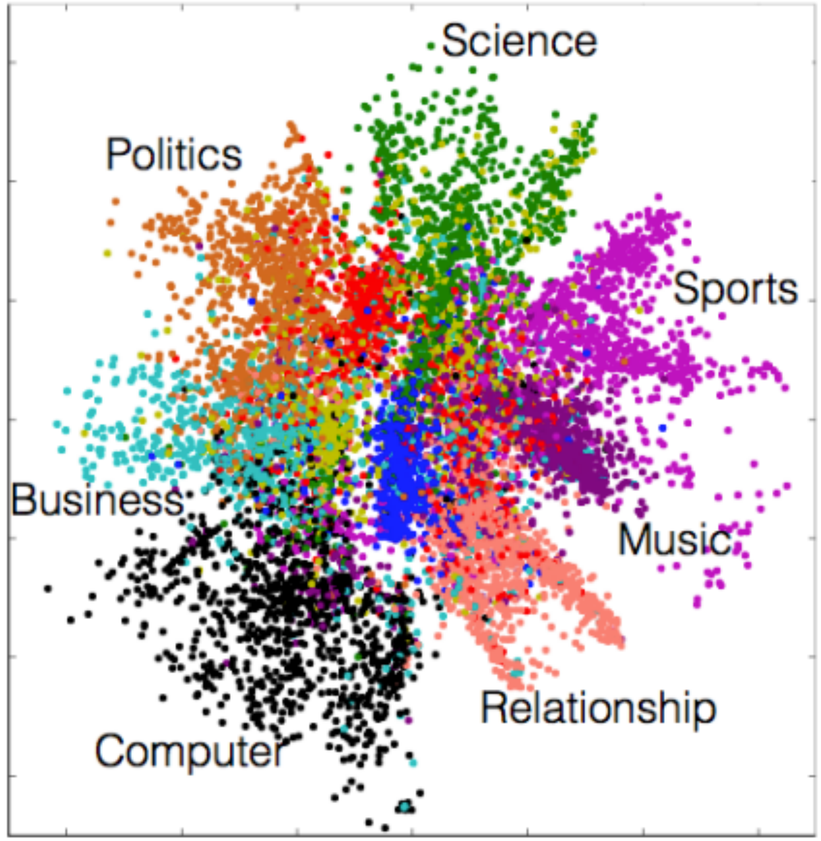
Variational Auto-Encoders (VAEs)

- **Text Classification**
  - Training a VAE
  - Using Z as features to train a linear classifier with a small number of data
  - Dataset: Yahoo Answers (10 classes)



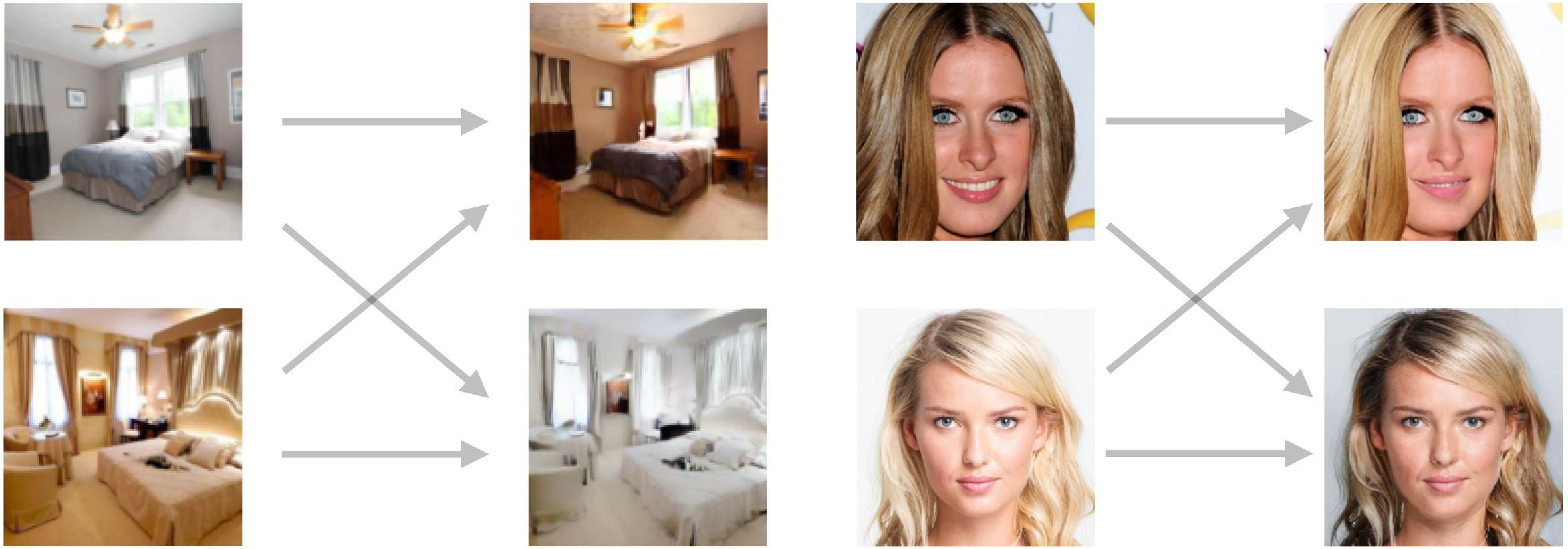
Variational Auto-Encoders (VAEs)

- **Text Classification**
  - Training a VAE
  - Using Z as features to train a linear classifier with a small number of data
  - Dataset: Yahoo Answers (10 classes)



VAEs & Flows: Advanced Applications

- **Advanced Representation Learning**
  - Non-autoregressive NMT
  - Decoupled and/or disentangled representations



Reading Materials

- **Relavant Papers**
  - Non-Autoregressive NMT
  - Decoupled Representations
  - Efficient Attention Mechanims