

DSC291: Advanced Statistical Natural Language Processing

Unsupervised Learning Classification

Zhiting Hu

Lecture 9, April 21, 2022

UC San Diego

HALICIOĞLU DATA SCIENCE INSTITUTE

Outline

- Representation learning
 - Variational Autoencoders (VAEs)
- Classification
 - Augmentation
 - Prompting

EM Algorithm

- Observed variables \mathbf{x} , latent variables \mathbf{z}
- To learn a model $p(\mathbf{x}, \mathbf{z}|\theta)$, we want to maximize the marginal log-likelihood

$$\ell(\theta; \mathbf{x}) = \log p(\mathbf{x}|\theta) = \log \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}|\theta)$$

- But it's too difficult
- EM algorithm:
 - maximize a lower bound of $\ell(\theta; \mathbf{x})$
 - Or equivalently, minimize an upper bound of $\ell(\theta; \mathbf{x})$
- Key equation:

$$\ell(\theta; \mathbf{x}) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \left[\log \frac{p(\mathbf{x}, \mathbf{z}|\theta)}{q(\mathbf{z}|\mathbf{x})} \right] + \text{KL}(q(\mathbf{z}|\mathbf{x}) || p(\mathbf{z}|\mathbf{x}, \theta))$$

EM Algorithm

- Observed variables \mathbf{x} , latent variables \mathbf{z}
- To learn a model $p(\mathbf{x}, \mathbf{z}|\theta)$, we want to maximize the marginal log-likelihood

$$\ell(\theta; \mathbf{x}) = \log p(\mathbf{x}|\theta) = \log \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}|\theta)$$

- But it's too difficult
- EM algorithm:
 - maximize a lower bound of $\ell(\theta; \mathbf{x})$
 - Or equivalently, minimize an upper bound of $\ell(\theta; \mathbf{x})$
- Key equation:

$$\ell(\theta; \mathbf{x}) = \underbrace{\mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \left[\log \frac{p(\mathbf{x}, \mathbf{z}|\theta)}{q(\mathbf{z}|\mathbf{x})} \right]}_{\text{Evidence Lower Bound (ELBO)}} + \text{KL}(q(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z}|\mathbf{x}, \theta))$$

$$= \underbrace{-F(q, \theta)}_{\text{Variational free energy}} + \text{KL}(q(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z}|\mathbf{x}, \theta))$$

Variational free energy

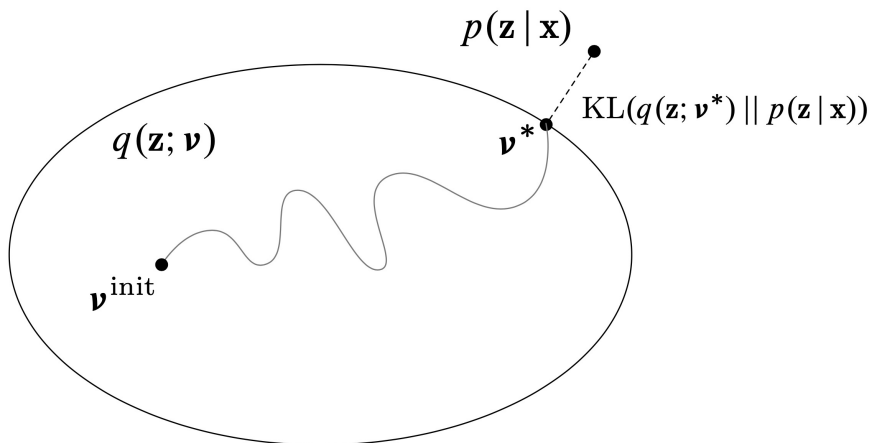
EM Algorithm

- The EM algorithm is coordinate-descent on $F(q, \theta)$
 - E-step: $q^{t+1} = \arg \min_q F(q, \theta^t) = p(\mathbf{z}|\mathbf{x}, \theta^t)$
 - the posterior distribution over the latent variables given the data and the current parameters
 - M-step: $\theta^{t+1} = \arg \min_{\theta} F(q^{t+1}, \theta) = \operatorname{argmax}_{\theta} \sum_{\mathbf{z}} q^{t+1}(\mathbf{z}|\mathbf{x}) \log p(\mathbf{x}, \mathbf{z}|\theta)$

$$\begin{aligned} \ell(\theta; \mathbf{x}) &= \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \left[\log \frac{p(\mathbf{x}, \mathbf{z}|\theta)}{q(\mathbf{z}|\mathbf{x})} \right] + \text{KL}(q(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z}|\mathbf{x}, \theta)) \\ &= -F(q, \theta) + \text{KL}(q(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z}|\mathbf{x}, \theta)) \end{aligned}$$

Quick Recap

- We often cannot compute posteriors $p(\mathbf{z}|\mathbf{x}, \theta)$, and so we need to approximate them, using variational methods.
- In variational Bayes, we'd like to find an approximation within some family that minimizes the KL divergence to the posterior, but we can't directly minimize this
- Therefore, we defined the ELBO, which we can maximize, and this is equivalent to minimizing the KL divergence.



Evidence Lower Bound (ELBO)

$$\ell(\theta; \mathbf{x}) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \left[\log \frac{p(\mathbf{x}, \mathbf{z}|\theta)}{q(\mathbf{z}|\mathbf{x})} \right] + \text{KL}(q(\mathbf{z}|\mathbf{x}) || p(\mathbf{z}|\mathbf{x}, \theta))$$

Black-box Variational Inference (BBVI)

- Variational distribution $q_{\lambda}(\mathbf{z}|\mathbf{x})$ with parameters λ , e.g.,
 - Gaussian mixture distribution:
 - Deep neural networks

- ELBO:

$$\mathcal{L}(\lambda) = \mathbb{E}_{q(\mathbf{z}|\lambda)}[\log p(\mathbf{x}, \mathbf{z})] - \mathbb{E}_{q(\mathbf{z}|\lambda)}[\log q(\mathbf{z}|\lambda)]$$

- Want to compute the gradient w.r.t variational parameters λ
- Need stochastic gradient estimates
 - The score function estimator (a.k.a log-derivative trick, REINFORCE)
 - The reparameterization trick (a.k.a the pathwise gradient estimator)

Computing Gradients of Expectations w/ reparametrization trick

- Loss: $\mathcal{L} = \mathbb{E}_{q_\lambda(\mathbf{z})}[f_\lambda(\mathbf{z})]$
- Assume that we can express the distribution $q_\lambda(\mathbf{z})$ with a transformation

$$\begin{aligned} \epsilon &\sim s(\epsilon) \\ \mathbf{z} &= t(\epsilon, \lambda) \end{aligned} \iff \mathbf{z} \sim q(\mathbf{z}|\lambda)$$

- E.g.,

$$\begin{aligned} \epsilon &\sim \text{Normal}(0, 1) \\ \mathbf{z} &= \epsilon\sigma + \mu \end{aligned} \iff \mathbf{z} \sim \text{Normal}(\mu, \sigma^2)$$

- Reparameterization gradient

$$\mathcal{L} = \mathbb{E}_{\epsilon \sim s(\epsilon)}[f_\lambda(\mathbf{z}(\epsilon, \lambda))]$$

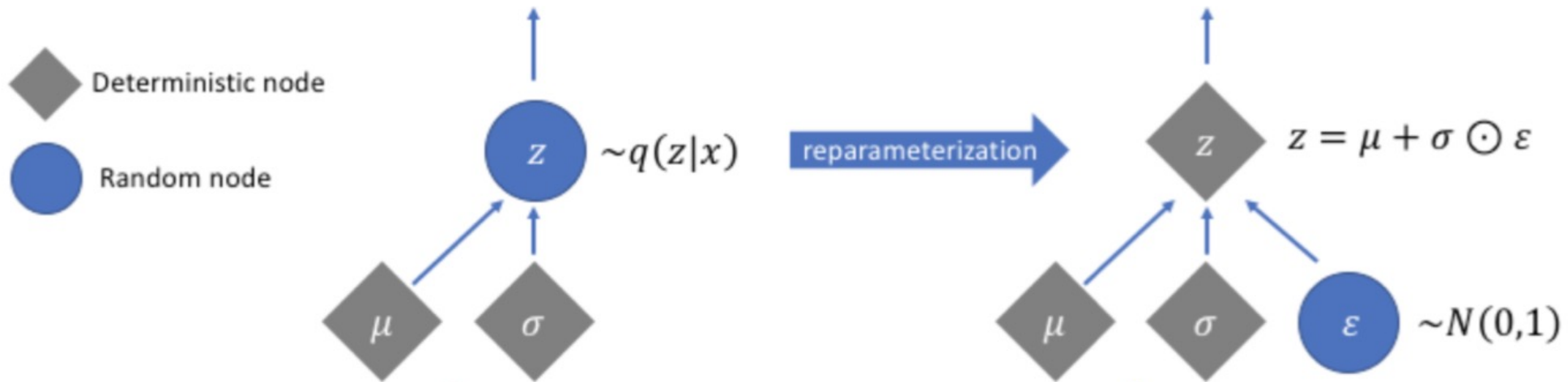
$$\nabla_\lambda \mathcal{L} = \mathbb{E}_{\epsilon \sim s(\epsilon)}[\nabla_{\mathbf{z}} f_\lambda(\mathbf{z}) \nabla_\lambda t(\epsilon, \lambda)]$$

- Pros: empirically, lower variance of the gradient estimate
- Cons: Not all distributions can be reparameterized

Reparameterization trick

- Reparameterizing Gaussian distribution

$$\begin{aligned} \epsilon &\sim \text{Normal}(0, 1) \\ z &= \epsilon\sigma + \mu \end{aligned} \iff z \sim \text{Normal}(\mu, \sigma^2)$$



BBVI with the reparameterization gradient

- ELBO:

$$\mathcal{L}(\lambda) = \mathbb{E}_{q(\mathbf{z}|\lambda)}[\log p(\mathbf{x}, \mathbf{z})] - \mathbb{E}_{q(\mathbf{z}|\lambda)}[\log q(\mathbf{z}|\lambda)]$$

- Gradient w.r.t. λ

$$\begin{aligned} \epsilon &\sim s(\epsilon) \\ z &= t(\epsilon, \lambda) \end{aligned} \iff z \sim q(z|\lambda)$$

$$\nabla_{\lambda} \mathcal{L} = \mathbb{E}_{\epsilon \sim s(\epsilon)} [\nabla_{\mathbf{z}} [\log p(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z})] \nabla_{\lambda} t(\epsilon, \lambda)]$$

Variational Autoencoders (VAEs)

Variational Auto-Encoders (VAEs)

VAEs are a combination of the following ideas:

- Variational Inference
 - ELBO
- Variational distribution parametrized as neural networks
- Reparameterization trick

Variational Auto-Encoders (VAEs)

- Model $p_{\theta}(\mathbf{x}, \mathbf{z}) = p_{\theta}(\mathbf{x}|\mathbf{z})p(\mathbf{z})$
 - $p_{\theta}(\mathbf{x}|\mathbf{z})$: a.k.a., generative model, generator, (probabilistic) decoder, ...
 - $p(\mathbf{z})$: prior, e.g., Gaussian
- Assume variational distribution $q_{\phi}(\mathbf{z}|\mathbf{x})$
 - E.g., a Gaussian distribution parameterized as **deep neural networks**
 - a.k.a, recognition model, inference network, (probabilistic) encoder, ...
- ELBO:

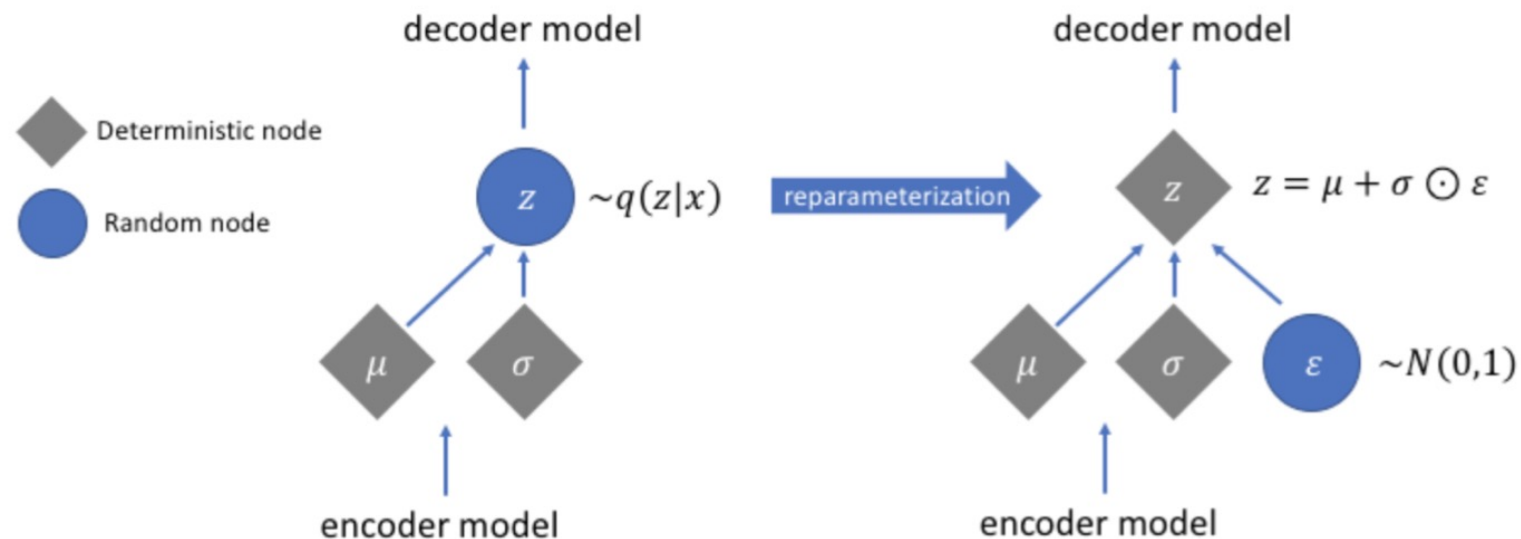
$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) &= \mathbb{E}_{q_{\phi}(\mathbf{z}|\mathbf{x})}[\log p_{\theta}(\mathbf{x}, \mathbf{z})] - H(q_{\phi}(\mathbf{z}|\mathbf{x})) \\ &= \mathbb{E}_{q_{\phi}(\mathbf{z}|\mathbf{x})}[\log p_{\theta}(\mathbf{x}|\mathbf{z})] - \text{KL}(q_{\phi}(\mathbf{z}|\mathbf{x}) || p(\mathbf{z}))\end{aligned}$$

Reconstruction

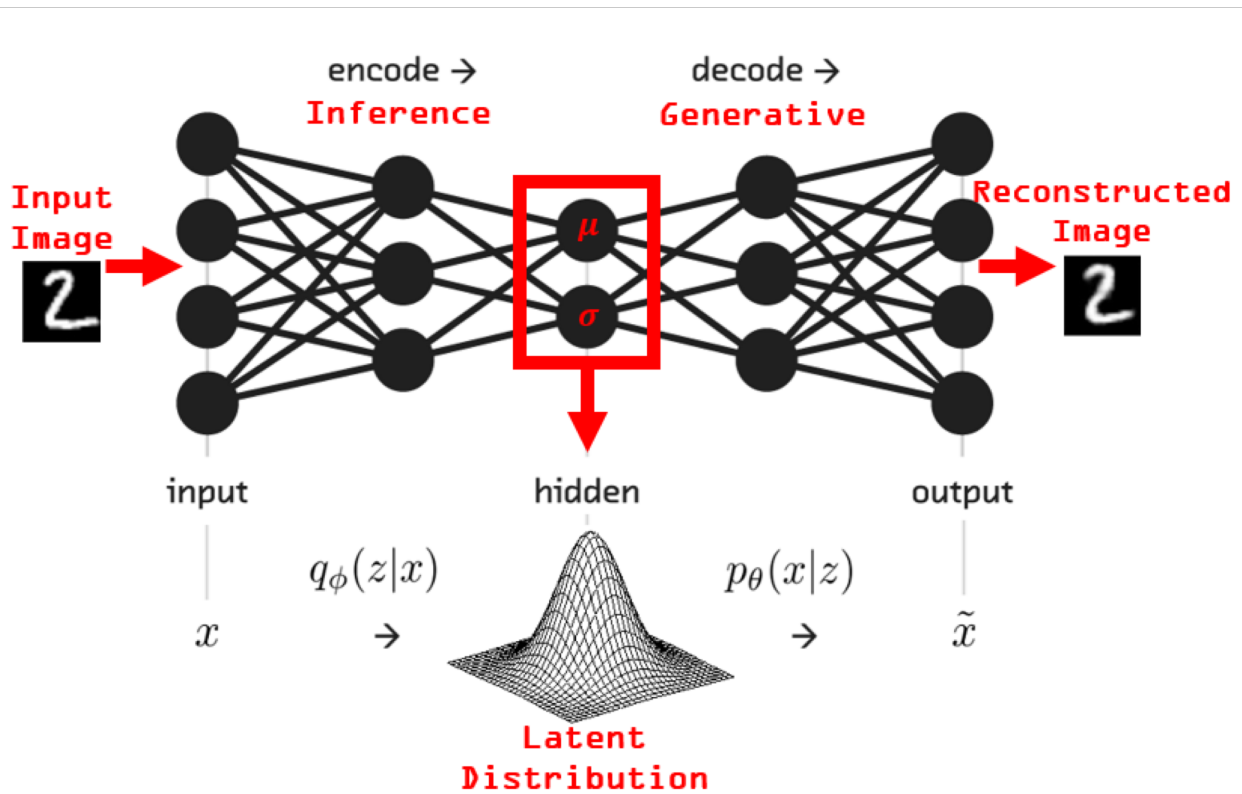
Divergence from prior
(KL divergence between two Gaussians
has an analytic form)

Variational Auto-Encoders (VAEs)

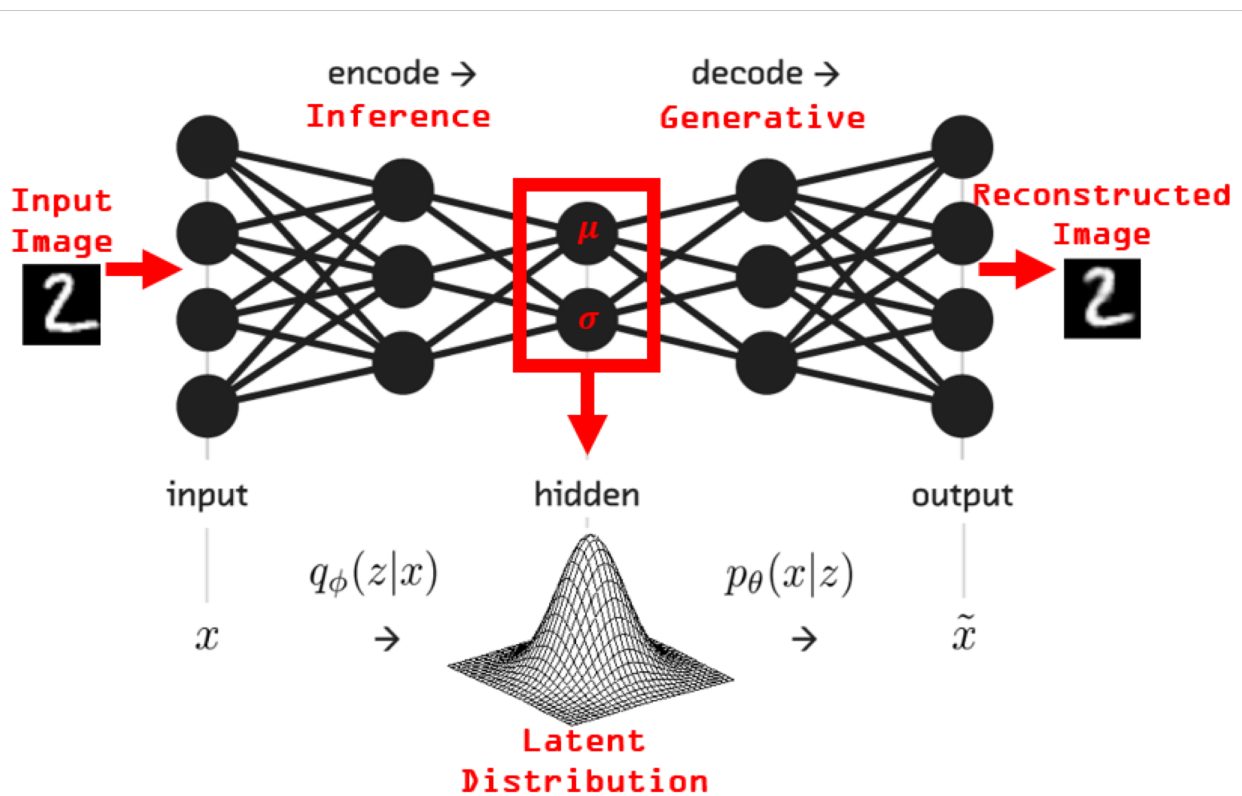
- ELBO:
$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} [\log p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})] - \text{H}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})) \\ &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} [\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] - \text{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) || p(\boldsymbol{z}))\end{aligned}$$
- Reparameterization:
 - $[\boldsymbol{\mu}; \boldsymbol{\sigma}] = f_{\boldsymbol{\phi}}(\boldsymbol{x})$ (a neural network)
 - $\boldsymbol{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim N(\mathbf{0}, \mathbf{1})$



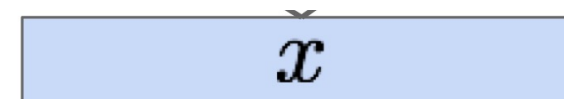
Example: VAEs for images



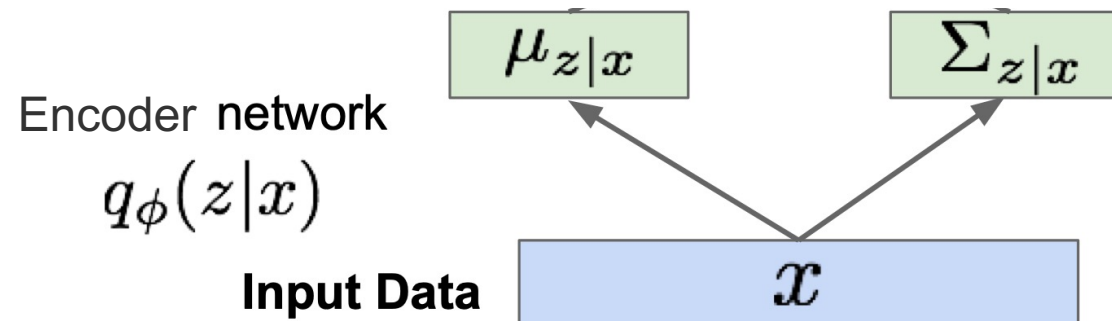
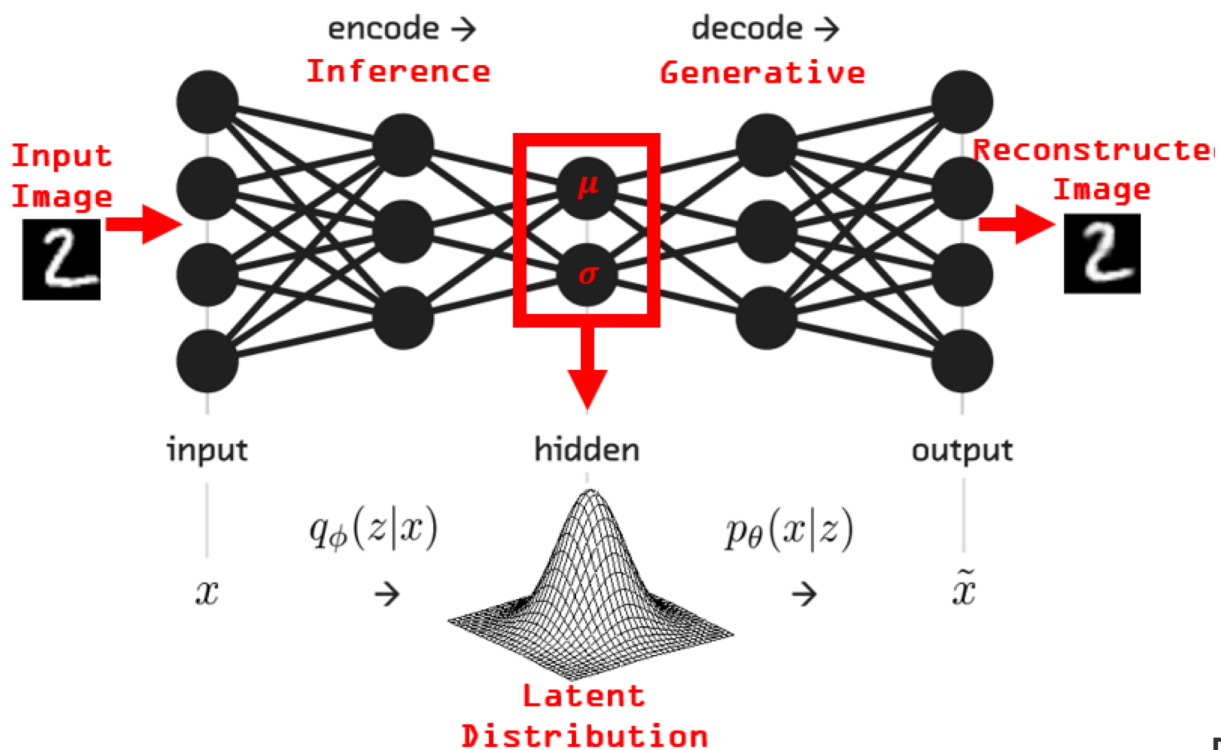
Example: VAEs for images



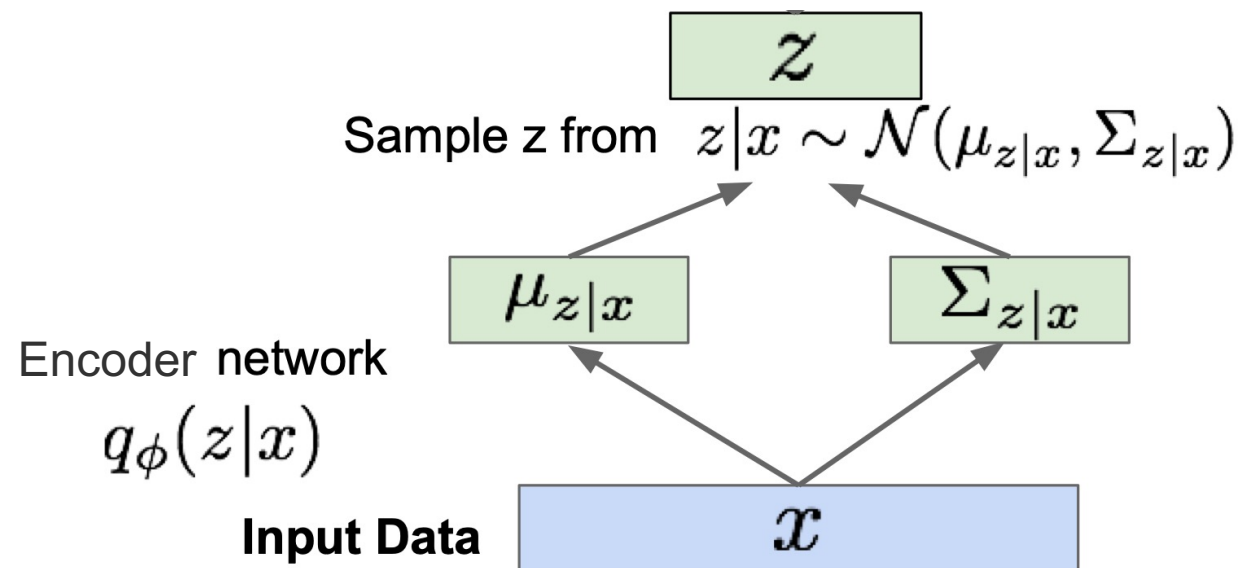
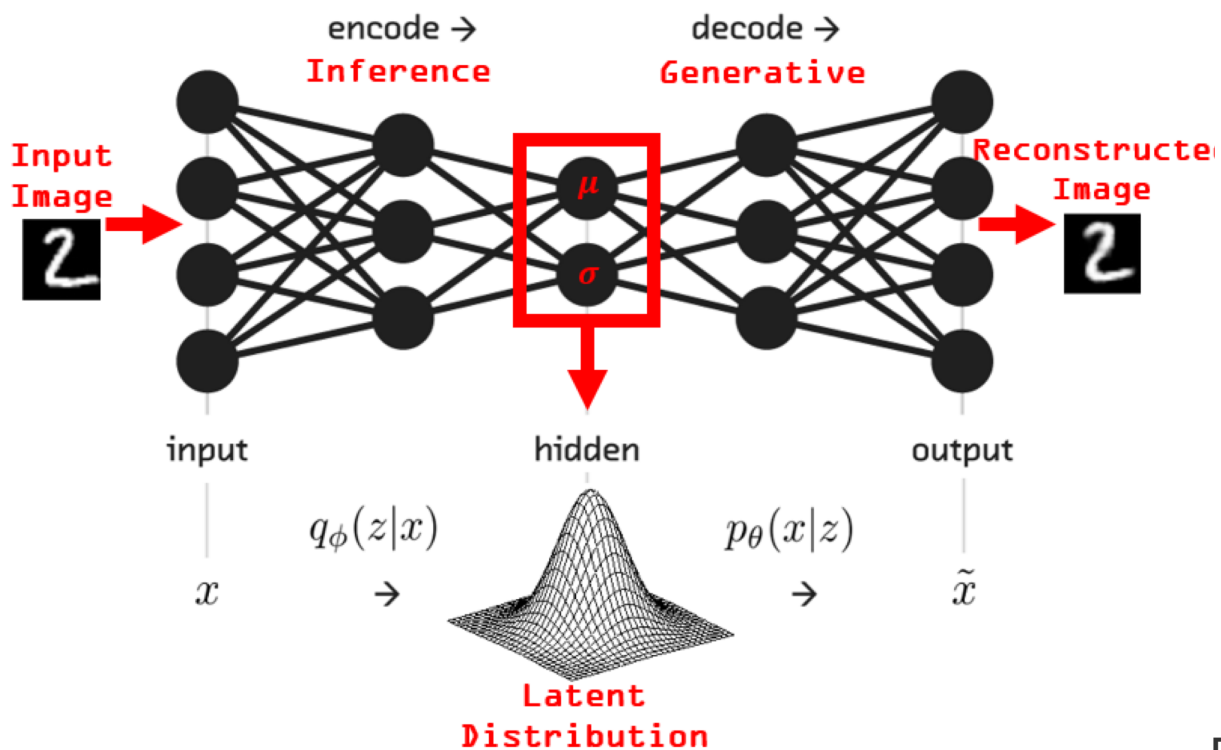
Input Data



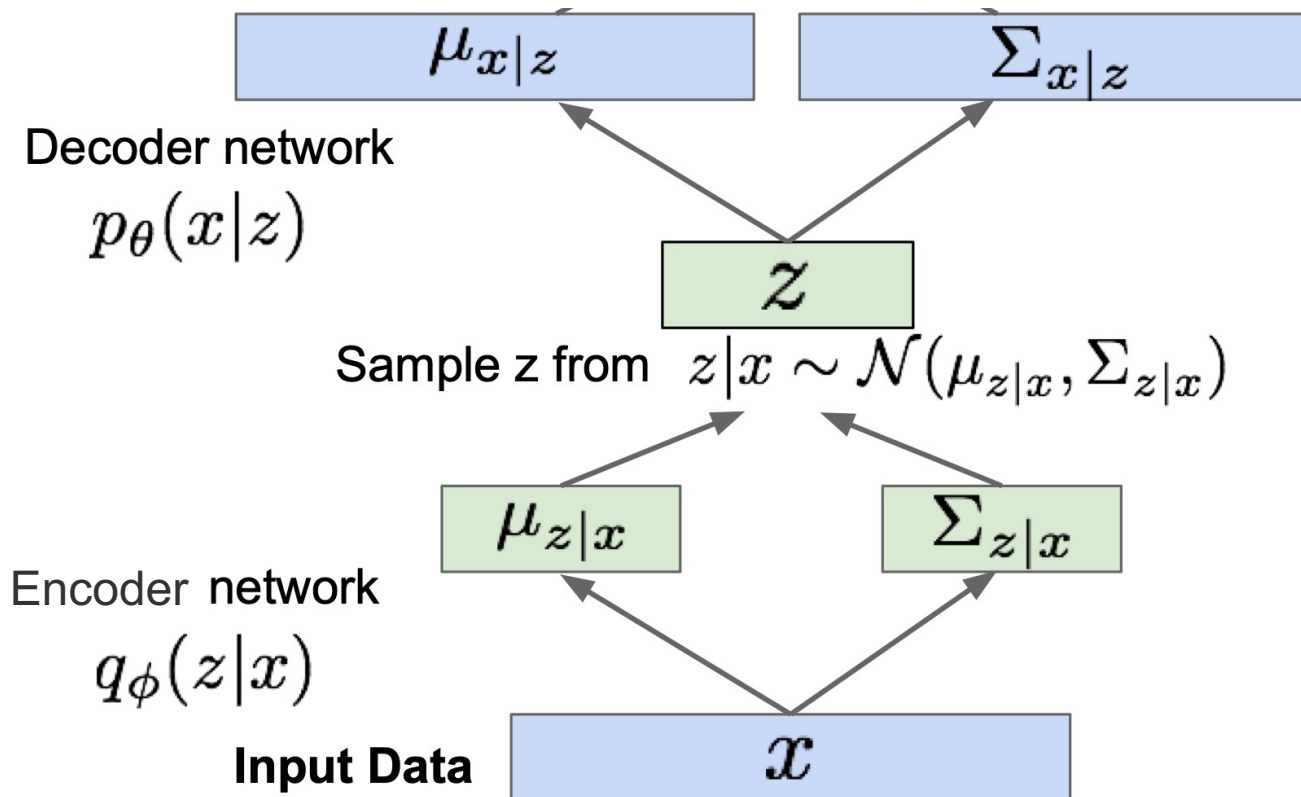
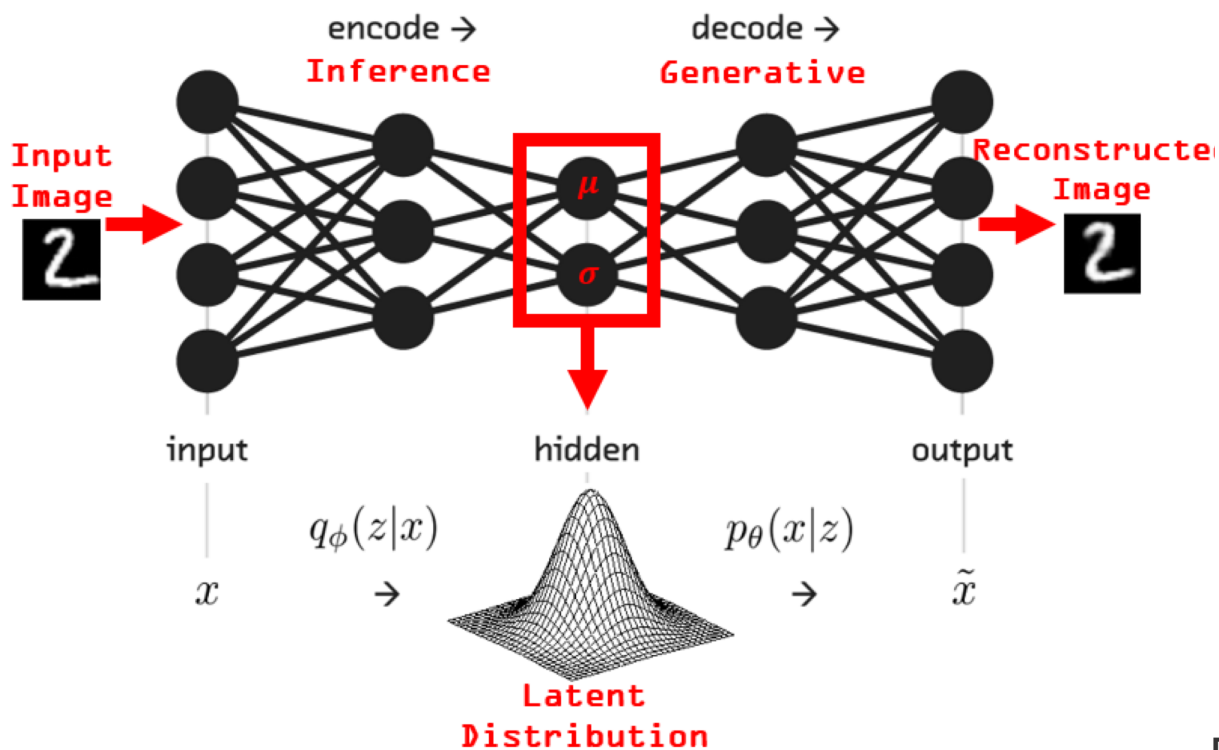
Example: VAEs for images



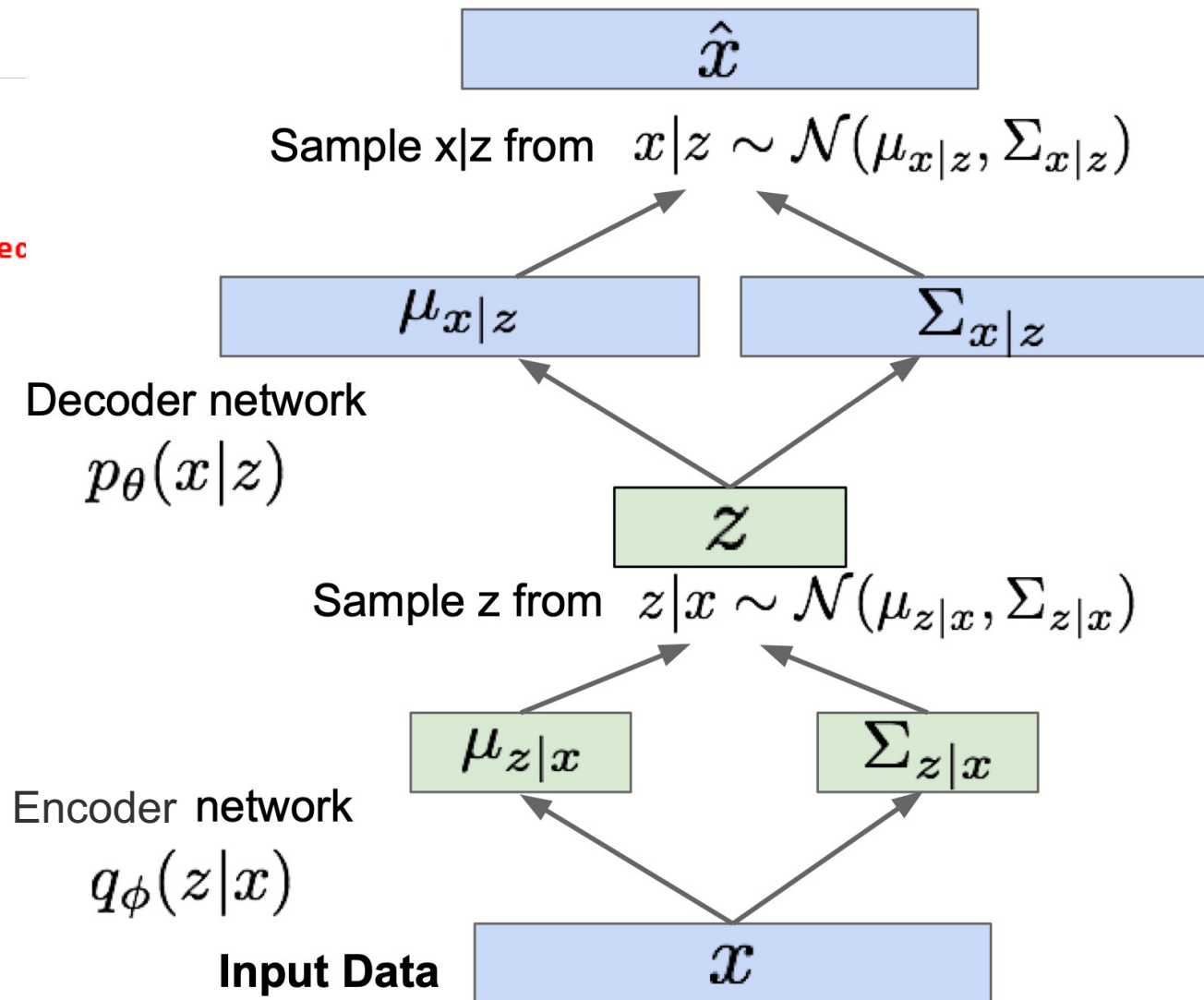
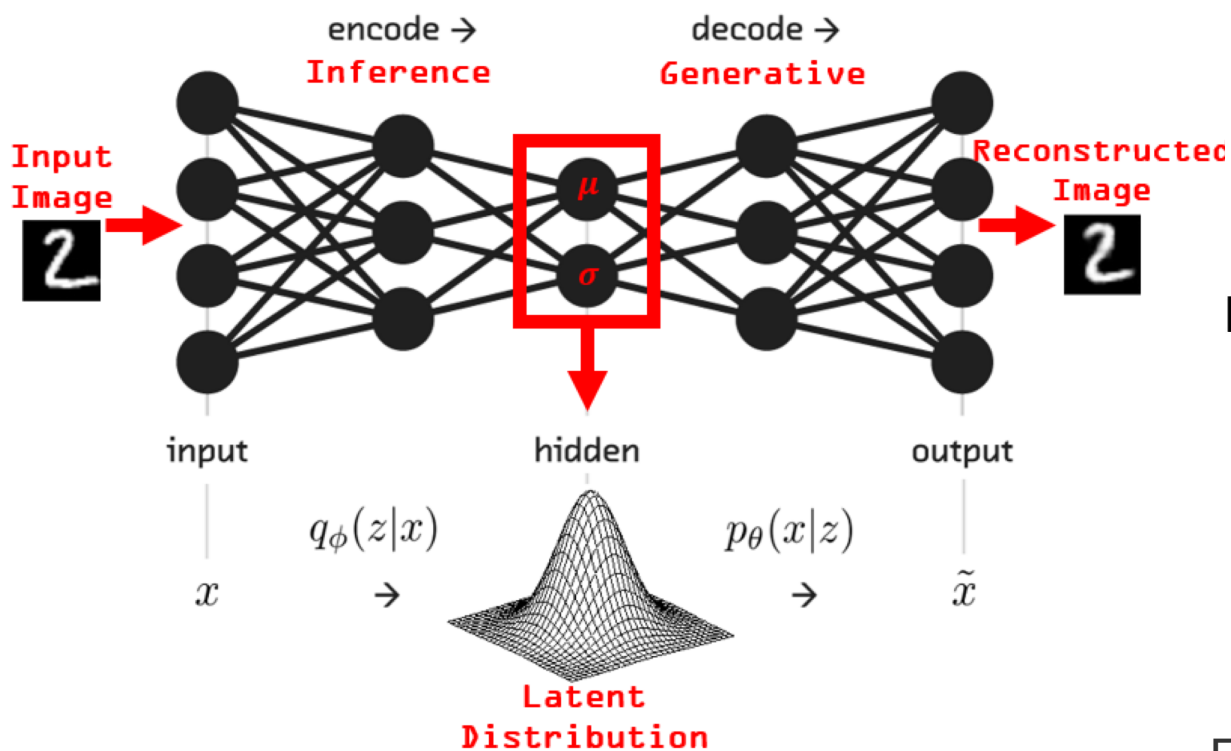
Example: VAEs for images



Example: VAEs for images



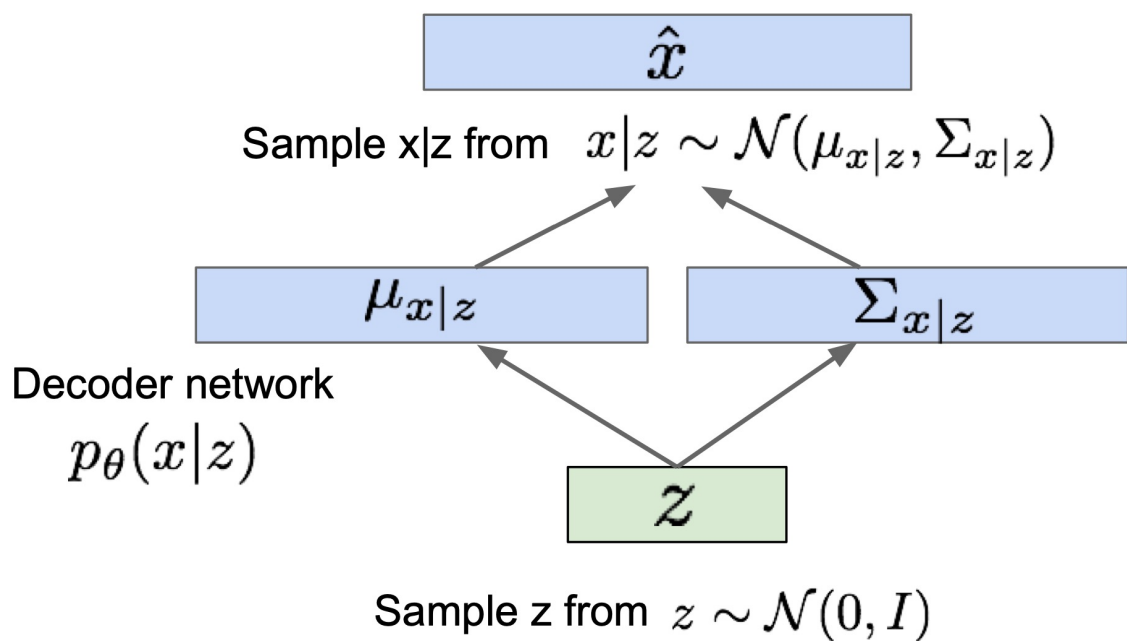
Example: VAEs for images



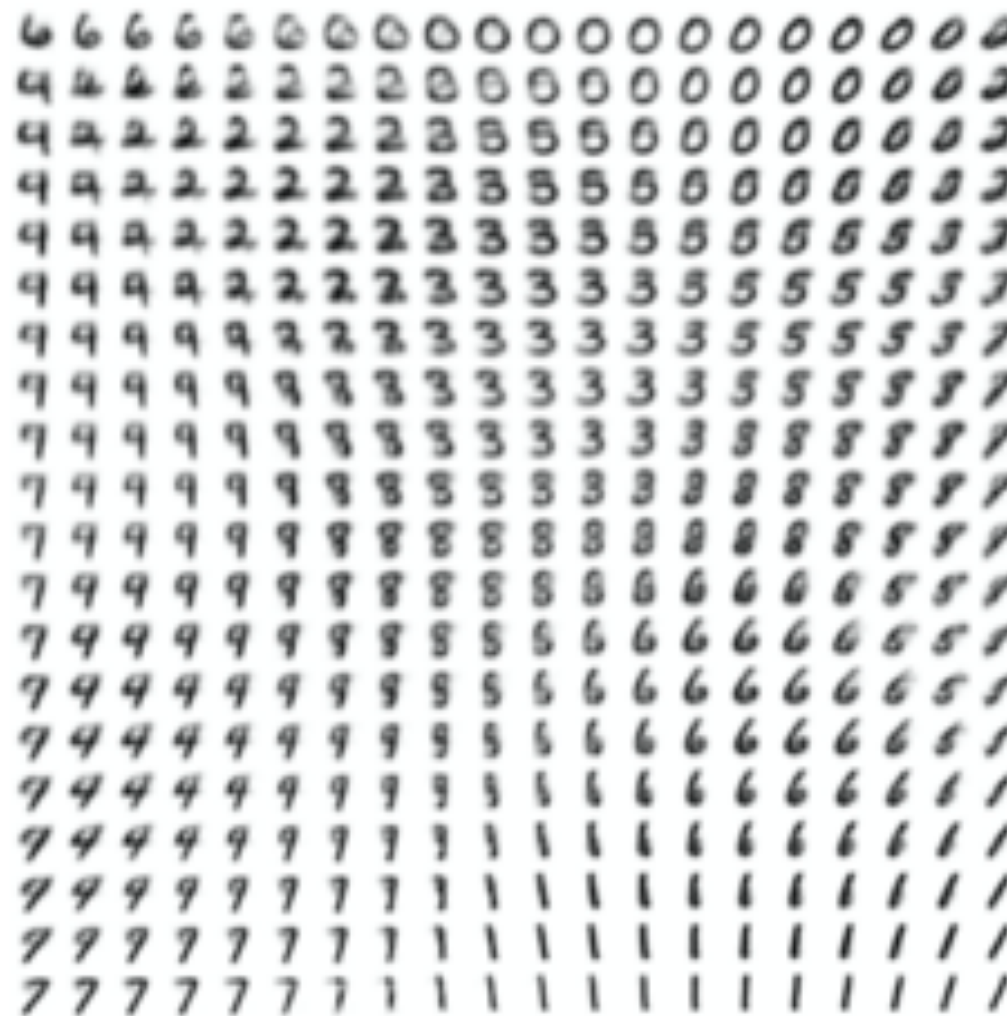
Example: VAEs for images

Generating samples:

- Use decoder network. Now sample z from prior!



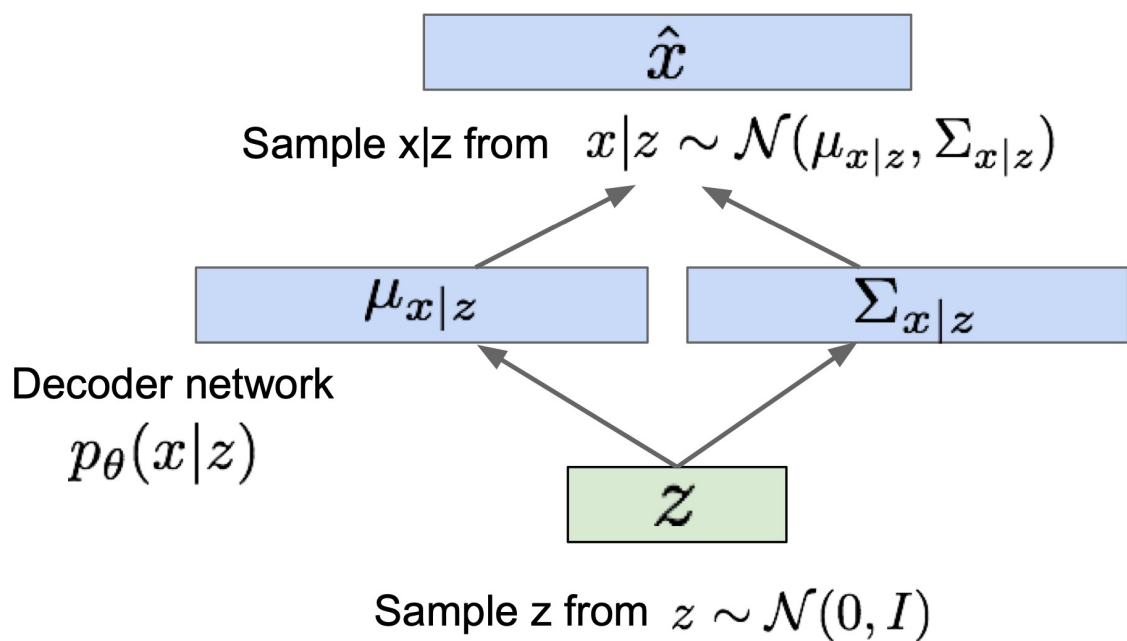
Data manifold for 2-d z



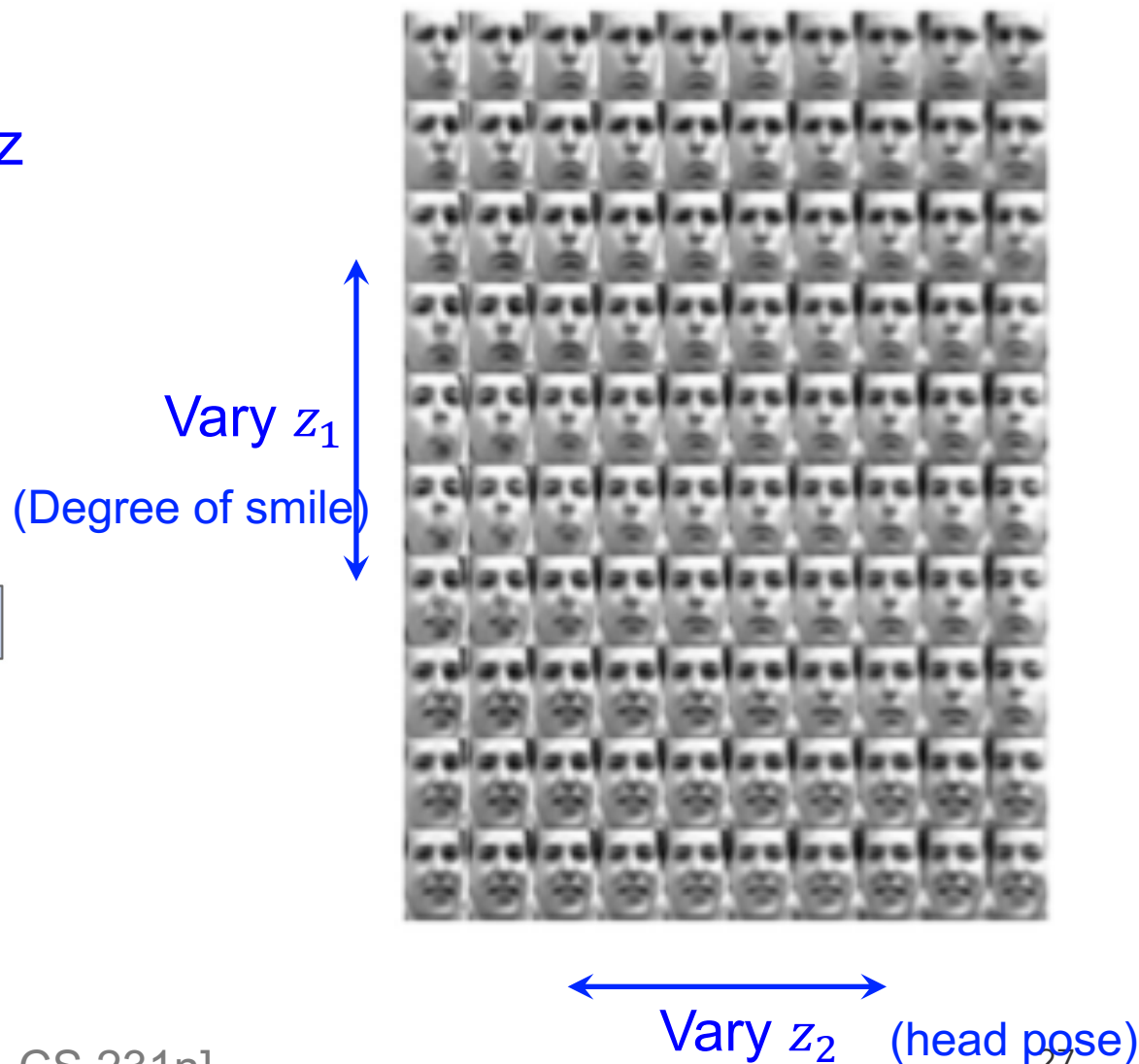
Example: VAEs for images

Generating samples:

- Use decoder network. Now sample z from prior!



Data manifold for 2-d z



Variational Auto-Encoders (VAEs)

Algorithm 1 Minibatch version of the Auto-Encoding VB (AEVB) algorithm. Either of the two SGVB estimators in section 2.3 can be used. We use settings $M = 100$ and $L = 1$ in experiments.

$\theta, \phi \leftarrow$ Initialize parameters

repeat

$\mathbf{X}^M \leftarrow$ Random minibatch of M datapoints (drawn from full dataset)

$\epsilon \leftarrow$ Random samples from noise distribution $p(\epsilon)$

$\mathbf{g} \leftarrow \nabla_{\theta, \phi} \tilde{\mathcal{L}}^M(\theta, \phi; \mathbf{X}^M, \epsilon)$ (Gradients of minibatch estimator (8))

$\theta, \phi \leftarrow$ Update parameters using gradients \mathbf{g} (e.g. SGD or Adagrad [DHS10])

until convergence of parameters (θ, ϕ)

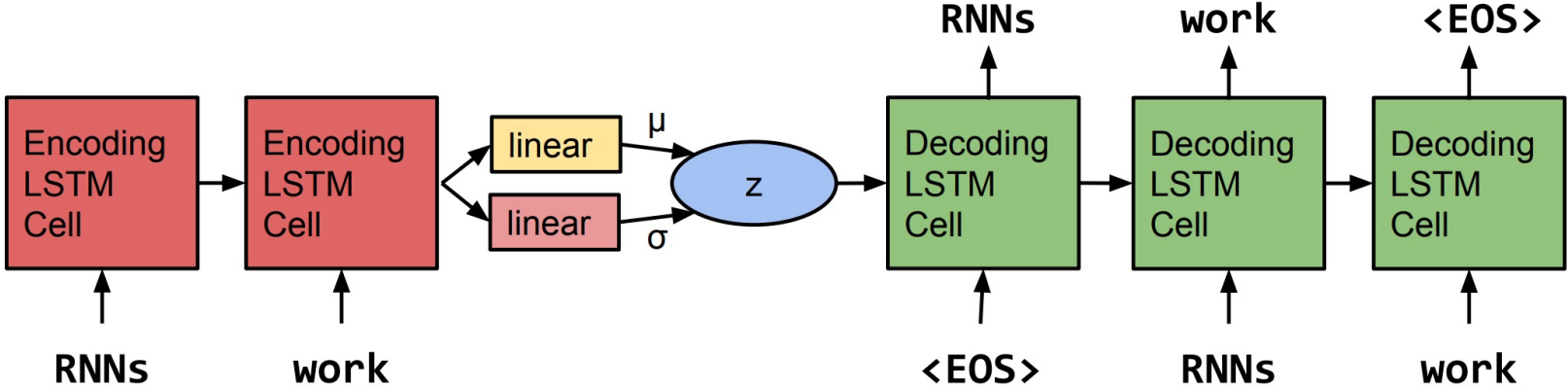
return θ, ϕ

[Kingma & Welling, 2014]

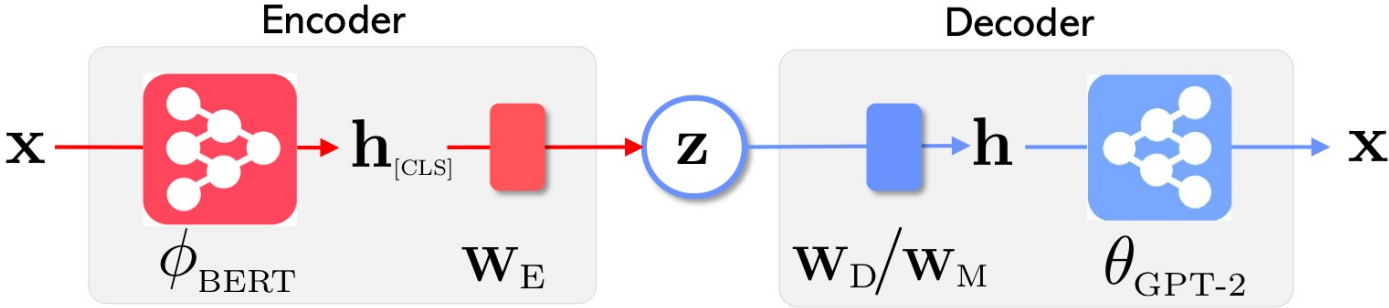
Note: Amortized Variational Inference

- Variational distribution as an **inference model** $q_{\phi}(\mathbf{z}|\mathbf{x})$ with parameters ϕ (which was traditionally factored over samples)
- Amortize the cost of inference by learning a **single** data-dependent inference model
- The trained inference model can be used for quick inference on new data

Example: VAEs for text



[Bowman et al., 2015: Generating sentences from a continuous space]



[Li et al., 2020: OPTIMUS: Organizing Sentences via Pre-trained Modeling of a Latent Space]

Example: VAEs for text

Latent code interpolation and sentences generation from VAEs

[Bowman et al., 2015]

“ i want to talk to you . ”

“i want to be with you . ”

“i do n’t want to be with you . ”

i do n’t want to be with you .

she did n’t want to be with him .

[Li et al., 2020]

0.0 children are looking for the water to be clear.
0.1 children are looking for the water.
0.2 children are looking at the water.
0.3 the children are looking at a large group of people.
0.4 the children are watching a group of people.
0.5 the people are watching a group of ducks.
0.6 the people are playing soccer in the field.
0.7 there are people playing a sport.
0.8 there are people playing a soccer game.
0.9 there are two people playing soccer.
1.0 there are two people playing soccer.

Example: VAEs for text

Source x_A a girl makes a silly face	Target x_B two soccer players are playing soccer
Input x_C <ul style="list-style-type: none">• a girl poses for a picture• a girl in a blue shirt is taking pictures of a microscope• a woman with a red scarf looks at the stars• a boy is taking a bath• a little boy is eating a bowl of soup	Output x_D <ul style="list-style-type: none">• two soccer players are at a soccer game.• two football players in blue uniforms are at a field hockey game• two men in white uniforms are field hockey players• two baseball players are at the baseball diamond• two men are in baseball practice

Table 2: Sentence transfer via arithmetic $z_D = z_B - z_A + z_C$. The output sentences are in blue.

Variational Auto-encoders: Summary

- A combination of the following ideas:
 - Variational Inference: ELBO
 - Variational distribution parametrized as neural networks
 - Reparameterization trick

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) = [\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] - \text{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) || p(\boldsymbol{z}))$$

← Reconstruction

↓ Divergence from prior

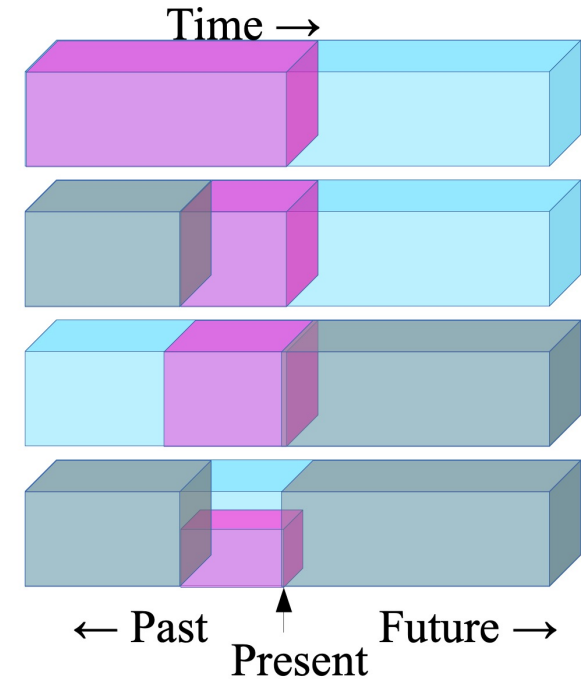


(Razavi et al., 2019)

- Pros:
 - Principled approach to generative models
 - Allows inference of $q(\boldsymbol{z}|\boldsymbol{x})$, can be useful feature representation for other tasks
- Cons:
 - Samples blurrier and lower quality compared to GANs
 - Tend to collapse on text data

Summary: Representation Learning

- Text x \rightarrow Representation z
- Self-supervised learning:
 - Language modeling: next-word prediction
 - GPT2, GPT3
 - Word embedding:
 - skip-gram: predicting context tokens
 - Contextualized embedding:
 - BERT: masked language model (MLM)
 - Contrastive learning: positive/negative samples, similarity measure / loss
- Unsupervised learning
 - EM algorithm
 - Variational inference
 - Variational autoencoders (VAEs)



Classification

Classification in NLP

- We approach many problems in NLP by treating them as problems of classification.
 - Input might be a document, a paragraph, a sentence, a word
 - Output is a label from a finite set of classes or labels, defined by your application or theory

Text (Document) Classification Examples

- ▶ Library-like subjects (e.g., the Dewey decimal system)
- ▶ News stories: politics vs. sports vs. business vs. technology ...
- ▶ Reviews of films, restaurants, products: positive vs. negative
- ▶ Author attributes: identity, political stance, gender, age, ...
- ▶ Email, arXiv submissions, etc.: spam vs. not
- ▶ What is the reading level of a piece of text?
- ▶ How influential will a scientific paper be?
- ▶ Will a piece of proposed legislation pass?
- ▶ What dialect is a text written in?
- ▶ Does the text contain content that will likely offend people?

Features of a Text

Running example:

$x =$ "The vodka was great, but don't touch the hamburgers."

A different representation of the text sequences: features.



"Bag of words" features

Features of a Text

Running example:

$x =$ "The vodka was great, but don't touch the hamburgers."

A different representation of the text sequences: features.

- ▶ Often, these are term (word or word sequence) frequencies.

E.g., $\phi_{\text{hamburgers}}^{\text{freq.}}(x) = 1$, $\phi_{\text{the}}^{\text{freq.}}(x) = 2$, $\phi_{\text{delicious}}^{\text{freq.}}(x) = 0$,

$\phi_{\text{don't touch}}^{\text{freq.}}(x) = 1$.



"Bag of words" features

Features of a Text

Running example:

$x =$ “The vodka was great, but don’t touch the hamburgers.”

A different representation of the text sequences: features.

- ▶ Often, these are term (word or word sequence) frequencies.

E.g., $\phi_{\text{hamburgers}}^{\text{freq.}}(x) = 1$, $\phi_{\text{the}}^{\text{freq.}}(x) = 2$, $\phi_{\text{delicious}}^{\text{freq.}}(x) = 0$,

$\phi_{\text{don't touch}}^{\text{freq.}}(x) = 1$.

- ▶ Can also be binary word “presence” features.

E.g., $\phi_{\text{hamburgers}}^{\text{presence}}(x) = 1$, $\phi_{\text{the}}^{\text{presence}}(x) = 1$, $\phi_{\text{delicious}}^{\text{presence}}(x) = 0$,

$\phi_{\text{don't touch}}^{\text{presence}}(x) = 1$.



“Bag of words” features

Features of a Text

Running example:

$x = \text{"The vodka was great, but don't touch the hamburgers."}$

A different representation of the text sequences: features.

- ▶ Often, these are term (word or word sequence) frequencies.
E.g., $\phi_{\text{hamburgers}}^{\text{freq.}}(\mathbf{x}) = 1$, $\phi_{\text{the}}^{\text{freq.}}(\mathbf{x}) = 2$, $\phi_{\text{delicious}}^{\text{freq.}}(\mathbf{x}) = 0$,
 $\phi_{\text{don't touch}}^{\text{freq.}}(\mathbf{x}) = 1$.
- ▶ Can also be binary word "presence" features.
E.g., $\phi_{\text{hamburgers}}^{\text{presence}}(\mathbf{x}) = 1$, $\phi_{\text{the}}^{\text{presence}}(\mathbf{x}) = 1$, $\phi_{\text{delicious}}^{\text{presence}}(\mathbf{x}) = 0$,
 $\phi_{\text{don't touch}}^{\text{presence}}(\mathbf{x}) = 1$.
- ▶ Transformations on word frequencies: logarithm, idf weighting

$$\forall v \in \mathcal{V}, \text{idf}(v) = \log \frac{n}{|i : \text{count}_{\mathbf{x}_i}(v) > 0|}$$
$$\phi_v^{\text{tfidf}}(\mathbf{x}) = \phi_v^{\text{freq.}}(\mathbf{x}) \cdot \text{idf}(v)$$



"Bag of words" features

idf weight

- $df(v)$ is the document frequency of word v : the number of documents that contain v
 - $df(v)$ is an inverse measure of the informativeness of v
 - $df(v) \leq N$
- We define the idf (inverse document frequency) of v by

$$\forall v \in \mathcal{V}, \text{idf}(v) = \log \frac{n}{|i : \text{count}_{\mathbf{x}_i}(v) > 0|}$$

$$\phi_v^{tfidf}(\mathbf{x}) = \phi_v^{freq.}(\mathbf{x}) \cdot \text{idf}(v)$$

...

Features of a Text

Running example:

$x = \text{"The vodka was great, but don't touch the hamburgers."}$

A different representation of the text sequences: features.

- ▶ Often, these are term (word or word sequence) frequencies.

E.g., $\phi_{\text{hamburgers}}^{\text{freq.}}(\mathbf{x}) = 1$, $\phi_{\text{the}}^{\text{freq.}}(\mathbf{x}) = 2$, $\phi_{\text{delicious}}^{\text{freq.}}(\mathbf{x}) = 0$,
 $\phi_{\text{don't touch}}^{\text{freq.}}(\mathbf{x}) = 1$.

- ▶ Can also be binary word “presence” features.

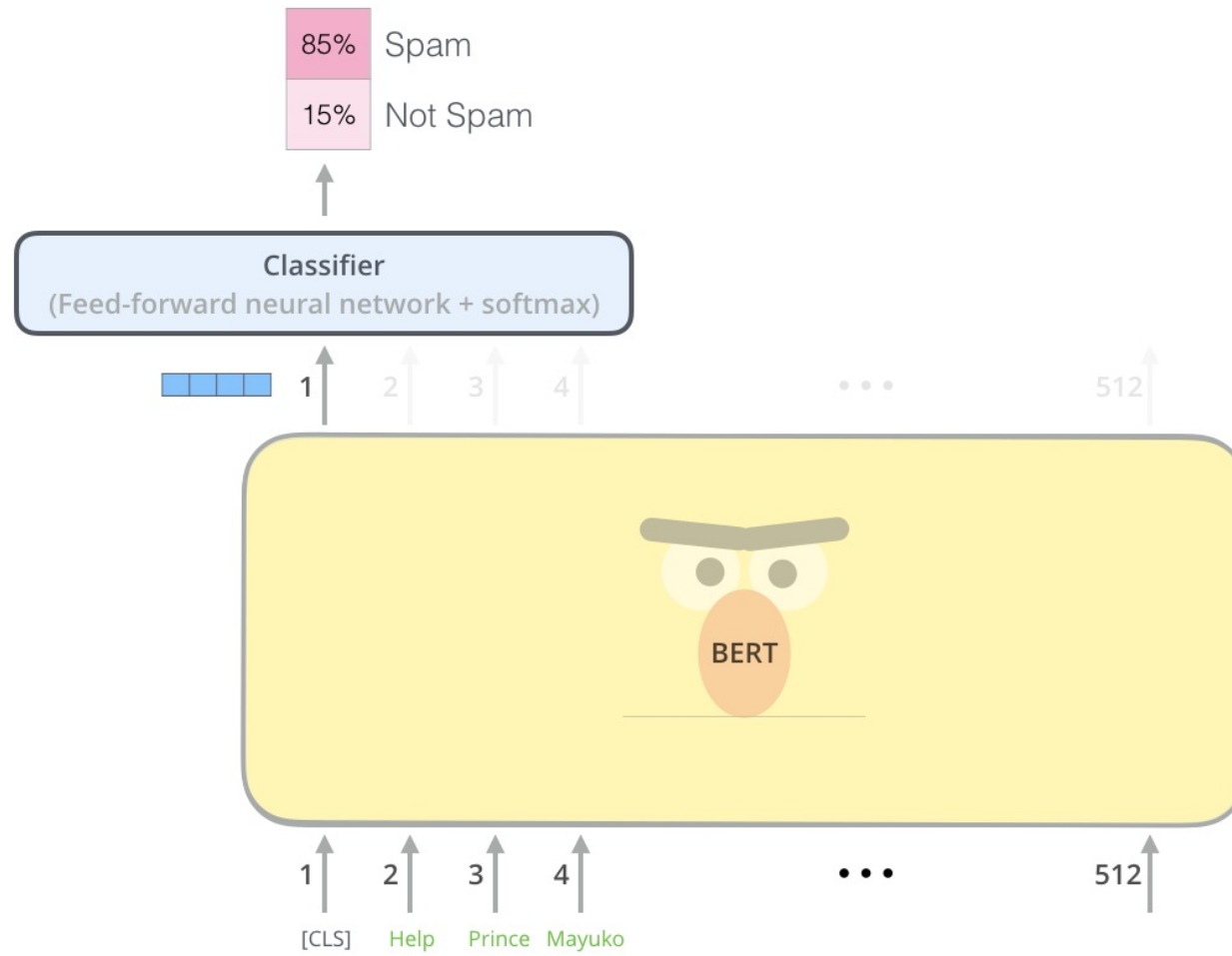
E.g., $\phi_{\text{hamburgers}}^{\text{presence}}(\mathbf{x}) = 1$, $\phi_{\text{the}}^{\text{presence}}(\mathbf{x}) = 1$, $\phi_{\text{delicious}}^{\text{presence}}(\mathbf{x}) = 0$,
 $\phi_{\text{don't touch}}^{\text{presence}}(\mathbf{x}) = 1$.

- ▶ Transformations on word frequencies: logarithm, idf weighting

$$\forall v \in \mathcal{V}, \text{idf}(v) = \log \frac{n}{|i : \text{count}_{\mathbf{x}_i}(v) > 0|}$$
$$\phi_v^{\text{tfidf}}(\mathbf{x}) = \phi_v^{\text{freq.}}(\mathbf{x}) \cdot \text{idf}(v)$$

- ▶ Neural features automatically learned from data

Classification using neural features



Use BERT for sentence classification

Classification with few labels

- Data augmentation
- Zero-/few-shot learning via prompting

Text data augmentation

- Applies label-preserving transformations on original data points to expand the data size

Methods	Level	Diversity	Tasks	Related Work
Synonym replacement	Token	Low	Text classification Sequence labeling	Kolomiyets et al. (2011), Zhang et al. (2015a), Yang (2015), Miao et al. (2020), Wei and Zou (2019)
Word replacement via LM	Token	Medium	Text classification Sequence labeling Machine translation	Kolomiyets et al. (2011), Gao et al. (2019) Kobayashi (2018), Wu et al. (2019a) Fadaee et al. (2017)
Random insertion, deletion, swapping	Token	Low	Text classification Sequence labeling Machine translation Dialogue generation	Iyyer et al. (2015), Xie et al. (2017) Artetxe et al. (2018), Lample et al. (2018) Xie et al. (2020), Wei and Zou (2019)
Compositional Augmentation	Token	High	Semantic Parsing Sequence labeling Language modeling Text generation	Jia and Liang (2016) , Andreas (2020) Nye et al. (2020), Feng et al. (2020) Furrer et al. (2020) , Guo et al. (2020)
Paraphrasing	Sentence	High	Text classification Machine translation Question answering Dialogue generation Text summarization	Yu et al. (2018), Xie et al. (2020) Chen et al. (2019), He et al. (2020) Chen et al. (2020c), Cai et al. (2020)
Conditional generation	Sentence	High	Text classification Question answering	Anaby-Tavor et al. (2020), Kumar et al. (2020) Zhang and Bansal (2019), Yang et al. (2020)

Text data augmentation

- Applies label-preserving transformations on original data points to expand the data size

White-box attack	Token or Sentence	Medium	Text classification Sequence labeling Machine translation	Miyato et al. (2017), Ebrahimi et al. (2018b) Ebrahimi et al. (2018a), Cheng et al. (2019), Chen et al. (2020d)
Black-box attack	Token or Sentence	Medium	Text classification Sequence labeling Machine translation Textual entailment Dialogue generation Text Summarization	Jia and Liang (2017) Belinkov and Bisk (2017), Zhao et al. (2017) Ribeiro et al. (2018), McCoy et al. (2019) Min et al. (2020), Tan et al. (2020)
Hidden-space perturbation	Token or Sentence	High	Text classification Sequence labeling Speech recognition	Hsu et al. (2017), Hsu et al. (2018) Wu et al. (2019b), Chen et al. (2021) Malandrakis et al. (2019), Shen et al. (2020)
Interpolation	Token	High	Text classification Sequence labeling Machine translation	Miao et al. (2020), Chen et al. (2020c) Cheng et al. (2020b), Chen et al. (2020a) Guo et al. (2020)

Text data augmentation: Examples

- Lexical Substitution
 - Thesaurus-based substitution



Text data augmentation: Examples

- Lexical Substitution
 - Thesaurus-based substitution
 - Word-embedding substitution

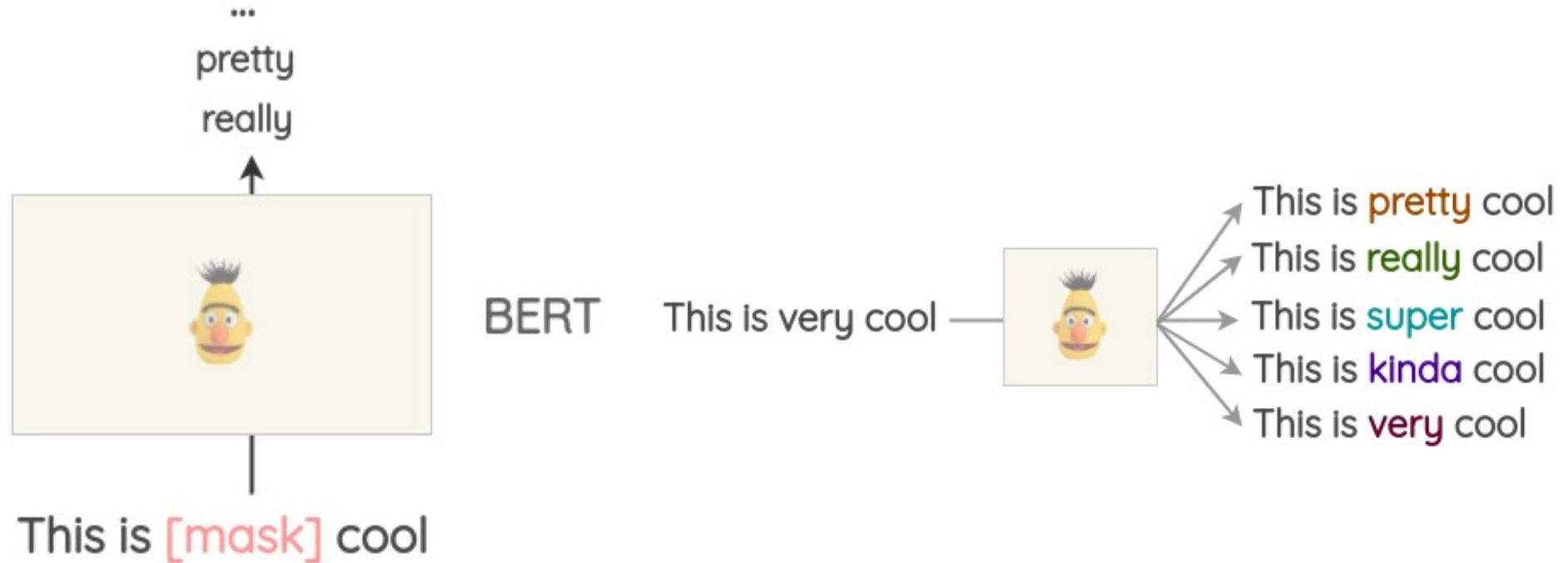
Nearest neighbors in word2vec



It is awesome → It is amazing
→ It is perfect
→ It is fantastic

Text data augmentation: Examples

- Lexical Substitution
 - Thesaurus-based substitution
 - Word-embedding substitution
 - Masked LM



Text data augmentation: Examples

- Lexical Substitution
 - Thesaurus-based substitution
 - Word-embedding substitution
 - Masked LM
 - TF-IDF based word replacement
 - words that have low TF-IDF scores are uninformative and thus can be replaced without affecting the ground-truth labels of the sentence.

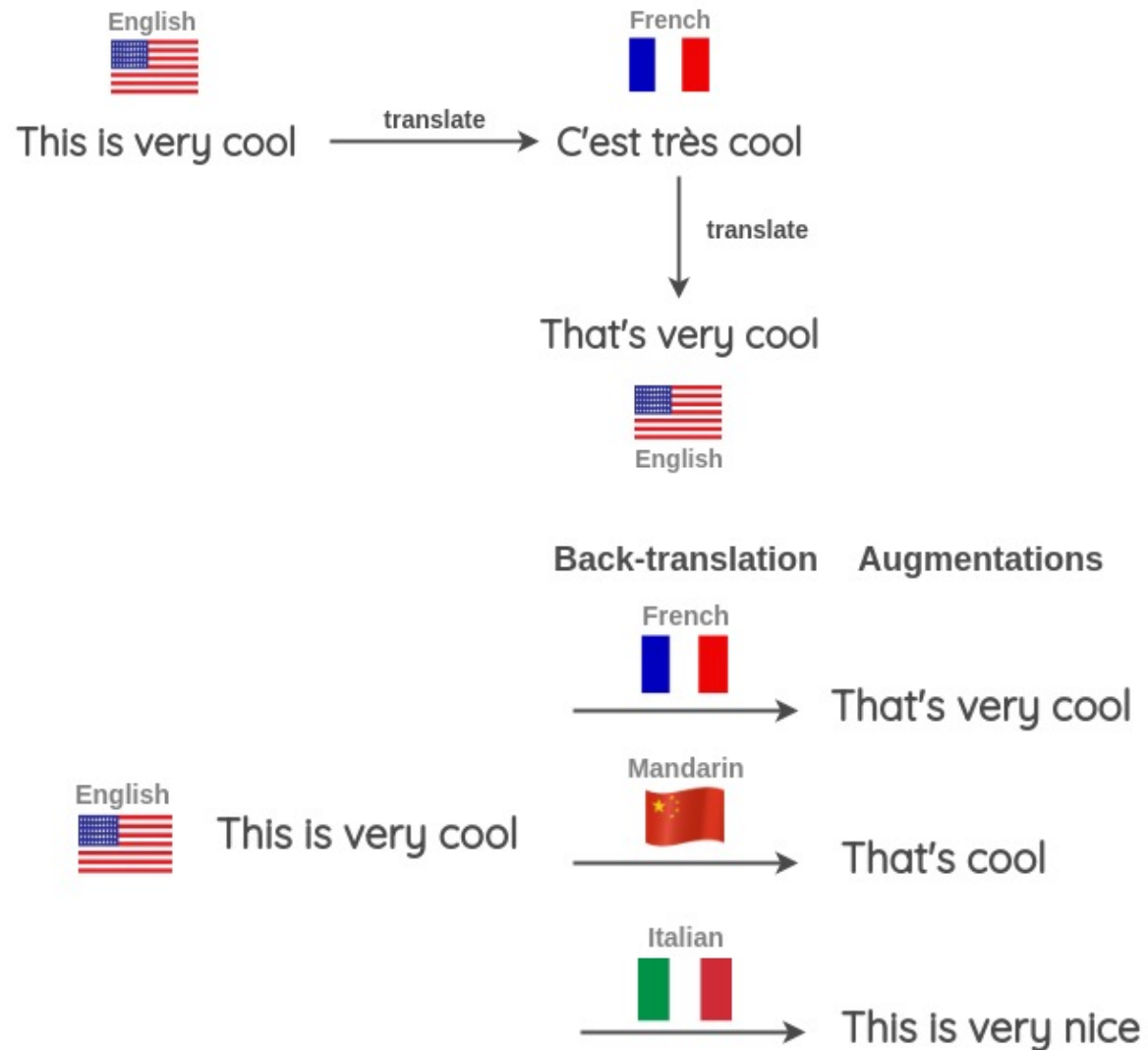
This virus has spread worldwide



A virus has spread worldwide

Text data augmentation: Examples

- Lexical Substitution
 - Thesaurus-based substitution
 - Word-embedding substitution
 - Masked LM
 - TF-IDF based word replacement
- Paraphrasing
 - Back Translation



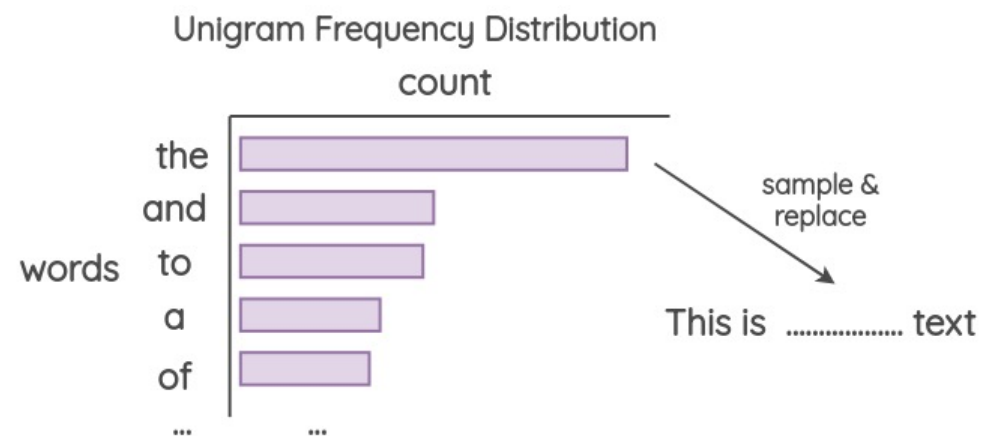
Text data augmentation: Examples

- Lexical Substitution
 - Thesaurus-based substitution
 - Word-embedding substitution
 - Masked LM
 - TF-IDF based word replacement
- Paraphrasing
 - Back Translation
- Random Noise Injection

Spelling error:

This is very cool → Thes is very cool
This is very cool → This id very cool

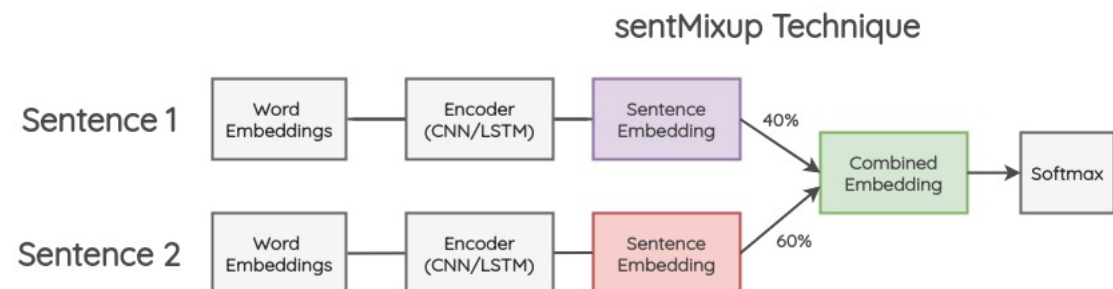
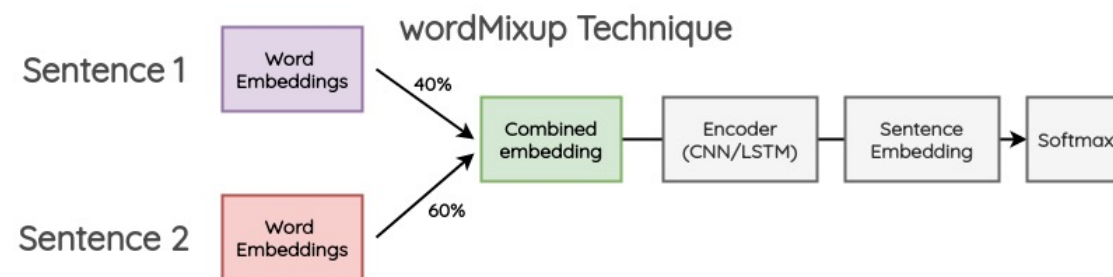
Unigram noising:



Text data augmentation: Examples

- Lexical Substitution
 - Thesaurus-based substitution
 - Word-embedding substitution
 - Masked LM
 - TF-IDF based word replacement
- Paraphrasing
 - Back Translation
- Random Noise Injection
- MixUp

Original Mixup algorithm



Text data augmentation: Examples

- Lexical Substitution
 - Thesaurus-based substitution
 - Word-embedding substitution
 - Masked LM
 - TF-IDF based word replacement
- Paraphrasing
 - Back Translation
- Random Noise Injection
- MixUp
- Generative Models
 - Finetune a large pre-trained LM (BERT, GPT2, etc)
 - Use the fine-tuned LM to generate new data

Finetune on training data

GPT2

Task: Learn to generate training data

Output: POSITIVE<SEP>It is very useful app<EOS>

Generate new samples

GPT2

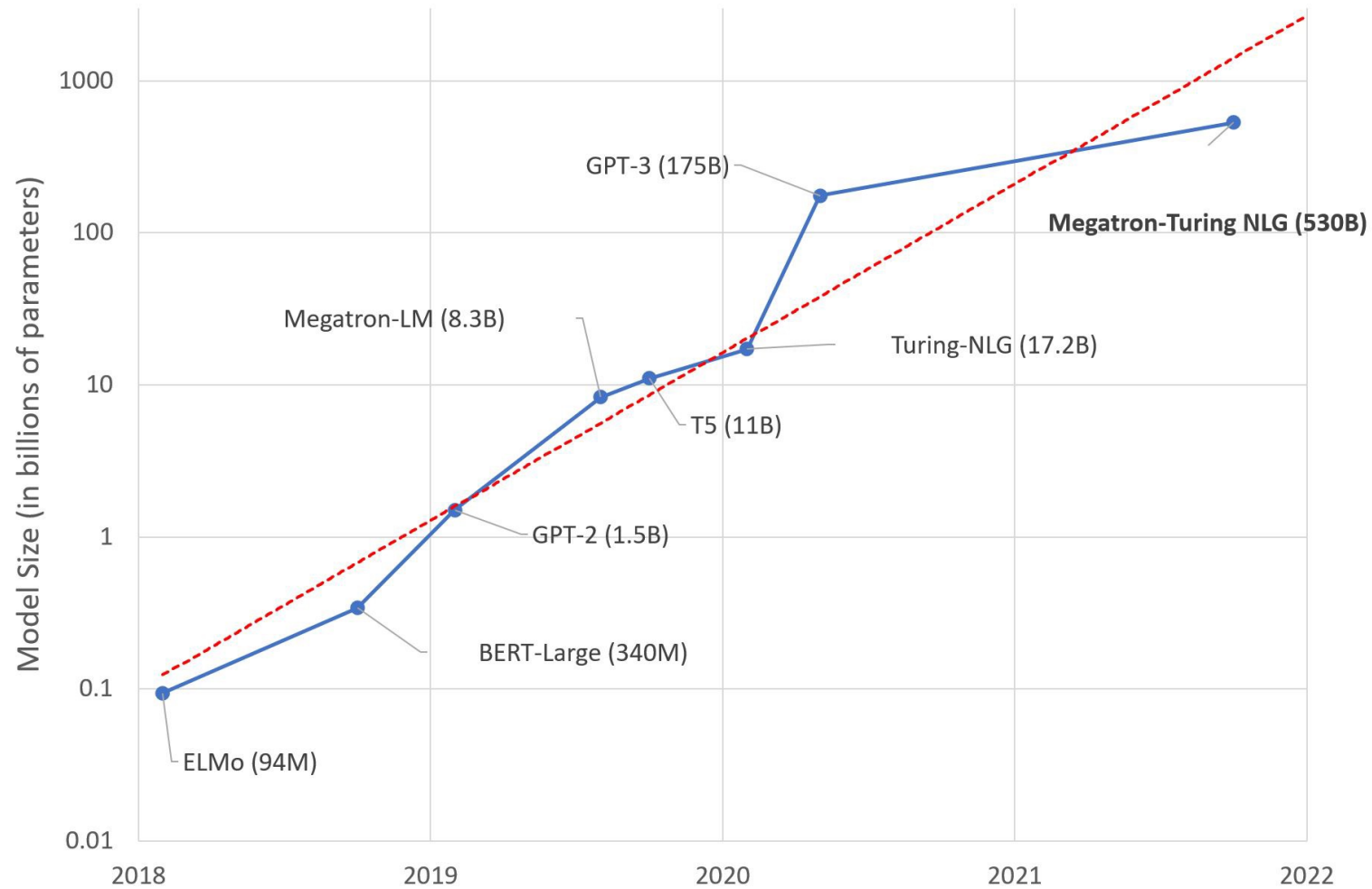
Prompt: POSITIVE <SEP>It is very

Generate: POSITIVE <SEP> It is very helpful tool<EOS>

Classification with few labels

- Data augmentation
- Zero-/few-shot learning via prompting

Increasing LM size



Traditional fine-tuning

Downstream training data

The model is trained via repeated gradient updates using a large corpus of example tasks.



Downstream test data

Zero-shot


The model predicts the answer given only a natural language description of the task. No gradient updates are performed.



No fine-tuning! Literally just take a pretrained LM and give it the prefix

One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

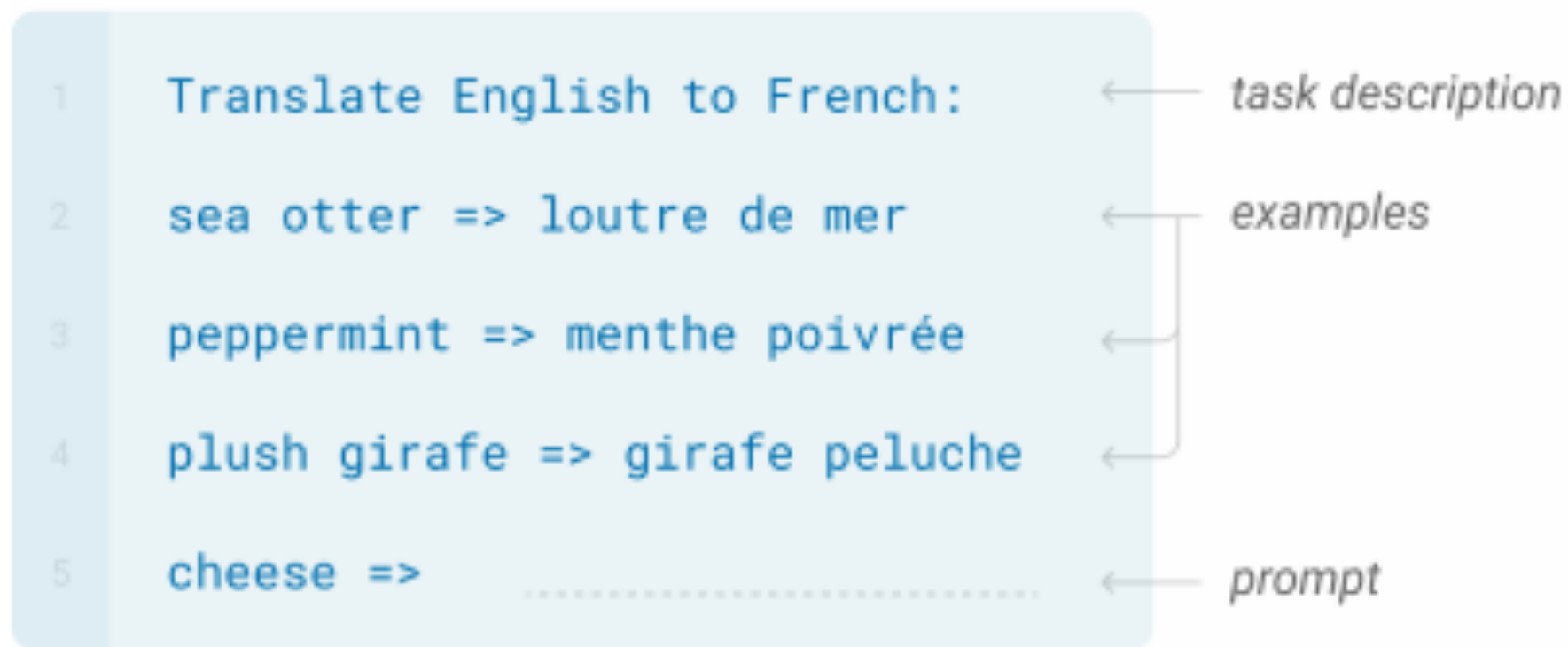


The diagram shows a light blue rounded rectangle containing three lines of text. To the right of the rectangle, three arrows point to the corresponding lines, each with a label. The first line is '1 Translate English to French:' with an arrow pointing to it labeled 'task description'. The second line is '2 sea otter => loutre de mer' with an arrow pointing to it labeled 'example'. The third line is '3 cheese =>' with an arrow pointing to it labeled 'prompt'. Below the third line, there is a horizontal dotted line.

```
1 Translate English to French: ← task description
2 sea otter => loutre de mer ← example
3 cheese => ← prompt
.....
```

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.



Performance (TriviaQA)

Question

Miami Beach in Florida borders which ocean?

What was the occupation of Lovely Rita according to the song by the Beatles

Who was Poopdeck Pappys most famous son?

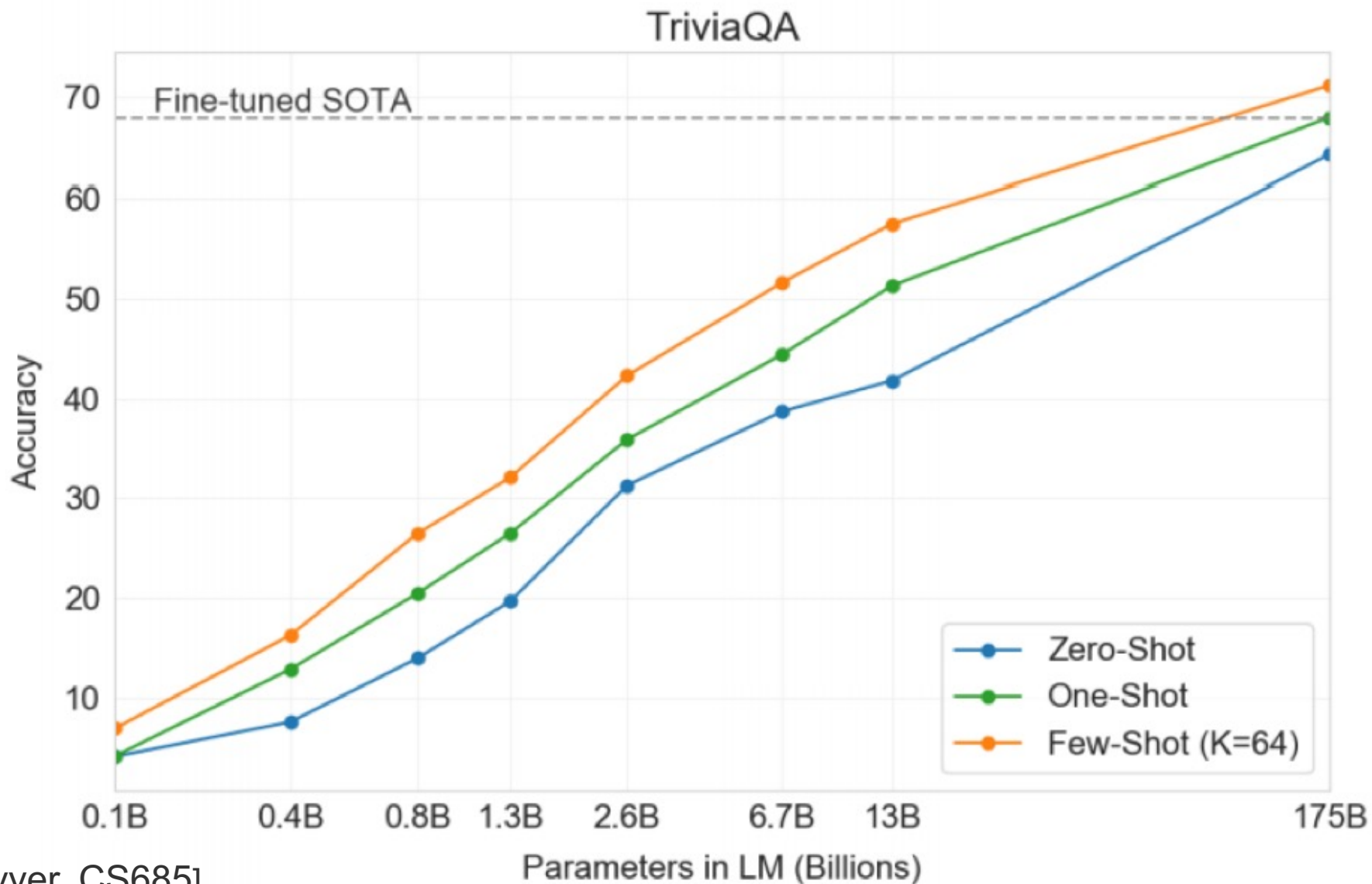
The Nazi regime was Germany's Third Reich; which was the first Reich?

At which English racecourse did two horses collapse and die in the parade ring due to electrocution, in February 2011?

Which type of hat takes its name from an 1894 novel by George Du Maurier where the title character has the surname O'Ferrall ?

What was the Elephant Man's real name?

Performance (TriviaQA)



Sub-optimal and sensitive discrete/hard prompts

Discrete/hard prompts

- natural language instructions/task descriptions

Problems

- requiring domain expertise/understanding of the model's inner workings
- performance still lags far behind SotA model tuning results
- sub-optimal and sensitive
 - prompts that humans consider reasonable is not necessarily effective for language models ([Liu et al., 2021](#))
 - pre-trained language models are sensitive to the choice of prompts ([Zhao et al., 2021](#))

Sub-optimal and sensitive discrete/hard prompts

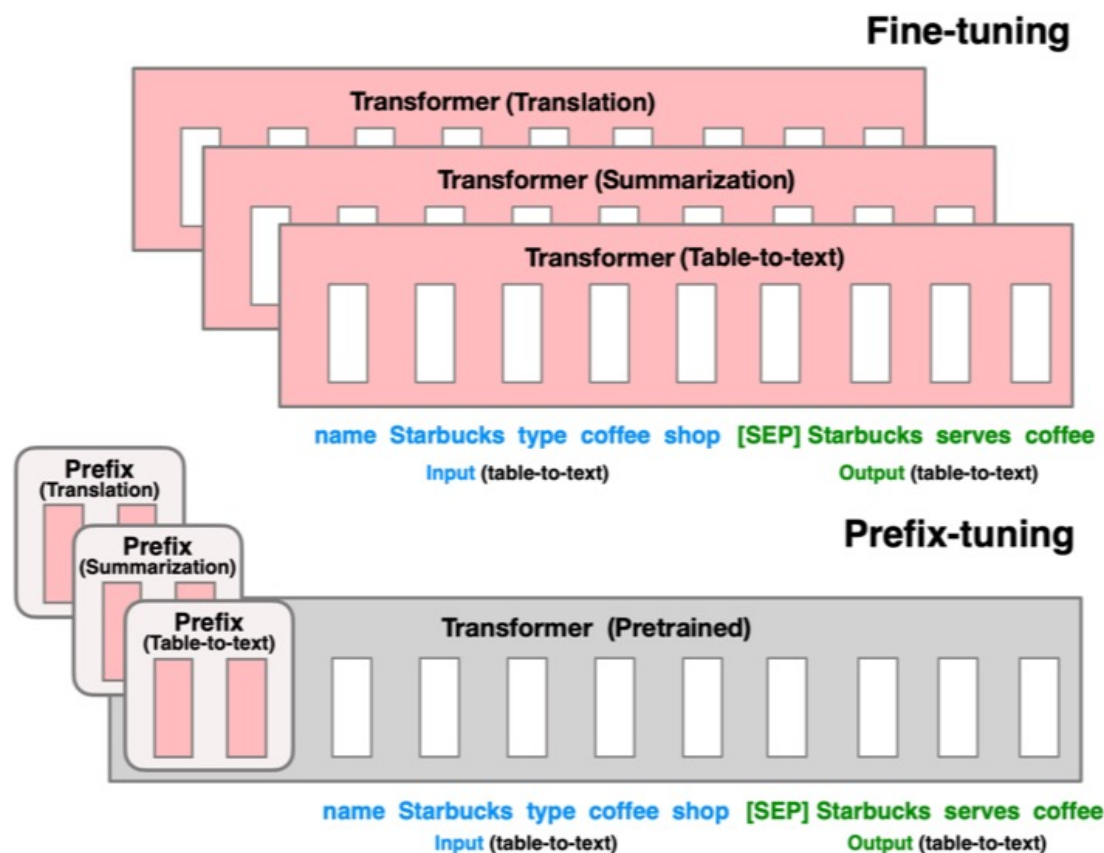
Prompt	P@1
[X] is located in [Y]. (<i>original</i>)	31.29
[X] is located in which country or state? [Y].	19.78
[X] is located in which country? [Y].	31.40
[X] is located in which country? In [Y].	51.08

Table 1. Case study on LAMA-TREx P17 with bert-base-cased. A single-word change in prompts could yield a drastic difference.

Liu et al., 2021

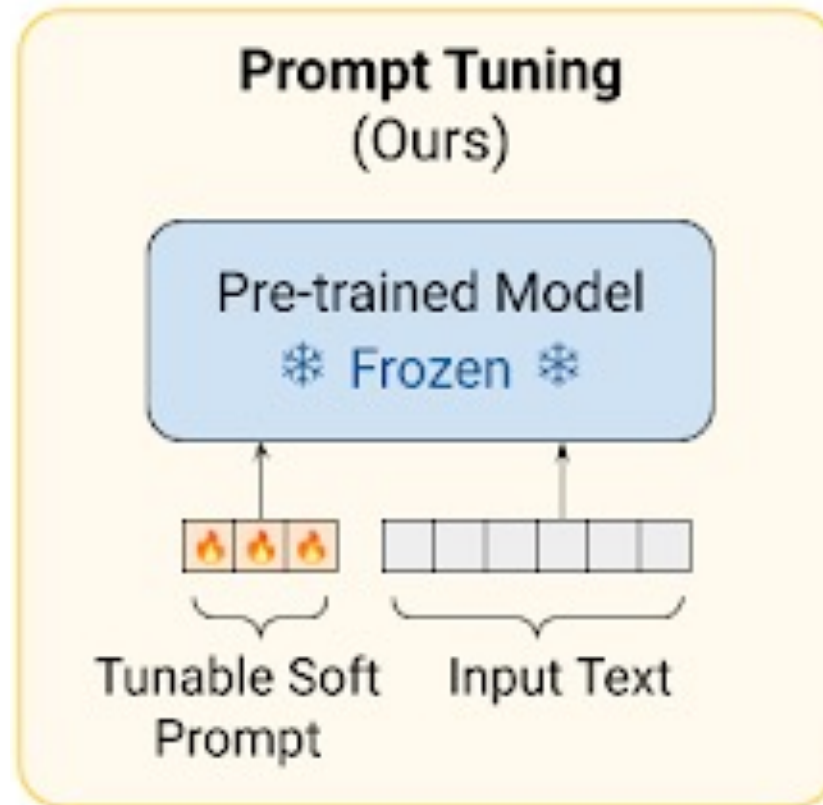
Continuous/soft prompts

- Learning continuous/soft prompts (Liu et al., 2021; Li and Liang., 2021; Qin and Eisner., 2021; Lester et al., 2021)
 - A sequence of additional task-specific tunable tokens prepended to the input text



Continuous/soft prompts

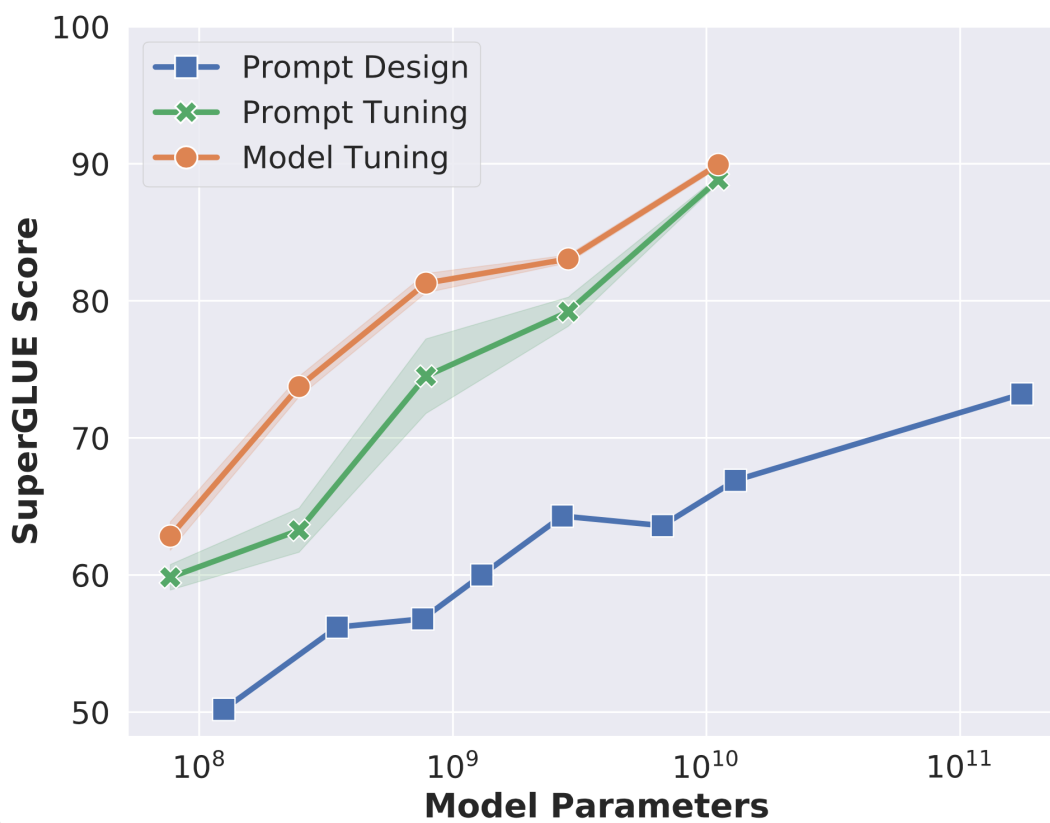
- Learning continuous/soft prompts (Liu et al., 2021; Li and Liang., 2021; Qin and Eisner., 2021; Lester et al., 2021)
 - A sequence of additional task-specific tunable tokens prepended to the input text



(Lester et al., 2021)

Continuous/soft prompts

- Learning continuous/soft prompts (Liu et al., 2021; Li and Liang., 2021; Qin and Eisner., 2021; Lester et al., 2021)
 - A sequence of additional task-specific tunable tokens prepended to the input text



Prompt Tuning becomes more competitive with scale

Key Takeaways

- Text features:
 - Bag-of-words, TF-IDF weighting, neural features
- Data augmentation
 - Lexical Substitution
 - Paraphrasing
 - Random Noise Injection
 - MixUp
 - Generative Models
- Prompting for few-shot learning

Questions?