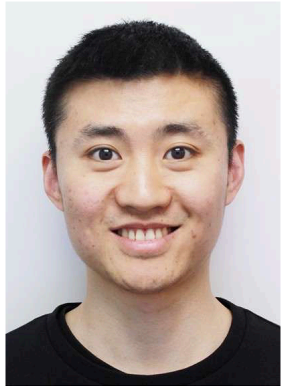
Discreteness in Neural Natural Language Processing

Lili Mou[a] Hao Zhou[b] Lei Li[b]

[a]Alberta Machine Intelligence Institute (Amii), University of Alberta
[b]ByteDance AI Lab
doublepower.mou@gmail.com
{zhouhao.nlp,lileilab}@bytedance.com

EMNLP-IJCNLP 2019 Tutorial



Part IV: Discrete Output Space

Roadmap

- Examples of discrete output space
- Challenges and Solutions of Discrete Output Space
  - From Continuous Outputs to Discrete Outputs
    - Embedding Matching by Softmax
  - Non-differentiable: Difficult for non-MLE training (e.g., GAN)
    - RL for Generation
    - Gumbel Softmax for Generation
  - Exponential Search Space
    - Hard for Global Inference
    - Hard for Constrained Decoding
- Case Study
  - Kernelized Bayesian Softmax
  - SeqGAN
  - Constrained Sentence Generation with CGMH

Outputs of NLP Tasks

$$X \xrightarrow[P(Y|X)]{\text{neural networks}} Y$$

Word
Sentence
Tree
Graph
...

Outputs of NLP Tasks

$$X \xrightarrow[P(Y|X)]{\text{neural networks}} Y \left\{ \begin{array}{c} \text{Word} \\ \text{Sentence} \\ \text{Tree} \\ \text{Graph} \\ \ldots \end{array} \right.$$

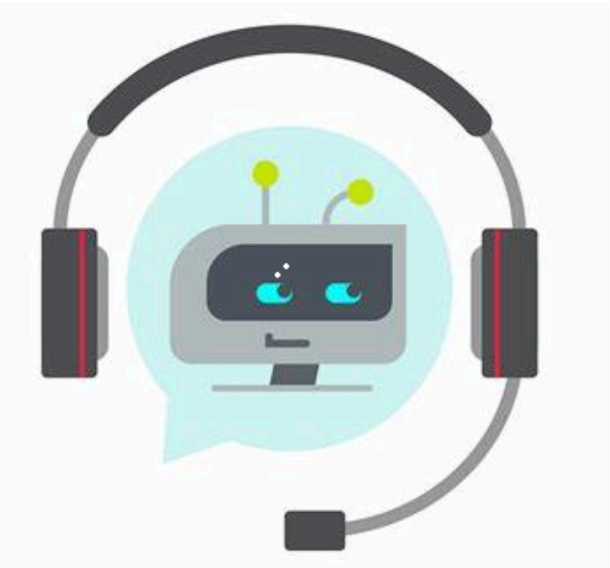
More complex discrete outputs such as sequence, tree or graph structures exit in NLP.

Output Sentences

Machine Writing

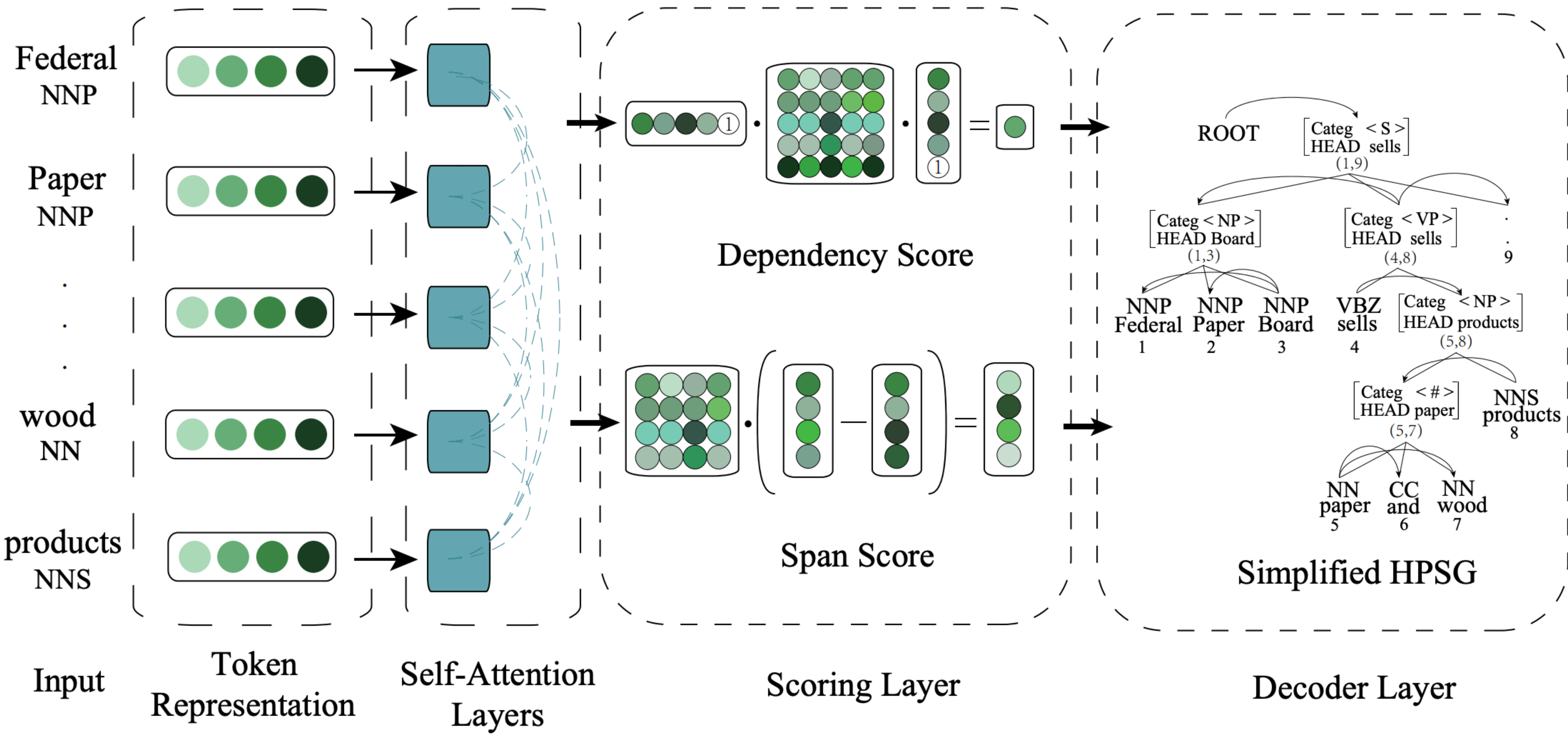
Question Answering

ChatBOT



Machine Translation

Output Trees



Zhou J, Zhao H. Head-driven phrase structure grammar parsing on Penn treebank, in ACL, 2019.

Roadmap

- Examples of discrete output space
- Challenges and Solutions of Discrete Output Space
  - From Continuous Outputs to Discrete Outputs
    - Embedding Matching by Softmax
  - Non-differentiable: Difficult for non-MLE training (e.g., GAN)
    - RL for Generation
    - Gumbel Softmax for Generation
  - Exponential Search Space
    - Hard for Global Inference
    - Hard for Constrained Decoding
- Case Study
  - Kernelized Bayesian Softmax
  - SeqGAN
  - Constrained Sentence Generation with CGMH

Challenges of the Discrete Output Space

Discrete outputs, especially the discrete sequence/structure outputs are non-trivial for handling in neural NLP.

Challenges of the Discrete Output Space

Discrete outputs, especially the discrete sequence/structure outputs are non-trivial for handling in neural NLP.

- From Continuous Outputs to Discrete Outputs
- Non-differentiable: fine for MLE but Non-trivial for other Training such as GAN
- Exponential Search Space
  - Hard for Global Inference
  - Hard for Constrained Decoding

Challenges of the Discrete Output Space

Discrete outputs, especially the discrete sequence/structure outputs are non-trivial for handling in neural NLP.

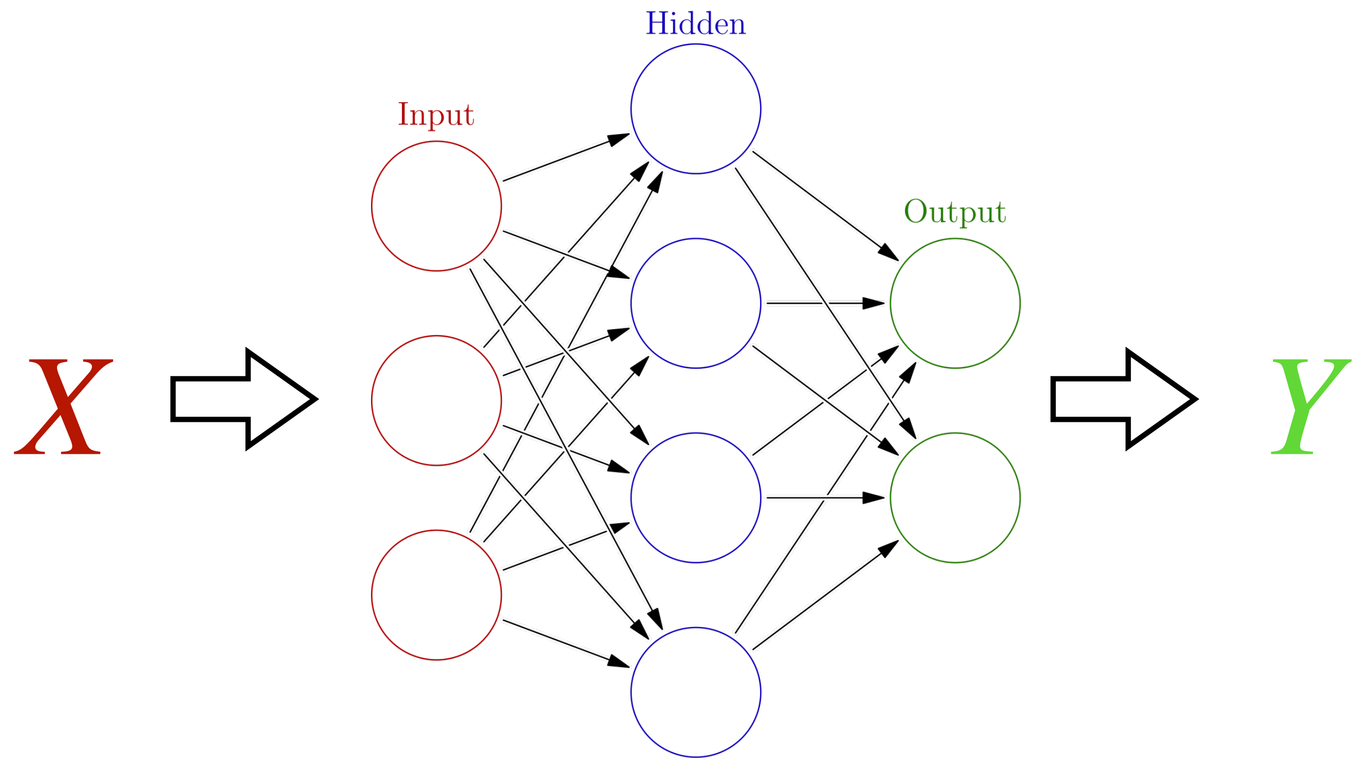
- From Continuous Outputs to Discrete Outputs
- Non-differentiable: fine for MLE but Non-trivial for other Training such as GAN
- Exponential Search Space
  - Hard for Global Inference
  - Hard for Constrained Decoding

In the next part, we will explain these challenges in detail and give some solutions.

Roadmap

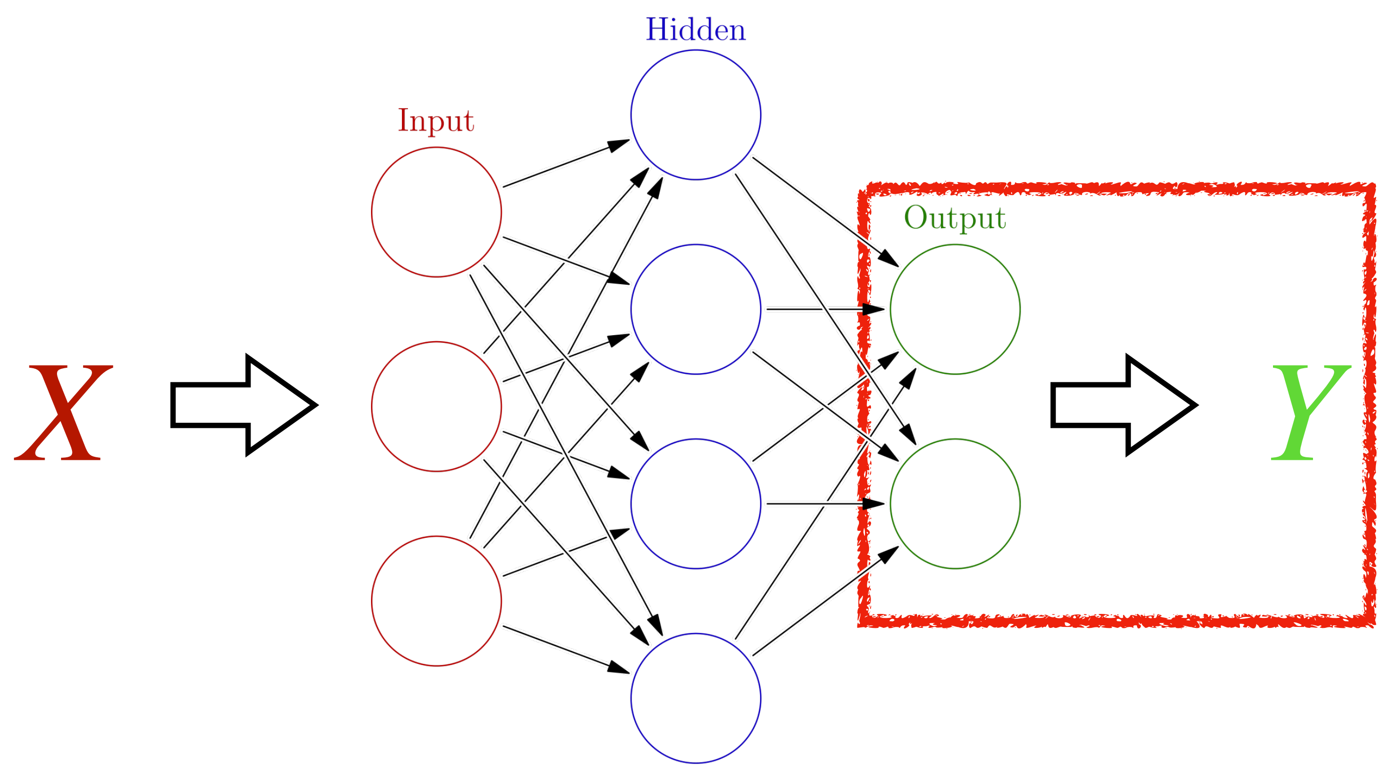
- Examples of discrete output space
- Challenges and Solutions of Discrete Output Space
  - From Continuous Outputs to Discrete Outputs
    - Embedding Matching by Softmax
  - Non-differentiable: Difficult for non-MLE training (e.g., GAN)
    - RL for Generation
    - Gumbel Softmax for Generation
  - Exponential Search Space
    - Hard for Global Inference
    - Hard for Constrained Decoding
- Case Study
  - Kernelized Bayesian Softmax
  - SeqGAN
  - Constrained Sentence Generation with CGMH

From Continuous to Discrete Outputs



$P(Y|X)$

From Continuous to Discrete Outputs



$P(Y \mid X)$

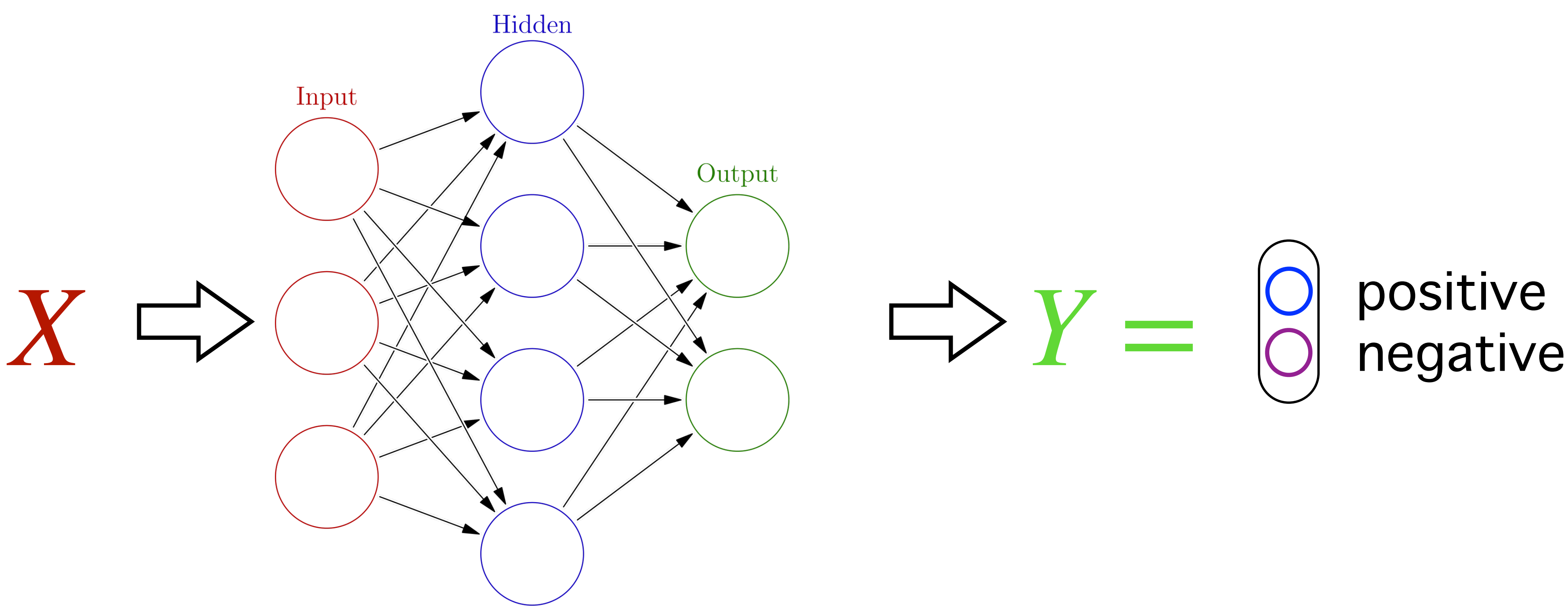
How to transform continuous outputs to discrete Y?

Embedding Matching by Softmax

A simple sentiment classification case:

Embedding Matching by Softmax

A simple sentiment classification case:



$P(Y|X)$ $\qquad$ $softmax(o \times W)$

MLE for Training

Maximum Likelihood Estimation:

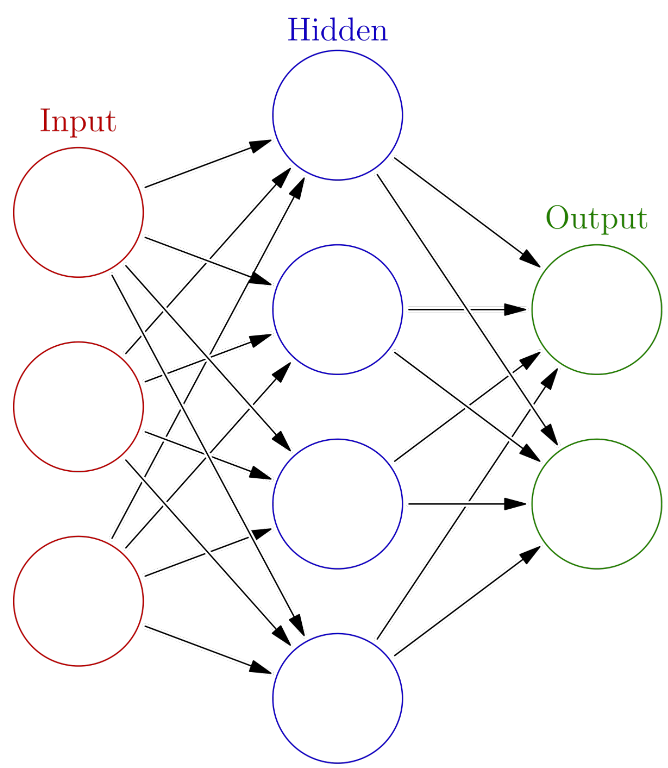
$$\min \mathbb{E}_{<X,Y>\sim p_{data}}[-\log p_\theta(Y|X)]$$

$$p_\theta(Y'|X) = \frac{\sigma(Y'|X)}{\sum_Y \sigma(Y|X)}$$

Partition function: two possibilities, namely, positive or negative.

How about Sequence

$X \Rightarrow$



$\Rightarrow Y =$ I like this tutorial

$$P(Y|X)$$

Exponential Hypothesis Space!

Maximum Likelihood Estimation:

$$min\ \mathbb{E}_{<X,Y>\sim p_{data}}[-log\ p_\theta(Y|X)]$$

$$p_\theta(Y') = \frac{\sigma(Y'|X)}{\sum_Y \sigma(Y|X)}$$

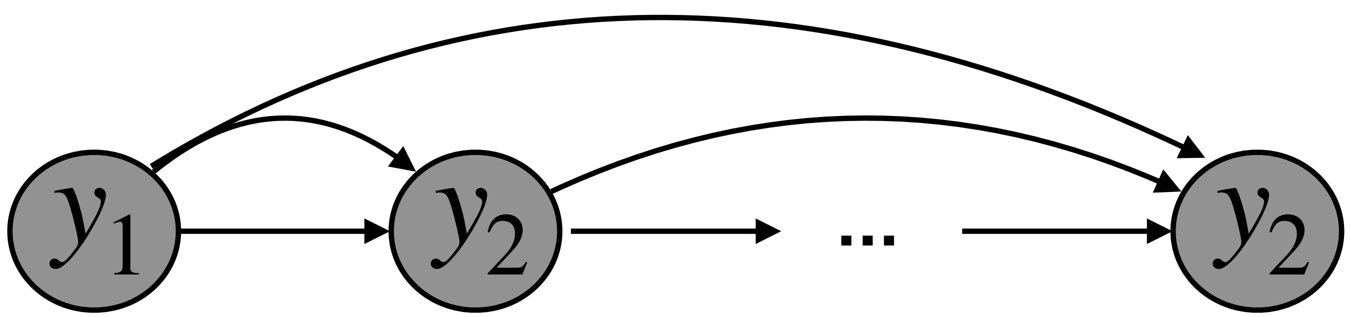
Calculating partition function directly requires **exponential time!**

$$\mathscr{V} \times \mathscr{V} \times \mathscr{V} \times \mathscr{V} \quad = \mathscr{V}^4$$

I like this tutorial

Locally Normalized Factorization

- Directed, fully-observed Bayesian network:



Decompose the joint distribution as a product of tractable conditionals:

Given $Y = [y_1, y_2, y_3 \ldots, y_n]$

$$p_\theta(Y) = \prod_{i=1}^{n} p_\theta(y_i \mid y_1, y_2, \ldots, y_{i-1}) = \prod_{i=1}^{n} p_\theta(y_i \mid y_{<i})$$

Exponential Hypothesis Space!

Maximum Likelihood Estimation:

$$min \ \mathbb{E}_{<X,Y>\sim p_{data}}[-log \ p_\theta(Y|X)]$$

$$p_\theta(Y') = \frac{\sigma(Y'|X)}{\sum_Y \sigma(Y|X)}$$

Calculating partition function directly requires **exponential time!**

But, under certain model structure, it is possible to computing within polynomial time

Tractable for Computing by Step by Step Factorization

$$p_\theta(y_i' \mid y_{<i}, X) = \frac{\sigma(y_i' \mid y_1 \ldots y_{i-1}, X)}{\sum_{y_i'} \sigma(y_i \mid y_1 \ldots y_{i-1}, X)}$$

Vocabulary Size

 Tractable for computing

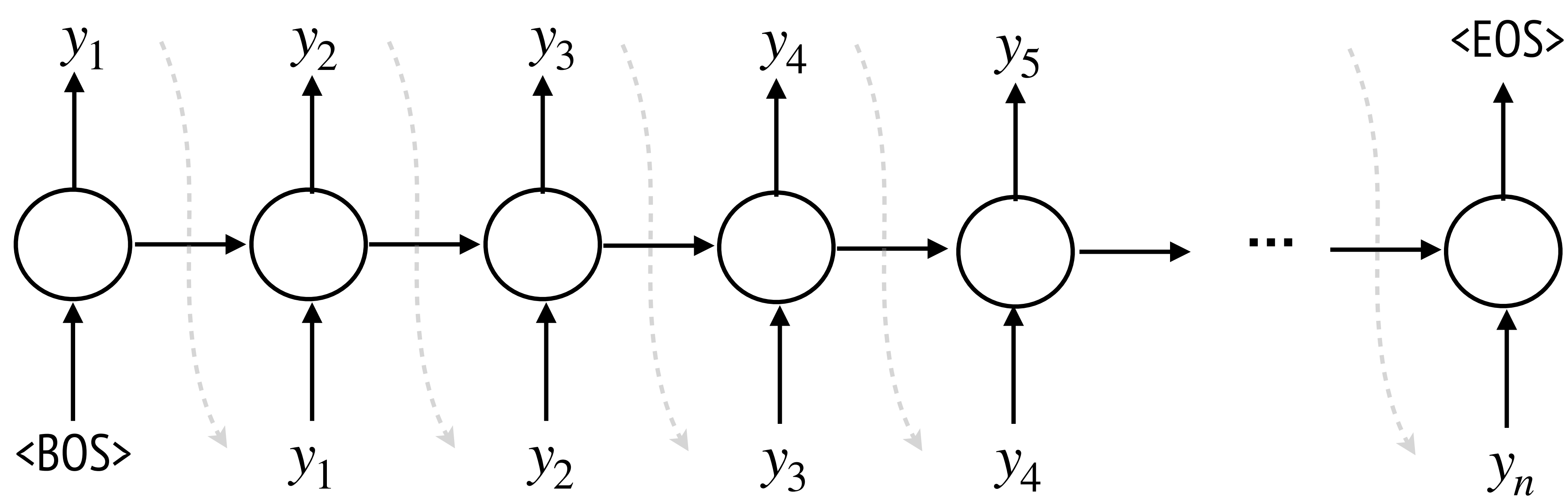
Parameterization by Neural Networks

$$p_\theta(y_i' \mid y_{<i}, X) = \frac{\sigma(y_i' \mid y_1 \ldots y_{i-1}, X)}{\sum_{y_i'} \sigma(y_i \mid y_1 \ldots y_{i-1}, X)}$$

Parameterization by RNN

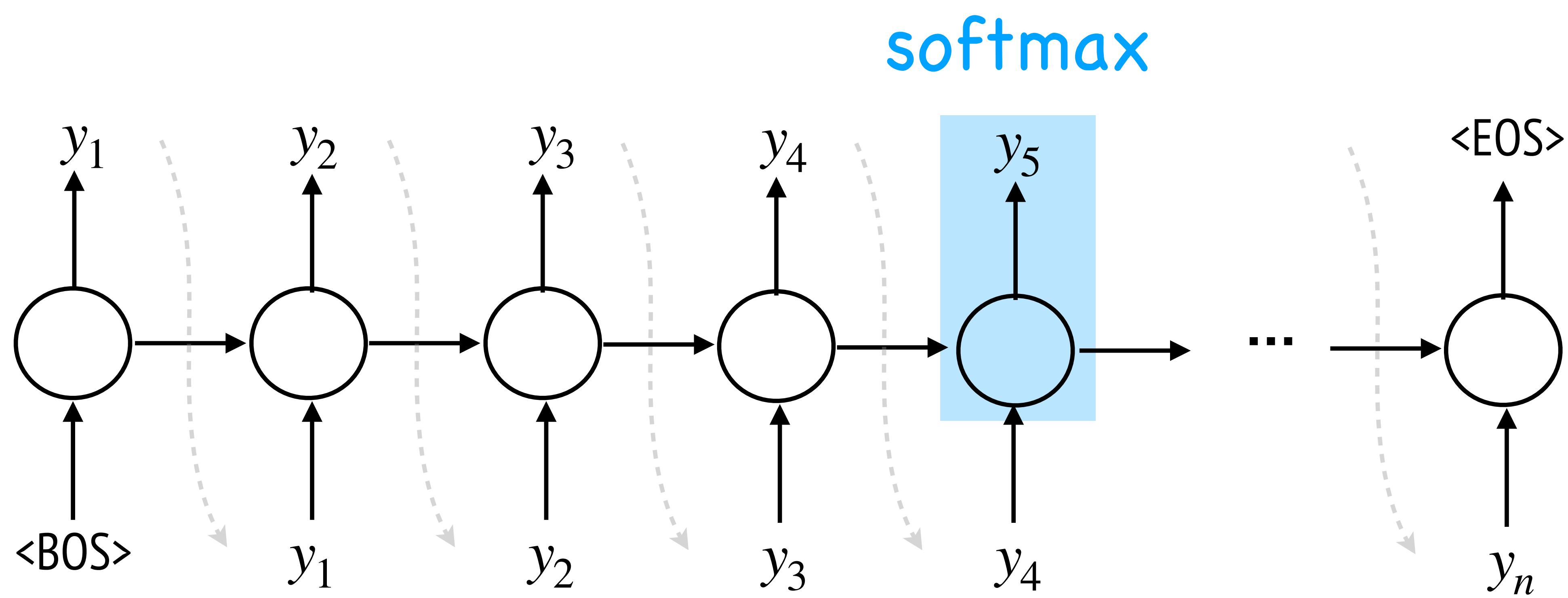
Text Generation as an Example

$$p_\theta(y_i' \mid y_{<i})$$



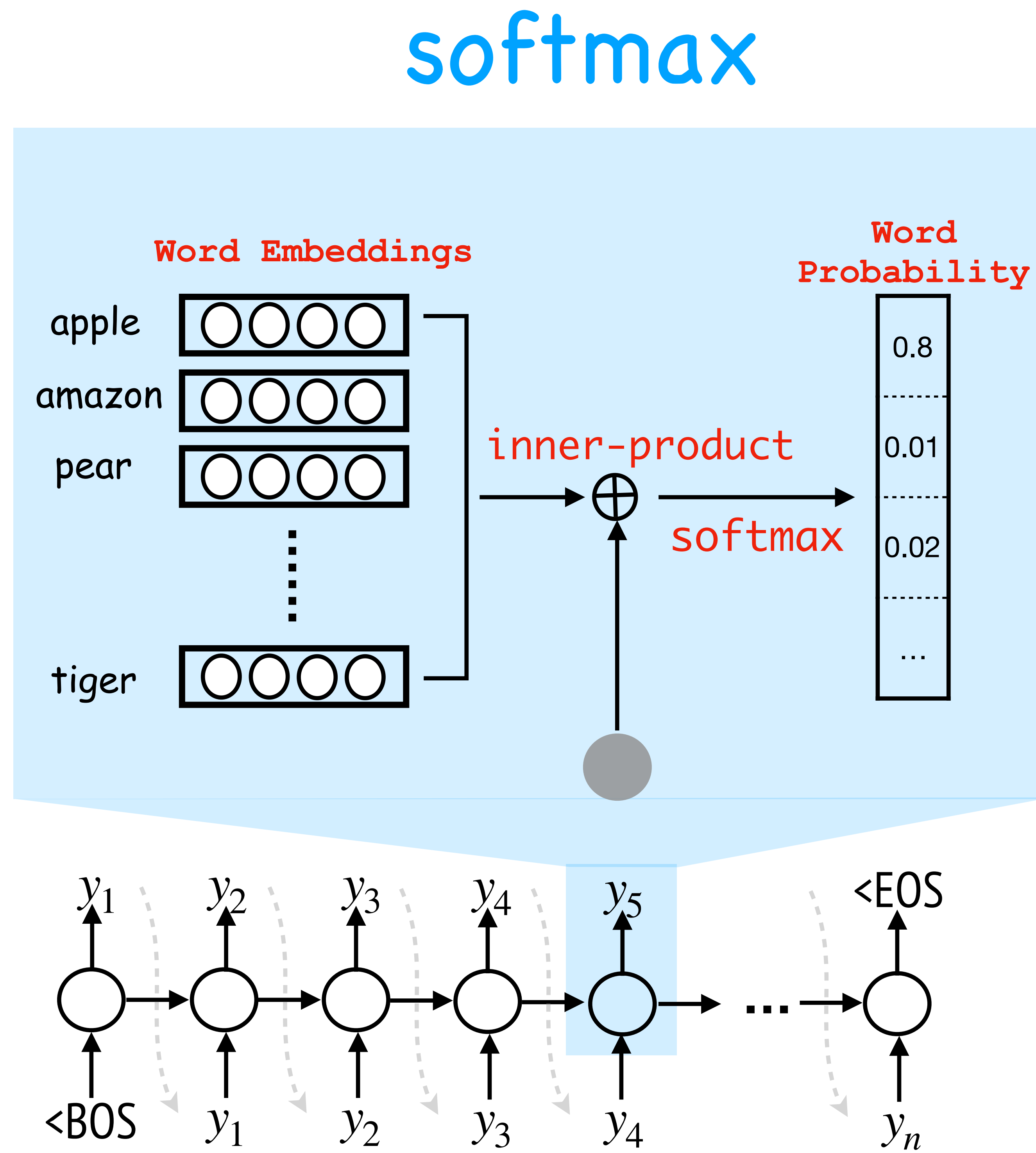
Softmax at Each Time Step

$$p_\theta(y_i' \mid y_{<i})$$

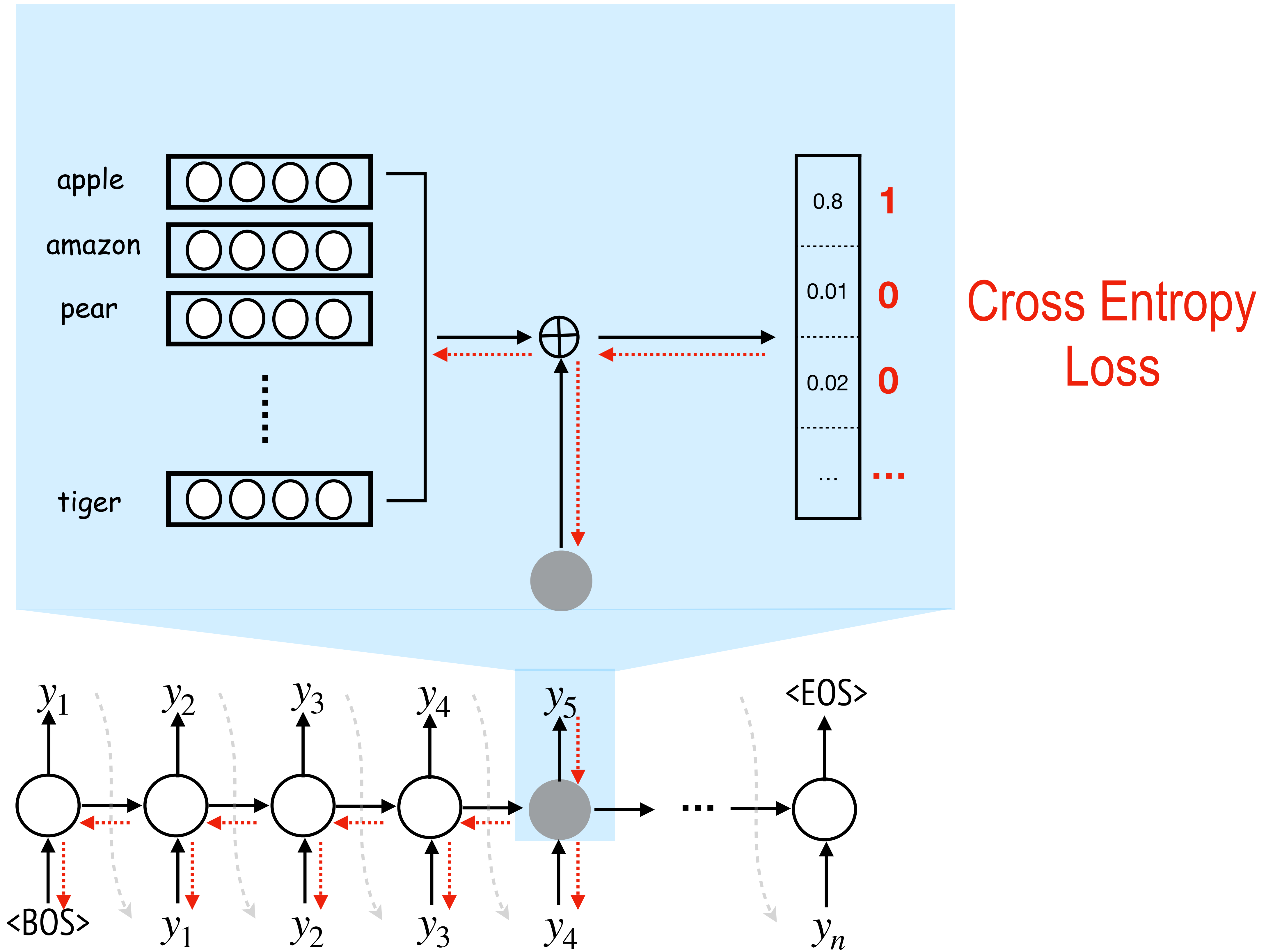


Embedding Matching inside the Softmax

$$p_\theta(y_i' \mid y_{<i})$$

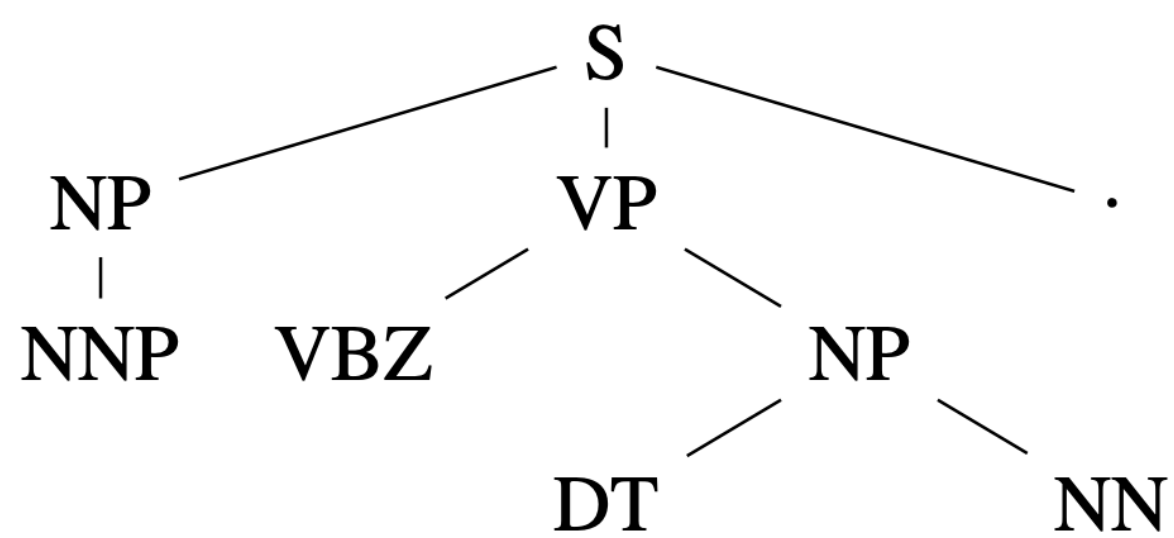


BackPropagation



Structures as Sequence Prediction

John has a dog . →

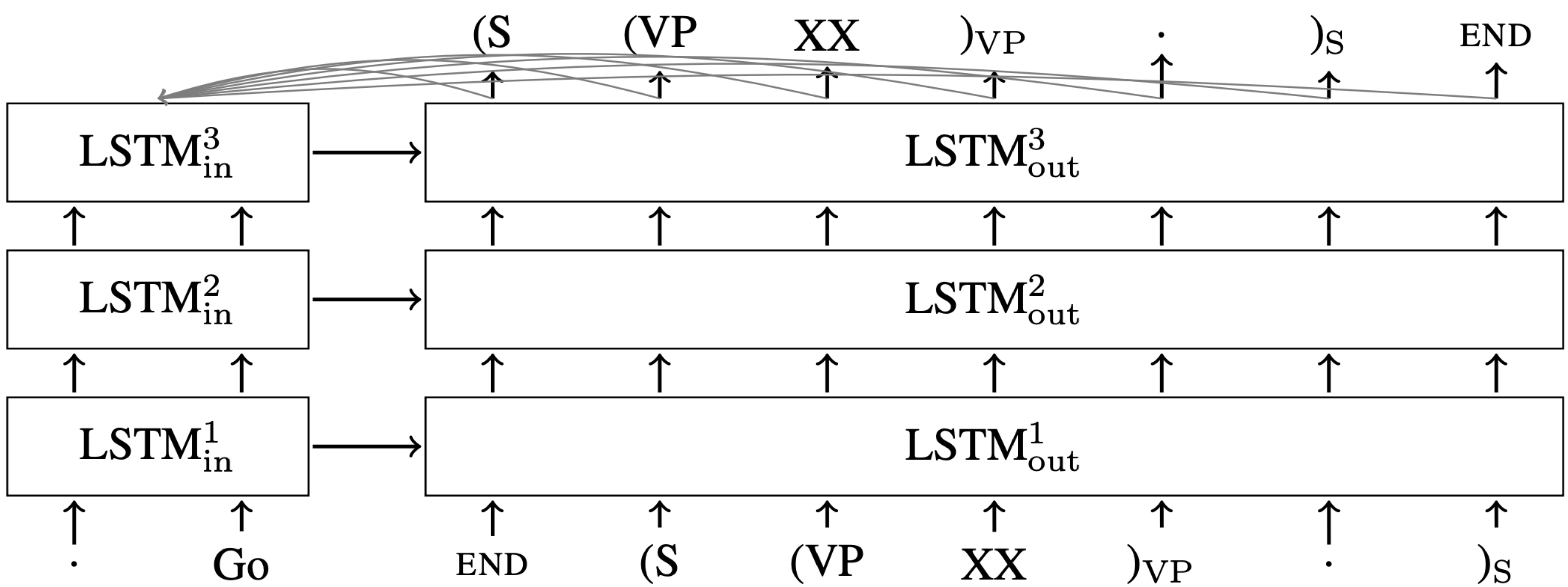


John has a dog . → (S (NP NNP $)_{\mathrm{NP}}$ (VP VBZ (NP DT NN $)_{\mathrm{NP}}$ $)_{\mathrm{VP}}$. $)_{\mathrm{S}}$

Linearizing the tree structure as a sequence of syntax labels.

Vinyals O, Kaiser Ł, Koo T, et al. Grammar as a foreign language, in NIPS, 2015.

Learning and Predicting Trees as a Sequence



Modeling the syntax parsing problem as a sequence to sequence prediction.

Vinyals O, Kaiser Ł, Koo T, et al. Grammar as a foreign language, in NIPS, 2015.

Roadmap

- Examples of discrete output space
- Challenges and Solutions of Discrete Output Space
  - From Continuous Outputs to Discrete Outputs
    - Embedding Matching by Softmax
  - Non-differentiable: Difficult for non-MLE training (e.g., GAN)
    - RL for Generation
    - Gumbel Softmax for Generation
  - Exponential Search Space
    - Hard for Global Inference
    - Hard for Constrained Decoding
- Case Study
  - Kernelized Bayesian Softmax
  - SeqGAN
  - Constrained Sentence Generation with CGMH

Non-Differentiable Problem

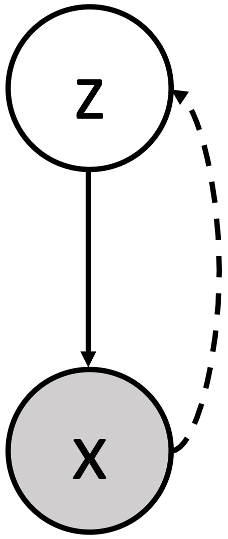
$$X \xrightarrow[P(Y \mid X)]{\text{neural networks}} Y$$

Non-Differentiable

- Fine for MLE but Non-trivial for other Training such as GAN.

$$X \xrightarrow[P(Y|X)]{\text{neural networks}} \boxed{Y} \longleftarrow \boxed{\text{Discriminator}}$$

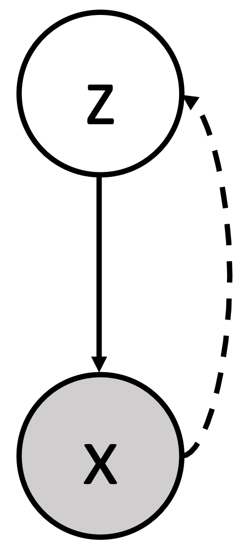
What's GAN ?



Generative Adversarial Networks:

$$\begin{aligned}\min_G \max_D L(D,G) &= \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \\ &= \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{x\sim p_g(x)}[\log(1 - D(x)]\end{aligned}$$

Generator vs. Discriminator



Generative Adversarial Networks:

$$\begin{aligned}\min_G \max_D L(D, G) &= \mathbb{E}_{x \sim p_r(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))] \\ &= \mathbb{E}_{x \sim p_r(x)}[\log D(x)] + \mathbb{E}_{x \sim p_g(x)}[\log(1 - D(x)]\end{aligned}$$

Discriminator Generator

Objective Revisit

Generative Adversarial Networks:

$$\begin{aligned}\min_G \max_D L(D, G) &= \mathbb{E}_{x \sim p_r(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))] \\ &= \mathbb{E}_{x \sim p_r(x)}[\log D(x)] + \mathbb{E}_{x \sim p_g(x)}[\log(1 - D(x))]\end{aligned}$$

Discriminator

Objective Revisit

Generative Adversarial Networks:

$$\min_G \max_D L(D, G) = \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]$$
$$= \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{x\sim p_g(x)}[\log(1 - D(x)]$$

Decoder

BackPropagation Fails

$$\begin{aligned}\min_G \max_D L(D,G) &= \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \\ &= \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{x\sim p_g(x)}[\log(1 - D(x))]\end{aligned}$$

Text is discrete, hard to propagate gradients from D to G !

Using RL or Gumbel Softmax

The same techniques used in dealing with the latent space such as RL or Gumbel softmax could also be adopted for handling the discrete output space.

Roadmap

- Examples of discrete output space
- Challenges and Solutions of Discrete Output Space
  - From Continuous Outputs to Discrete Outputs
    - Embedding Matching by Softmax
  - Non-differentiable: Difficult for non-MLE training (e.g., GAN)
    - RL for Generation
    - Gumbel Softmax for Generation
  - Exponential Search Space
    - Hard for Global Inference
    - Hard for Constrained Decoding
- Case Study
  - Kernelized Bayesian Softmax
  - SeqGAN
  - Constrained Sentence Generation with CGMH

Exponential Hypothesis Space

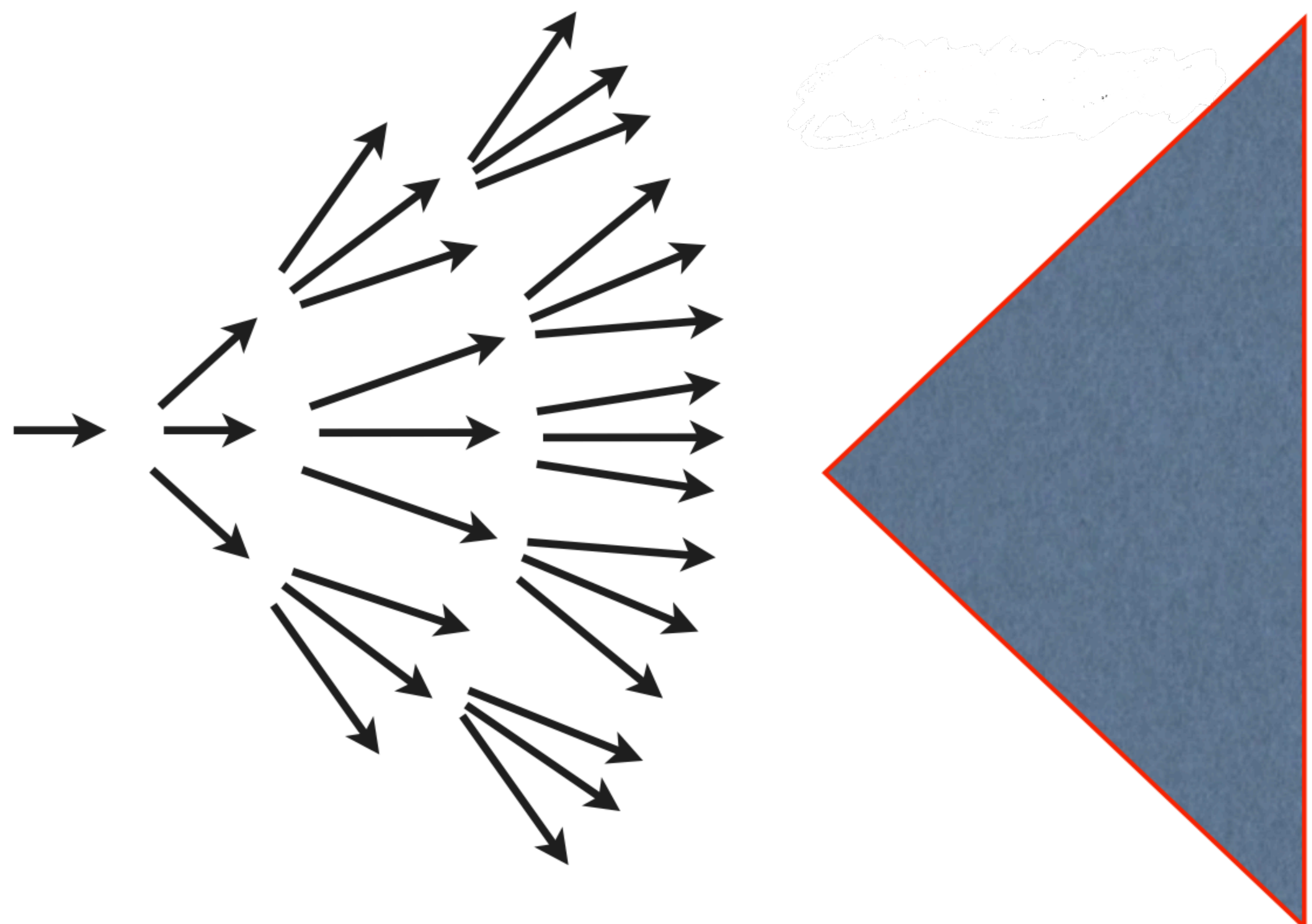


Figure from Liang Huang's slides

Hard for Global Inference

- Inference for decoding
  - Hard to yield the best scored output in the exponential space
- Inference in training (globally normalized model)
  - Non-trivial to compute the partition function

Inference for Decoding

$$\arg\max_{Y} \log p_\theta(Y \mid X) = \arg\max_{Y} \sum_{i=1}^{n} \log p_\theta(y_i \mid y_1, y_2, \ldots, y_{i-1}, X) = \arg\max_{Y} \sum_{i=1}^{n} \log p_\theta(y_i \mid y_{<i}, X)$$

Exponential search space

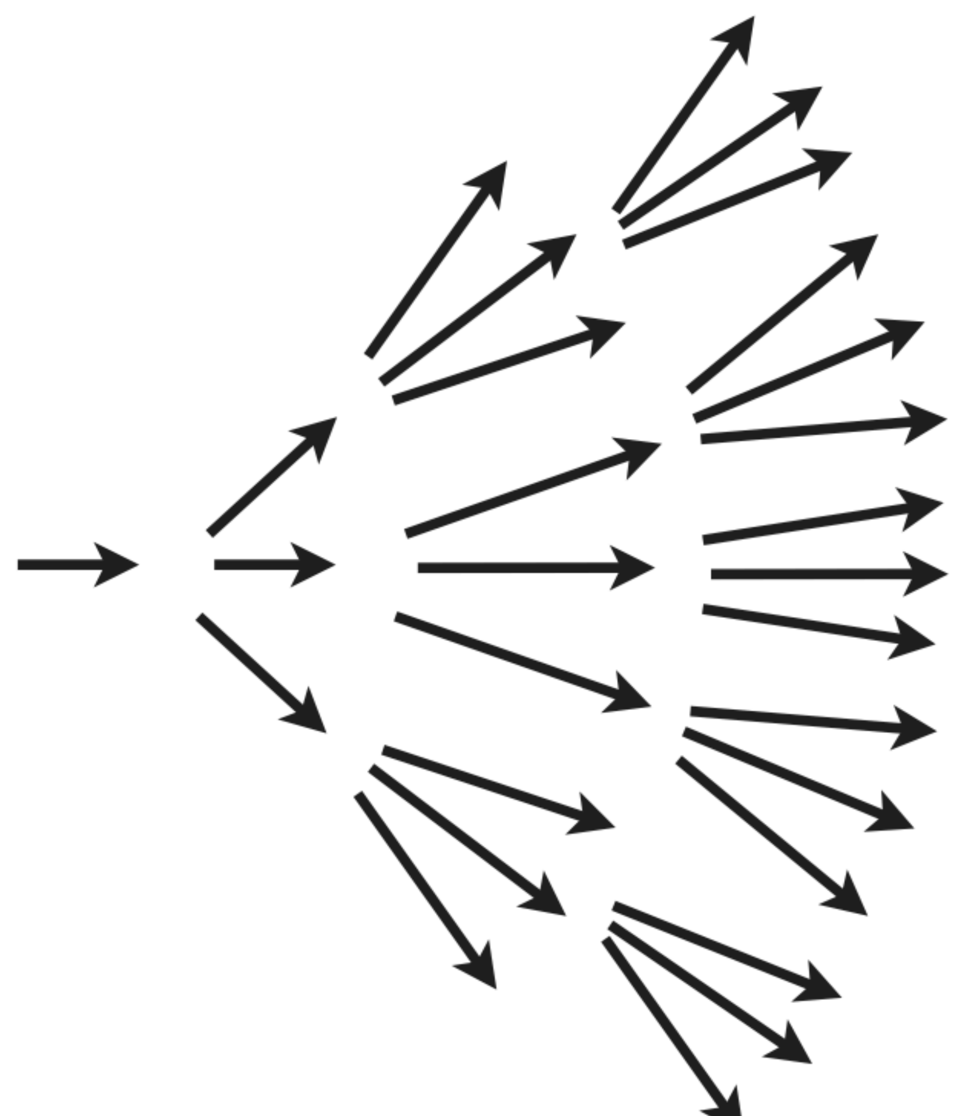
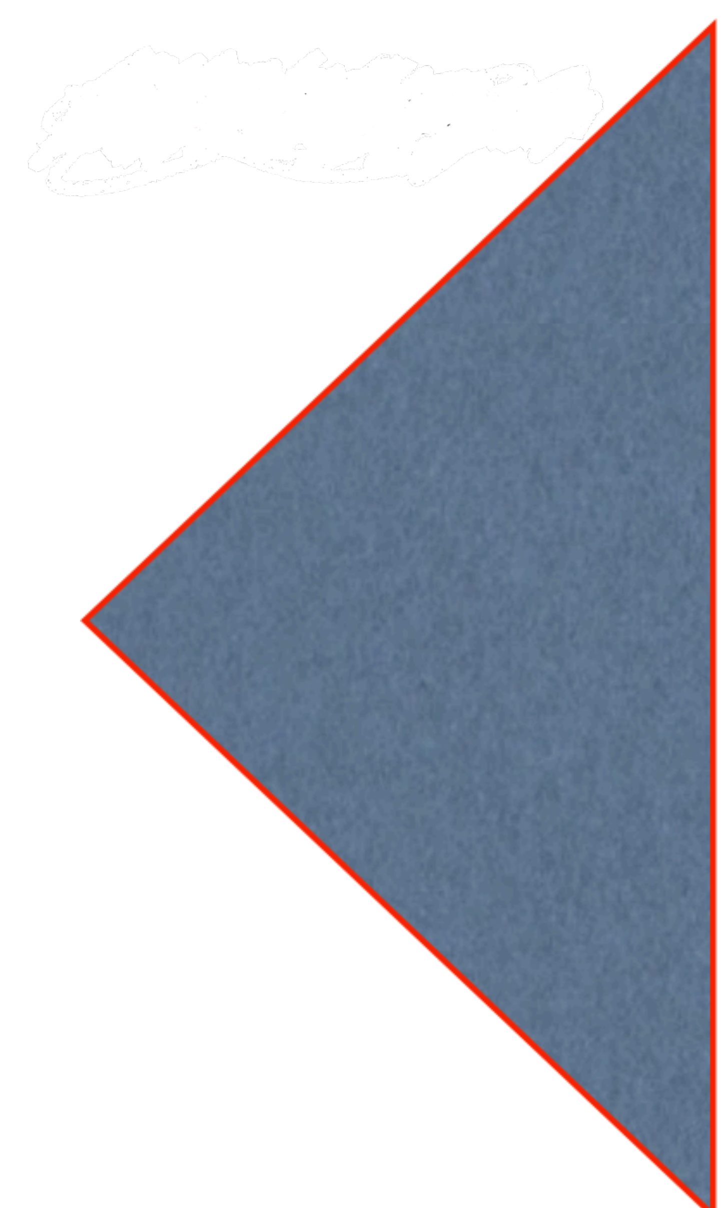


Figure from Liang Huang's slides



Beam Search

$$\underset{Y \in BEAM}{\arg\max} \log p_\theta(Y \mid X) = \underset{Y \in BEAM}{\arg\max} \sum_{i=1}^{n} \log p_\theta(y_i \mid y_{<i}, X)$$

Heuristic search by beam search

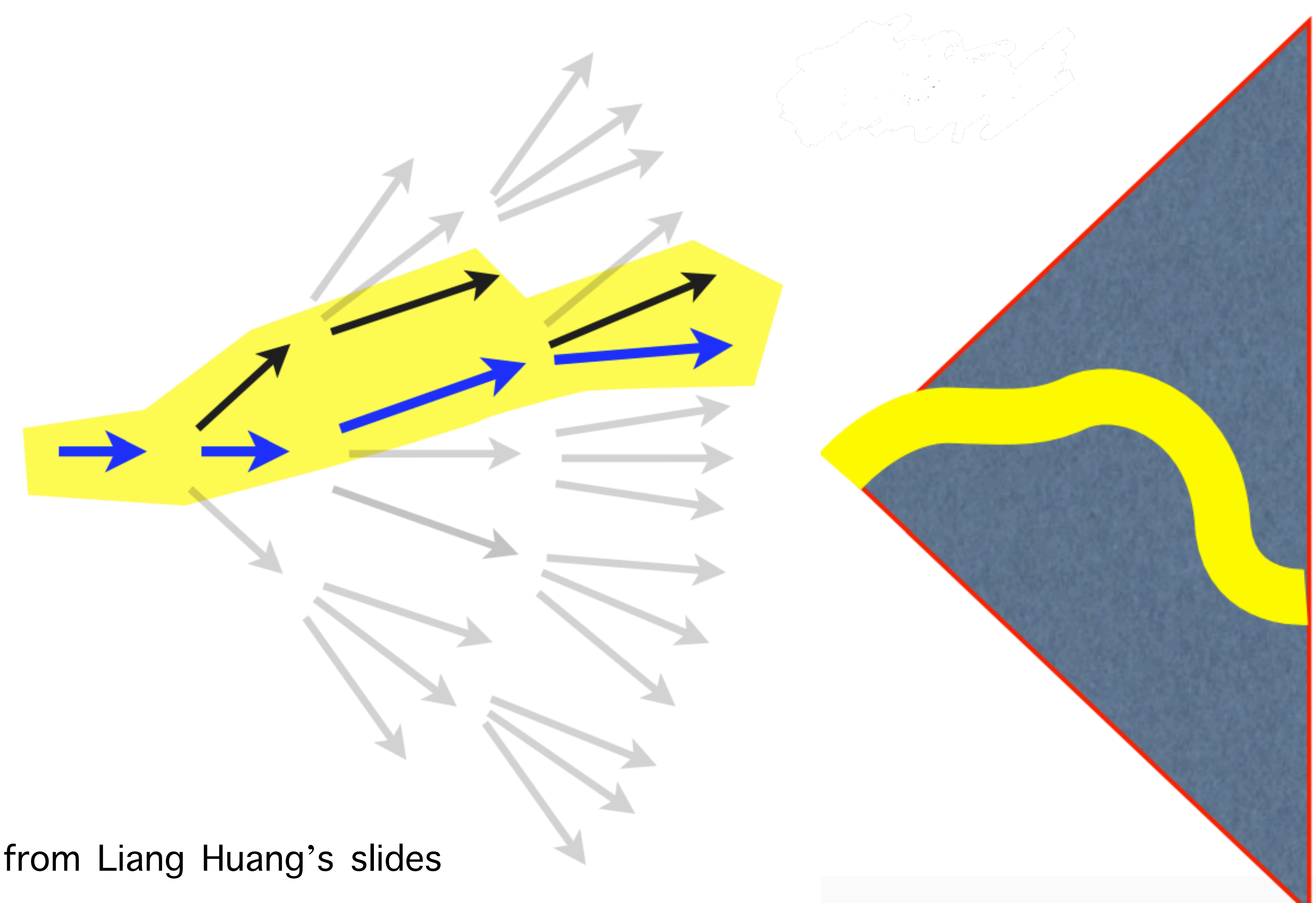


Figure from Liang Huang's slides

Inference in Training

Maximum Likelihood Estimation:

$$min\ \mathbb{E}_{<X,Y>\sim p_{data}}[-log\ p_\theta(Y|X)]$$

$$p_\theta(Y') = \frac{\sigma(Y'|X)}{\sum_Y \sigma(Y|X)}$$

Calculating partition function directly requires **exponential time**!

Approximated Globally Normalized Model

Maximum Likelihood Estimation:

$$min\ \mathbb{E}_{<X,Y>\sim p_{data}}[-log\ p_\theta(Y|X)]$$

$$p_\theta(Y') = \frac{\sigma(Y'|X)}{\boxed{\sum_Y \sigma(Y|X)}}$$

$$\Downarrow$$

$$\boxed{\sum_{Y \in BEAM} \sigma(Y|X)}$$

Inference in Training

Maximum Likelihood Estimation:

$$min\ \mathbb{E}_{<X,Y>\sim p_{data}}[-log\ p_\theta(Y|X)]$$

$$p_\theta(Y') = \frac{\sigma(Y'|X)}{\boxed{\sum_Y \sigma(Y|X)}}$$

$$\Downarrow$$

$$\boxed{\sum_{Y \in BEAM} \sigma(Y|X)}$$

Contrastive divergence using beam search as sampling

Inference in Training

Global Normalized Structured Prediction:

$$min\ \mathbb{E}_{<X,Y>\sim p_{data}}[-log\ p_\theta(Y|X)]$$

$$p_\theta(Y') = \frac{\sigma(Y'|X)}{\sum_{Y \in BEAM} \sigma(Y|X)}$$

Contrastive divergence using beam search as sampling

Hao Zhou, Yue Zhang, Shujian Huang and Jiajun Chen. A neural probabilistic structured-prediction model for transition-based dependency parsing, in ACL, 2015.

Daniel Andor, Chris Alberti, David Weiss, et al., 2016. Globally normalized transition-based neural networks, in ACL, 2016.

Wiseman S, Rush A M. Sequence-to-sequence learning as beam-search optimization, in EMNLP, 2016.

Challenges of Discrete Output Structures

- From Continuous Outputs to Discrete Outputs
- Non-differentiable: fine for MLE but Non-trivial for other Training such as GAN
- Exponential Search Space
  - Hard for Global Inference
  - Hard for Constrained Decoding

Constrained Decoding

Constrained Decoding:

$$\arg\max_{Y} \quad p_\theta(Y \mid X),$$

$$\textbf{s.t.} \qquad Y \text{ satisfy } \mathbf{C} = \{C_1, C_2, \ldots, C_n\}$$

The decoding outputs should satisfy a set of constraints.

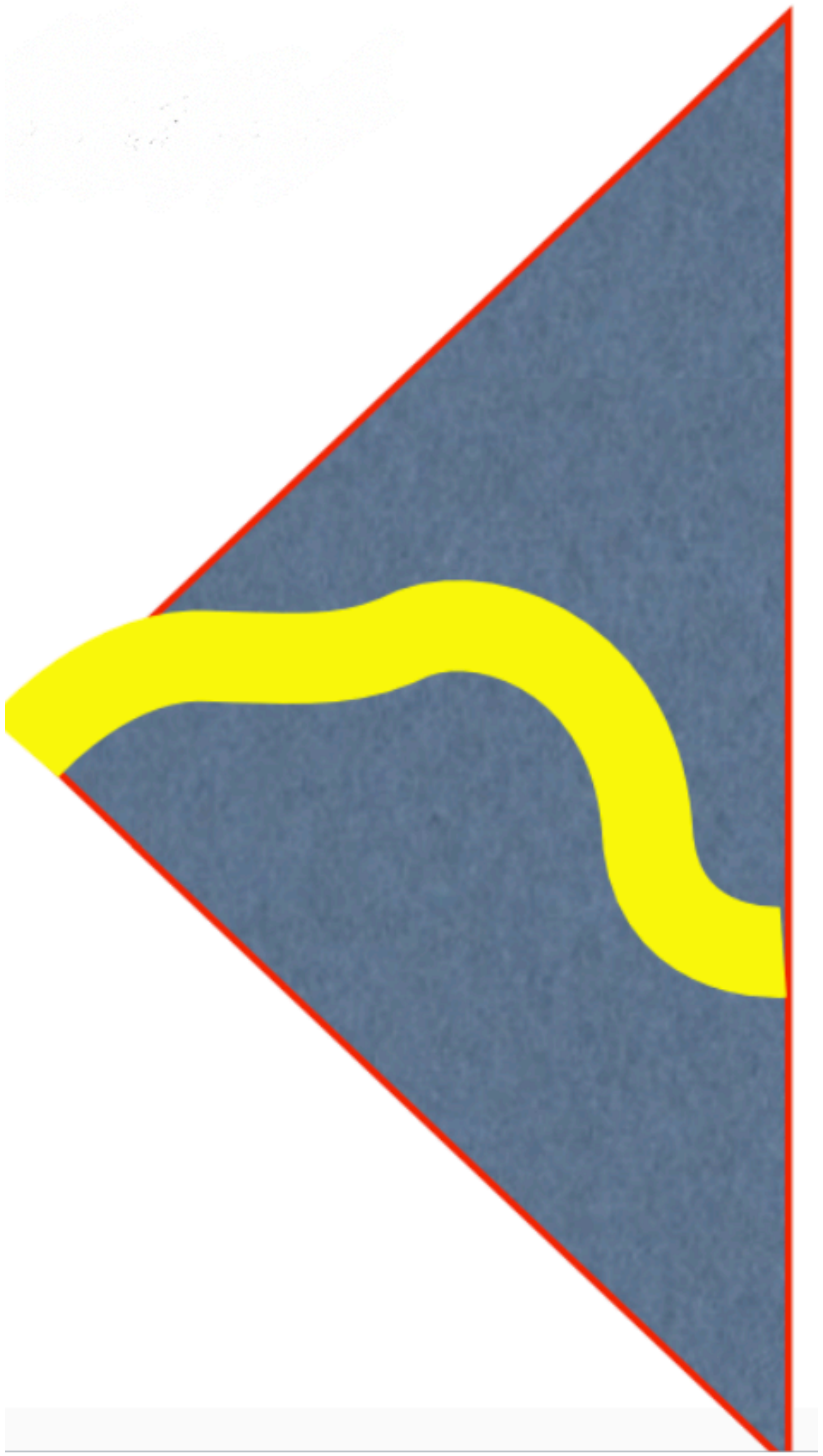
Constraints Definition

- Generating sentence satisfying constraints:
  - Hard constrains: Keyword must occur in sentences
    - E.g. Juice -> Brand natural juice, specially made for you

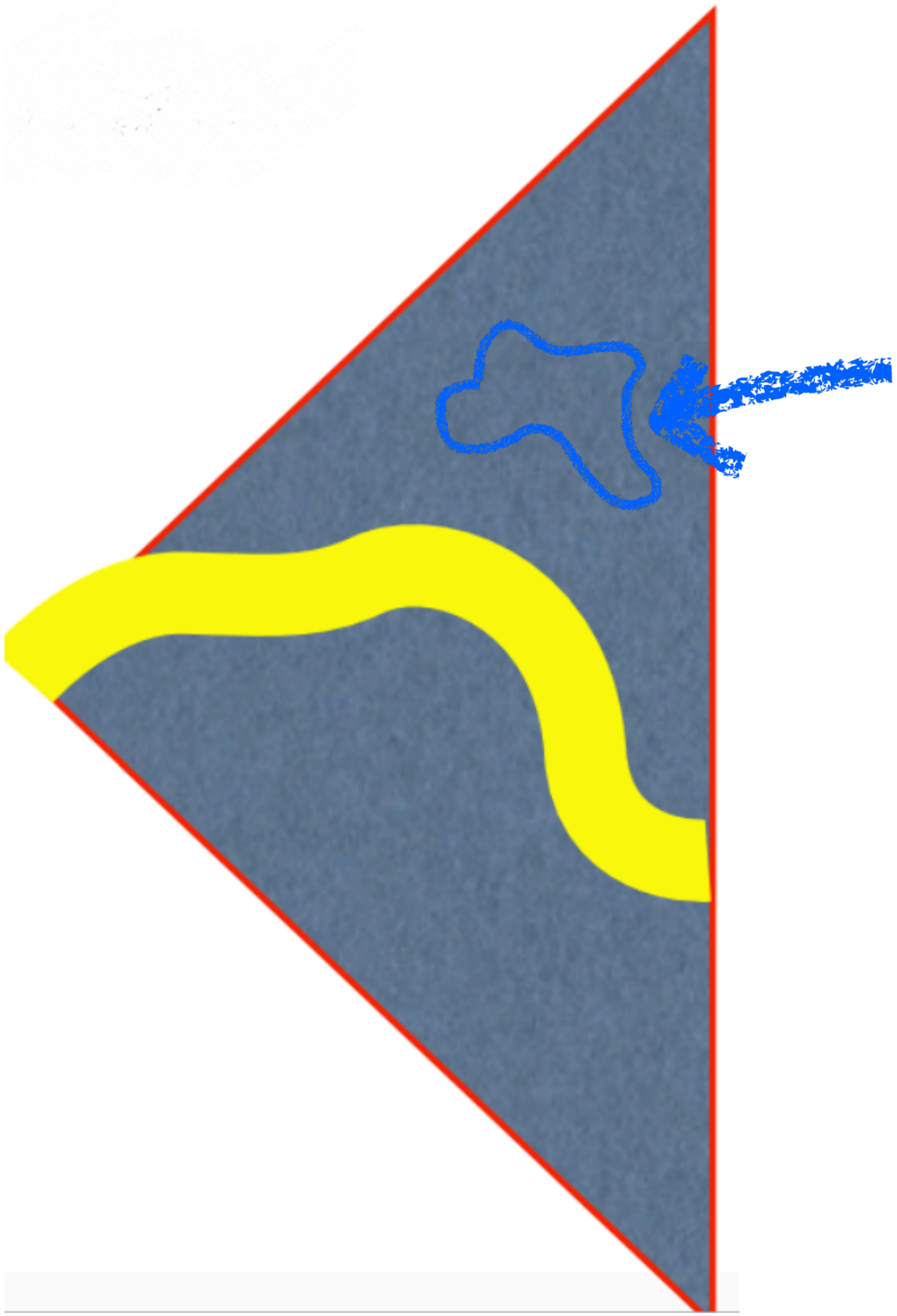
Constraints Definition

- Generating sentence satisfying constraints:
  - Hard constrains: Keyword must occur in sentences
    - E.g. Juice -> Brand natural juice, specially made for you
  - Soft constrains: Semantically similar to a given sentence (paraphrase)
    - E.g. The movie is a great success -> It is one of my favorite movies

Beam search over the Search Space

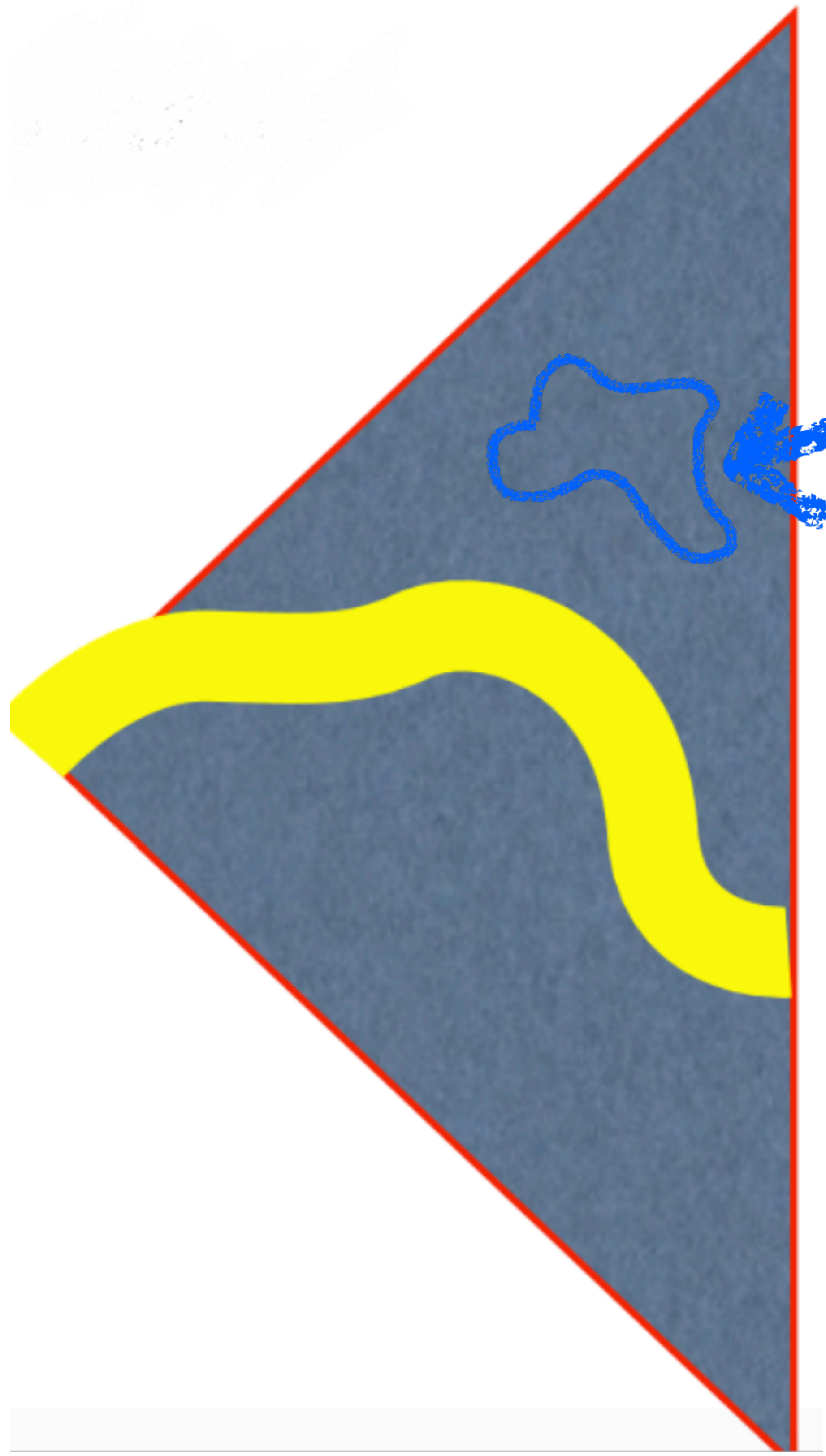


Vanilla Beam Search Fails



Desired outputs
satisfying constraints

Vanilla Beam Search Fails



Desired Outputs
satisfying constraints

Vanilla beam search may hardly find the
desired outputs under specific constraints.

Advanced Approaches

Targets of Constrained Decoding

Target Distribution of Constrained Decoding:

$$\pi(Y) = \underbrace{\prod_i p_\theta(y_i \mid y_{<i})}_{\text{Density of the original model}} \times \underbrace{\prod_{C \in \mathbf{C}} p_C(Y)}_{\text{Indicator functions for constraints}}$$

Density of the original model

Indicator functions for constraints

No Direct Sampling Method

Target Distribution of Constrained Decoding:

$$\pi(Y) = \underbrace{\prod_i p_\theta(y_i \mid y_{<i})}_{} \times \underbrace{\prod_{C \in \mathbf{C}} p_C(Y)}_{}$$

Density of the original model

Indicator functions for constraints

However, $\pi(Y)$ is quite high dimensional, and no direct sampling method.

Generation by Sampling

The constrained decoding problem turns to be sampling instances from a high dimensional distribution.

Generation by Sampling

The constrained decoding problem turns to be sampling instances from a high dimensional distribution.

Ning Miao, Hao Zhou, Lili Mou, Lei Li and Ruin Yan, CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling, in AAAI, 2019.

Main Idea of CGMH

- Instead of sampling from $\pi(x)$ directly, generating samples iteratively:
  - –Starting with initial keywords
  - –next sentence based on modification of previous
  - –action proposals to modify the sentences
- Metropolis-Hastings Algorithm

Miao et al., CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling, in AAAI, 2019.

Generation by Local Changes

- Suppose we have a blueprint

The book is interesting <EOS>

Generation by Local Changes

- Suppose we have a blueprint

The book is interesting <EOS>

~~The~~ book is interesting <EOS>

This

Generation by Local Changes

- Suppose we have a blueprint

The book is interesting <EOS>

~~The~~ book is interesting <EOS>

This

quite

Generation by Local Changes

- Suppose we have a blueprint

The book is interesting <EOS>

fascinating

~~The~~ book is ~~interesting~~ <EOS>

This

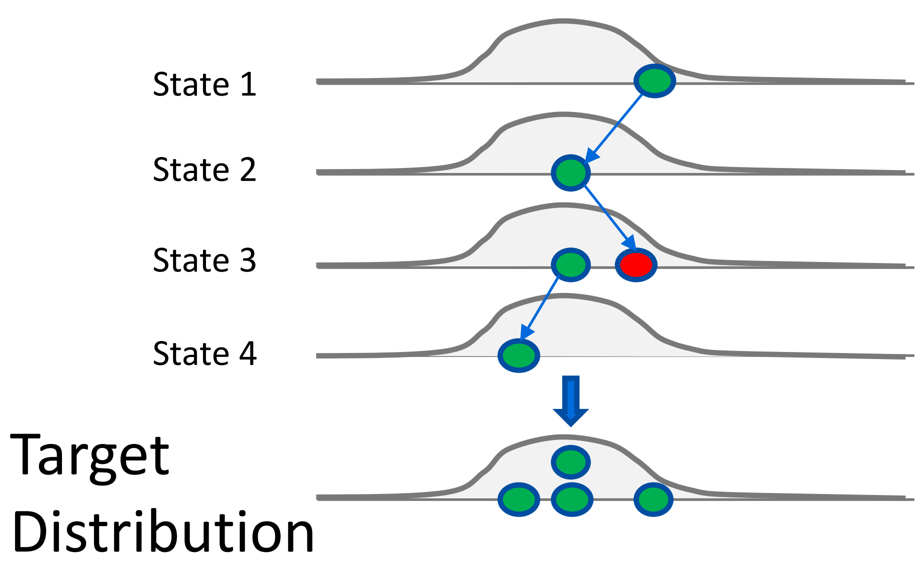
quite

Metropolis Hastings Sampling

Metropolis-Hastings(MH) perform sampling by first **proposes** a transition, and then **accepts or rejects** the transition.

$$A(x'|x_{t-1})$$
$$= \min(1, \frac{\pi(x') \cdot g(x_{t-1}|x')}{\pi(x_{t-1}) \cdot g(x'|x_{t-1})})$$

g is proposal distribution



Metropolis—Hastings Sampler

- **Algorithm**
  - Start from an arbitrary initial state $x^{(0)}$
  - For every step t

    $g(x' \mid x)$: arbitrary proposal distribution

    Propose a new state $x' \sim g(x' \mid x^{(t)})$

    Accept x' w.p. $A(x' \mid x) = \min\left\{1, \frac{p(x')g(x^{(t)} \mid x')}{p(x)g(x' \mid x^{(t)})}\right\}$, i.e.,

    $x^{(t+1)} = x'$

    Reject x' otherwise, i.e., $x^{(t+1)} = x^{(t)}$
  - Return $x^{(t)}$ with a large t

CGMH

CGMH performs constrained generation by:

1. Pretrain Language Model prob;
2. Start from a initial sentence;
3. Propose a new action and accept/reject the action.

CGMH: Action Proposal

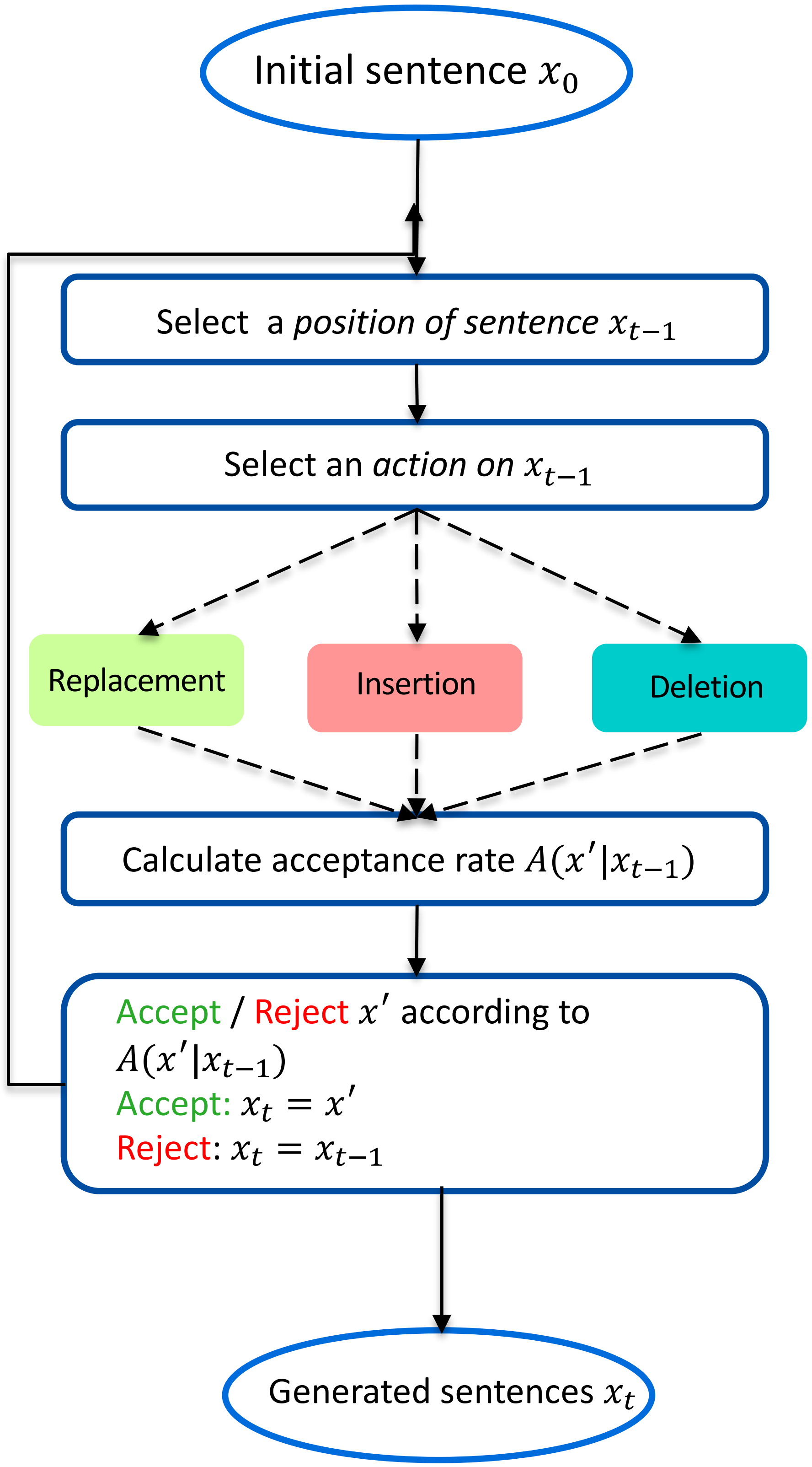
- We use MH algorithm to sample from $\pi(x)$
  - From a sentence x_{t-1}, we propose an action on one word of x_{t-1}.
  - Actions include:
    1. Replacement: change a word to another one
    2. Insertion: add a word
    3. Deletion: remove a word

CGMH: Acceptance Ratio

- Calculate the acceptance rate:

$$A(x'|x_{t-1}) = \min(1, \frac{\pi(x')\cdot g(x_{t-1}|x')}{\pi(x_{t-1})\cdot g(x'|x_{t-1})})$$

- Accept x' with probability $A(x'|x_{t-1})$



Proof Sketch (Cont.)

- MH Sampler satisfies detailed balance
  - $\forall x, y,$ if $x \neq y,$ $p(x) \cdot \mathcal{T}_{x \to y} = p(x) \cdot g(y|x) \cdot \min\left\{1, \frac{p(y)g(x|y)}{p(x)g(y|x)}\right\}$ (1)

    $p(y) \cdot \mathcal{T}_{y \to x} = p(y) \cdot g(x|y) \cdot \min\left\{1, \frac{p(x)g(y|x)}{p(y)g(x|y)}\right\}$ (2)
  - W.L.O.G., we assume $p(x)g(y|x) \geq p(y)g(x|y)$

    $(1) = p(y) \cdot g(x|y)$

    $(2) = p(y) \cdot g(x|y)$
  - $\forall x, y,$ if $x = y,$ $p(x)\mathcal{T}_{x \to y} = p(y)\mathcal{T}_{y \to x}$ also holds

Case Study

- Embedding Matching by softmax
  - Kernelized Bayesian Softmax
- RL for Generation
  - SeqGAN
- Generation by Sampling
  - Constrained Sentence Generation with CGMH
  - Generating Adversarial Examples for Natural Languages

Case Study 1

- Embedding Matching by Softmax
  - Kernelized Bayesian Softmax
- RL for Generation
  - SeqGAN
- Generation by Sampling
  - Constrained Sentence Generation with CGMH
  - Generating Adversarial Examples for Natural Languages

Kernelized Bayesian Softmax

Kernelized Bayesian Softmax

KerBS: Kernelized Bayesian Softmax

$$P(x_t = i) = \sum_{j \in 0,1,\ldots,N_i} P(x_t = s_i^j)$$

where $P(x_t = s_i^j) = \dfrac{\exp(\mathcal{K}_{\theta_i^j}(h_t, w_i^j))}{\sum_k \sum_{r \in 0,1,\ldots,N_k} \exp(\mathcal{K}_{\theta_k^r}(h_t, w_k^r))}$

$$\mathcal{K}_\theta(h, e) = |h||e|(a \exp(-\theta\ cos(h, e)) - a)$$

Here h is hidden state, e is embedding, θ is a parameter controlling the embedding variances of each sense and $a = \dfrac{-\theta}{2(\exp(-\theta) + \theta - 1))}$ is a normalization factor.

Ning Miao, Hao Zhou, Chengqi Zhao, Wenxian Shi and Lei Li, Kernelized Bayesian Softmax for Text Generation, in NeurIPS, 2019.

Why KerBS?

Model capacity of softmax is not OK 

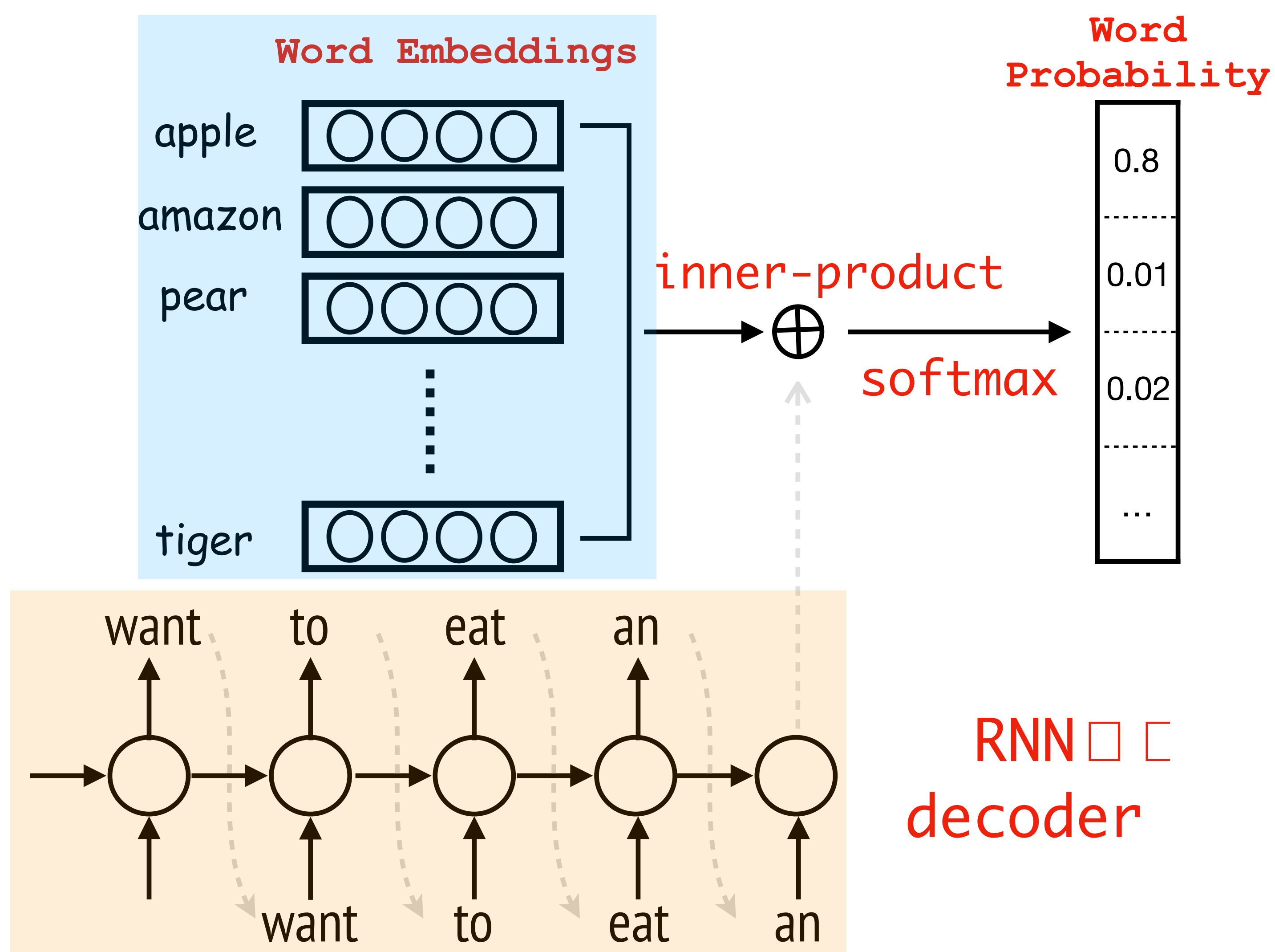
	Word2Vec	BERT
Category	Context Independent	Context Dependent
Capacity	Low	High
Performance	Bad	Good

Motivated by BERT, we may need context dependent embedding for text generation!

Text Generation as Matching

Text Generation is Embedding Matching

Context Independent Embedding



Context Dependent Embedding

RNN□ □
decoder

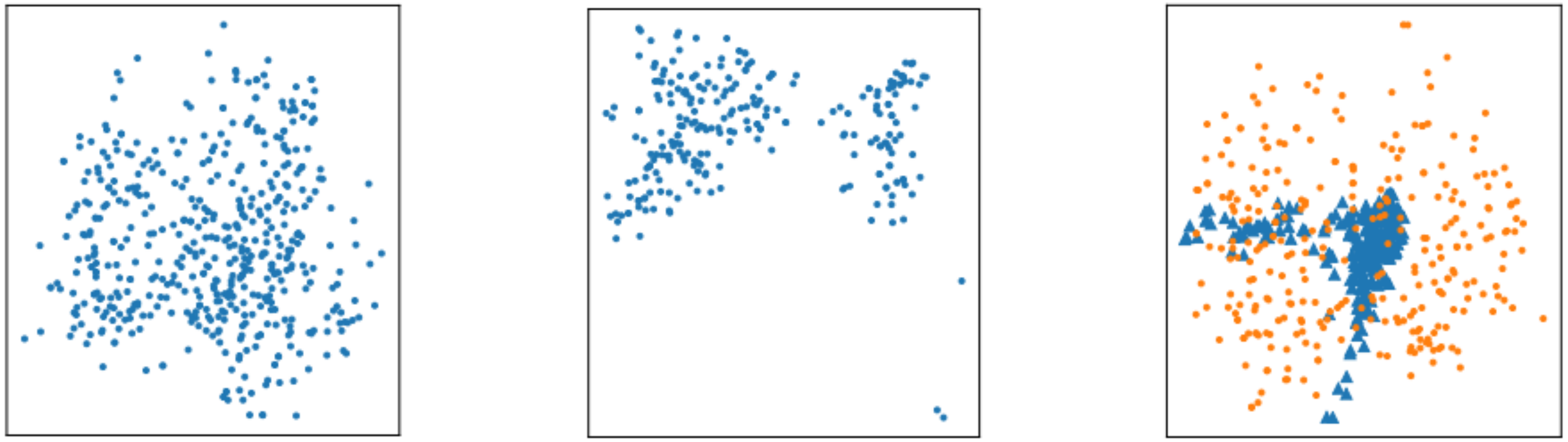
Bottleneck of Text Generation

Bottleneck of text generation is the softmax

Embedding matrix in softmax should have larger capacity.

Visualization of BERT

- Multi-Sense & Varying Variances

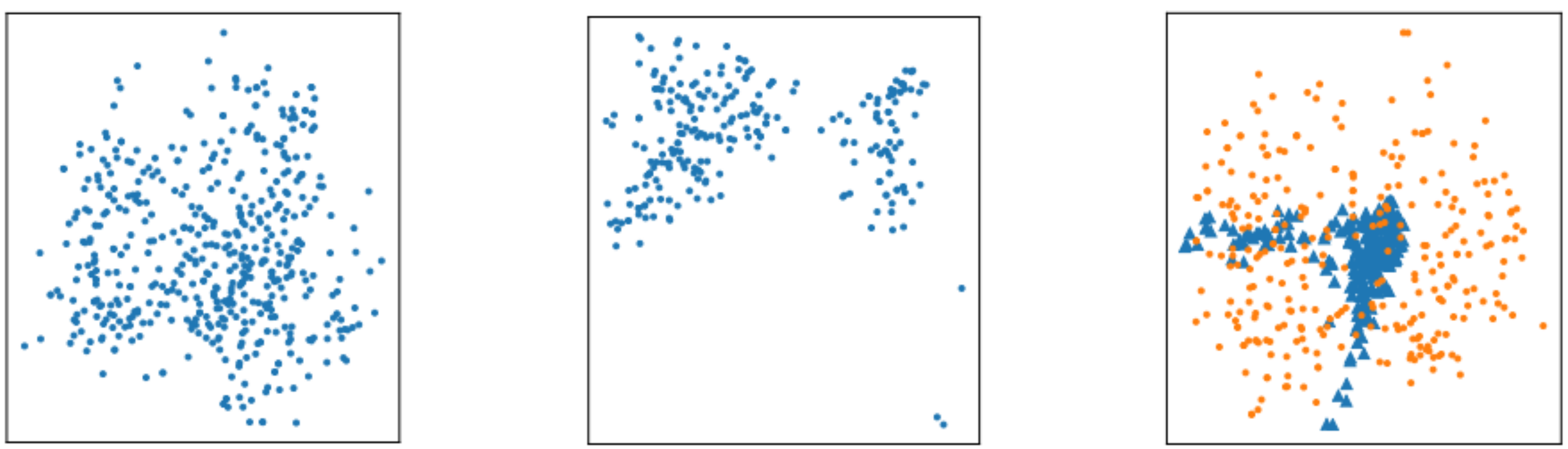


(a) computer (b) monitor (c) car and vehicle

Softmax ***can*** handle this situation

Visualization of BERT

- Multi-Sense & Varying Variances

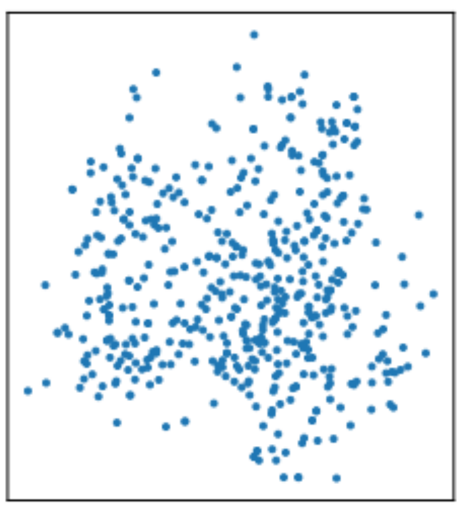


(a) computer (b) monitor (c) car and vehicle

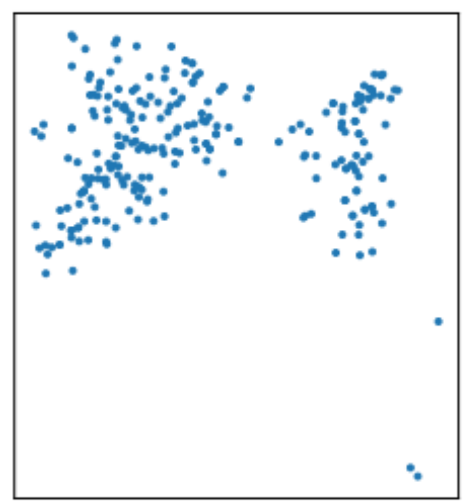
Softmax **can't** handle **multisense**.

Visualization of BERT

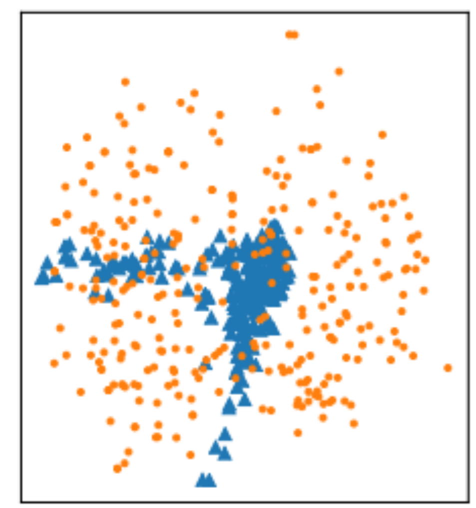
- Multi-Sense & Varying Variances



(a) computer



(b) monitor

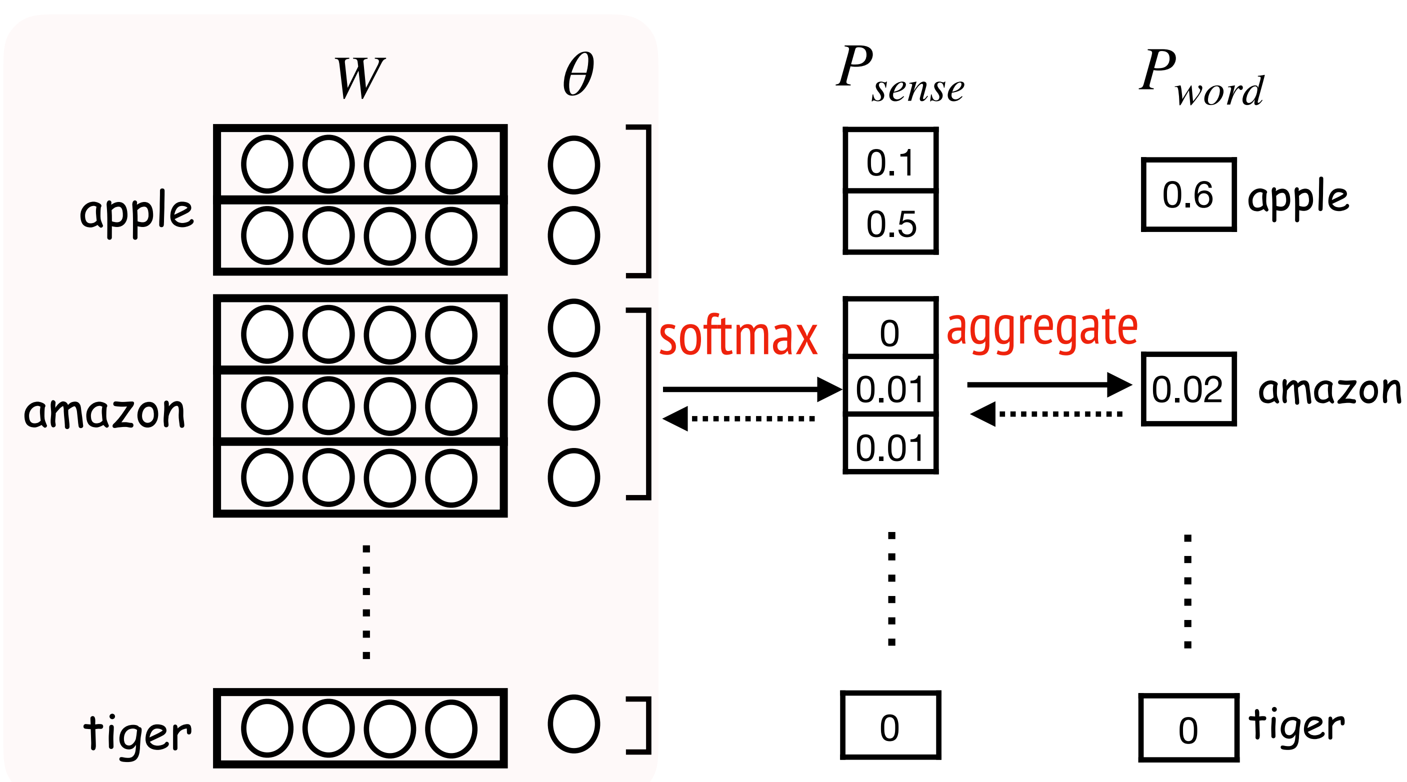


(c) car and vehicle

Softmax **can't** handle **multisense** and varying **variance**s.

KerBS - Multisense

Each word may have several senses. KerBS allocates a vector for each sense.



KerBS - Multisense

After getting the probabilities of each sense, KerBS sums up all sense probabilities of same word.

$$P(x_t = i) = \sum_{j \in 0,1,\ldots,N_i} P(x_t = s_i^j)$$

KerBS - Varying Variances

The distribution of each word's output vectors have different variances.

We use a variable kernel to represent varying variances.

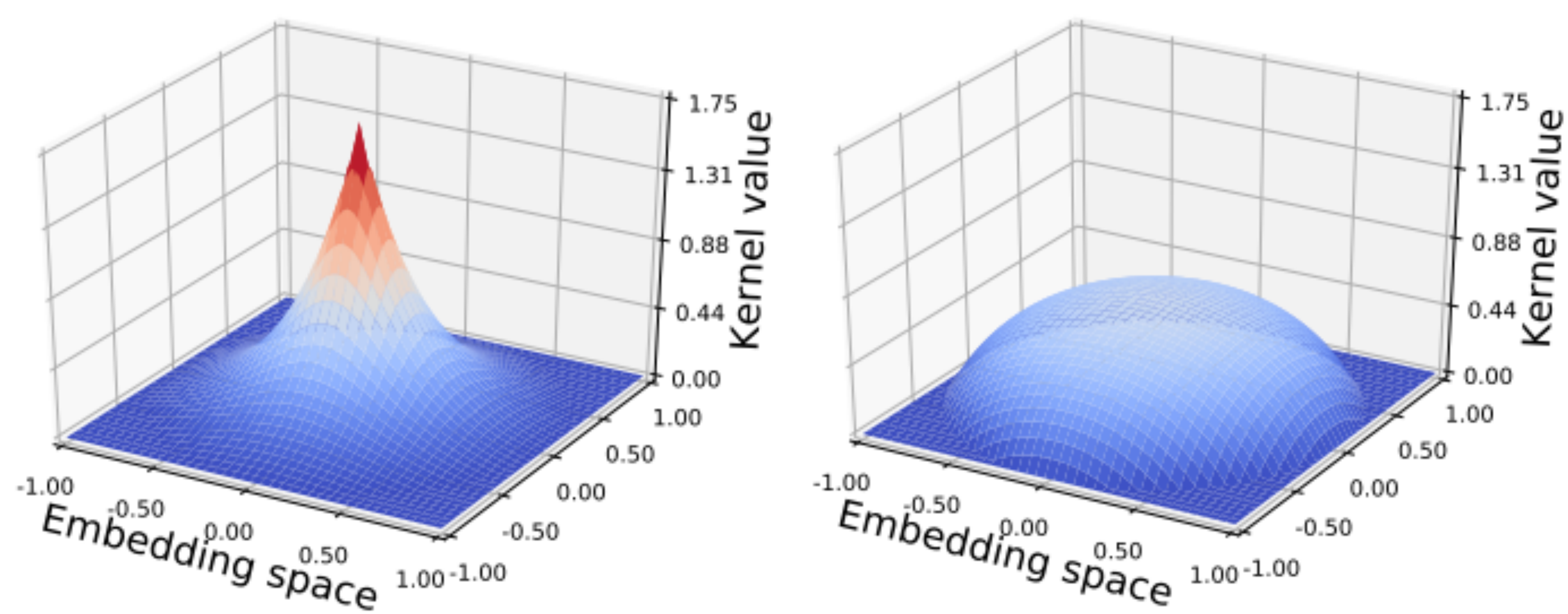
$$P(x_t = s_i^j) = \frac{\exp(\mathcal{K}_{\theta_i^j}(h_t, w_i^j))}{\sum_k \sum_{r \in 0,1,\ldots,N_k} \exp(\mathcal{K}_{\theta_k^r}(h_t, w_k^r))}$$

$$\mathcal{K}_\theta(h, e) = |h||e|(a \exp(-\theta\ cos(h, e)) - a)$$

Note that when $\theta \to 0, \mathcal{K}_\theta(h, e) \to |h||e|\ cos(h, e)$, which is regular Euclidean norm!

KerBS - Varying Variances

The distribution of each word's output vectors have different variances. We use a variable kernel to represent varying variances.



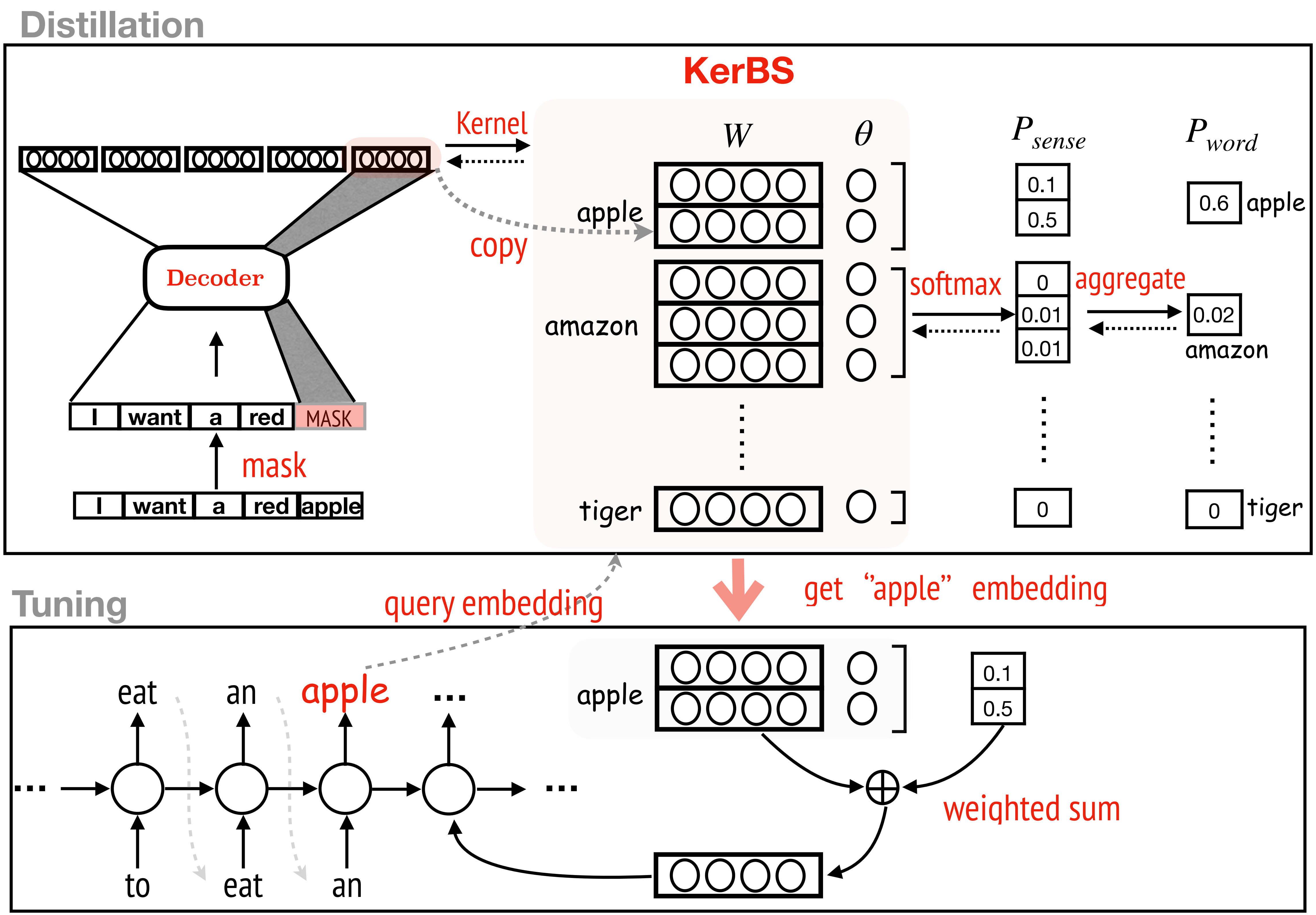
(a) θ = -2 (b) θ = 2

Figure 2: Kernel shapes of different θ.

How to decide the sense number of each word?

Dynamically change each word's sense number while training. Delete senses that are less used. Add senses to words which are not well fitted.

Dynamic Allocation



Theoretical Guarantee

Lemma 1

KerBS has the ability to learn the multi-sense property. If the real distribution of context vectors consists of several disconnected clusters, KerBS will learn to represent as many as these clusters

KerBS can capture the multi-sense property.

Lemma 2

KerBS has the ability to learn model variances. For distributions with larger variances, KerBS learns larger θ.

KerBS can learn varying variances.

Experiments-Setting

We test KerBS on 3 text generation tasks:

1. **Machine Translation (MT)** is conducted on IWSLT'16 De-En, which contains 196k pairs of sentences for training.
2. **Language modeling (LM)** is included. Following previous work, we use a 300k, 10k and 30k subset of One-Billion-Word Corpus for training, validating and testing.
3. **Dialog generation (Dialog)** is also included. We employ the DailyDialog dataset for experiment.

Main Results

Table 1: Performance of KerBS on Seq2Seq.

Tasks	Metrics	Seq2Seq	Seq2Seq+ MoS [Yang et al., 2018]	SeqSeq + KerBS
MT	BLEU-4	25.91	26.45	**27.28**
LM	PPL	103.12	102.72	**102.17**
Dialog	BLEU-1	16.56	13.73	**17.85**
	Human Eval.	1.24	1.04	**1.40**

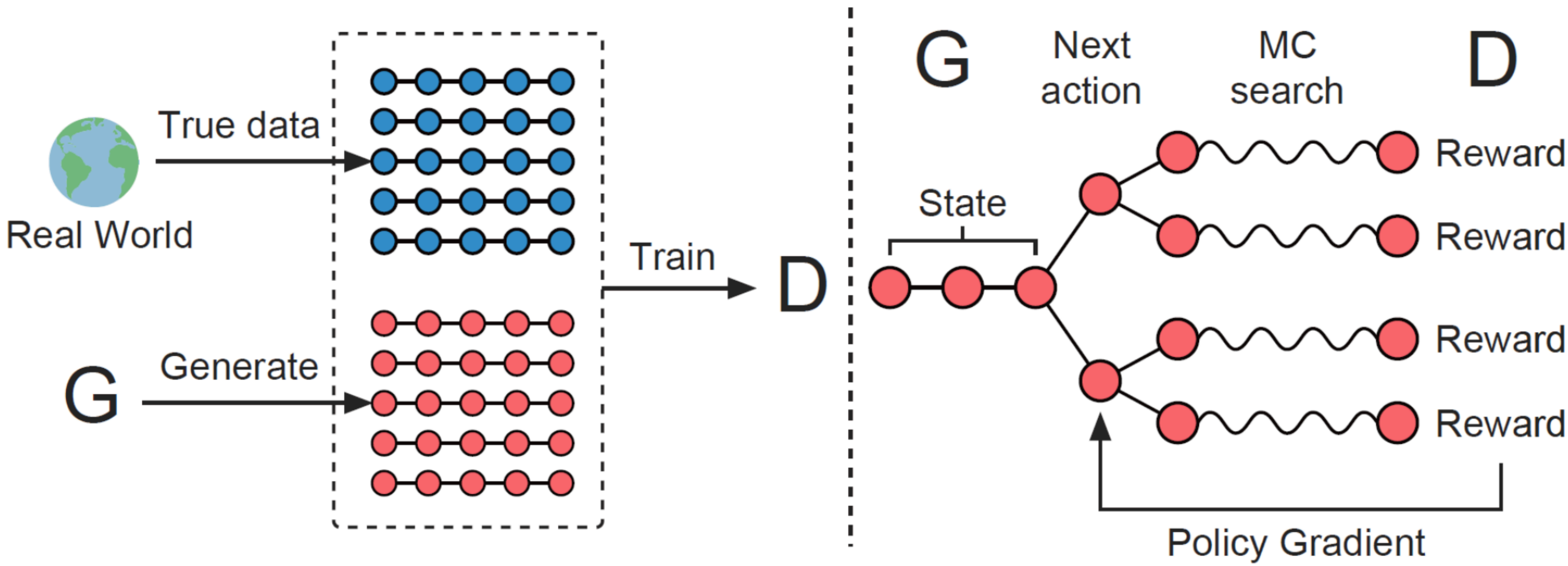
Table 2: Performance of KerBS on Transformer.

Tasks	Metrics	Transformer	Transformer + MoS [Yang et al., 2018]	Transformer + KerBS
MT	BLEU-4	29.61	28.54	**30.90**
Dialog	BLEU-1	10.61	9.81	**10.90**

Case Study 2

- Greedy Embedding Matching
  - Kernelized Bayesian Softmax
- RL for Generation
  - SeqGAN
- Generation by Sampling
  - Constrained Sentence Generation with CGMH
  - Generating Adversarial Examples for Natural Languages

SeqGAN



Directly applying RL to use Discriminator outputs as reward for updating Generator.

BackPropagation Fails

- Sentence is discrete, BP fails in such case
  - RL
  - Gumbel Softmax

Variance of gradient is very large!
Hard for training :(

RL for Text Generation

Strategies to deal with discontinuity | GAN models

GANs for text generation

- **Policy Gradient**
  - **SeqGAN**: First GAN on discrete sentence space.
  - **RankGAN**: Use rank information to mitigate gradient vanishing.
  - **LeakGAN**: Use feature extracted by D to guide G.
- **Gumbel Softmax**
  - **GumbelGAN**: Use Gumbel-trick to handle discontinuity.
  - **TextGAN**: Use feature matching for training.
  - **RelGAN**: Build stronger D and G. The first practical Gumbel GAN.
  - **LATEXT-GAN**: Combines Gumbel GAN and AAE
- **AAE (Adversarial Autoencoder)**
  - **ARAE**: Perform GAN on embedding space.
  - **LATEXT-GAN** : Combines Gumbel GAN and AAE

MLE Outperforms different GAN Variants

Model	NLL_{oracle}
SeqGAN (Yu et al., 2017)	8.74
RankGAN (Lin et al., 2017)	8.25
LeakGAN (Guo et al., 2017)	7.04
IRL (Shi et al., 2018)	6.91
MLE ($\alpha = 1.0$)	9.40
MLE ($\alpha = 0.4$)	5.50
MLE ($\alpha = 0.001$)	**4.58**

Table 2: NLL_{oracle} measured on the synthetic task *(lower is better)*. All results are taken from their respective papers. An MLE-trained model with reduced temperature easily improves upon these GAN variants, producing the highest quality sample.

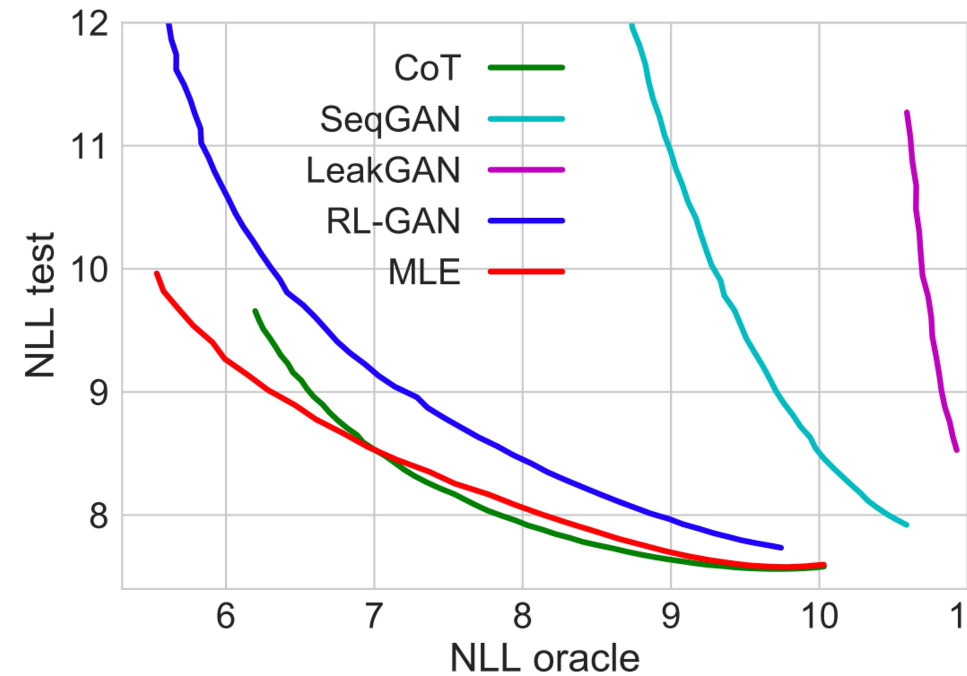


Figure 3: Effect of temperature tuning on the global metrics *(lower is better for both metrics)* for the synthetic task. The GAN cross-validated on quality only lies outside the figure because of severe mode collapse.

Caccia M, Caccia L, Fedus W, et al. Language gans falling short[J]. arXiv preprint arXiv:1811.02549, 2018.

Case Study 3

- Greedy Embedding Matching
  - Kernelized Bayesian Softmax
- RL for Generation
  - SeqGAN
- Generation by Sampling
  - Constrained Sentence Generation with CGMH
  - Generating Adversarial Examples for Natural Languages

Advertisement Slogan by Constrained Generation

Keywords from Advertiser

Advertisement Slogan

<u>Rin</u> <u>clothes</u> <u>bright</u>

Sampling in Sentence Space

CGMH performs Metropolis-Hastings sampling directly in sentence space:

Step	Action	Acc/Rej	Sentences
0	[Input]		BMW sports
1	Insert	Accept	BMW sports car
2	Insert	Accept	BMW the sports car
...	...	...	...
6	Insert	Accept	BMW , the sports car of daily life
7	Replace	Accept	BMW , the sports car of future life
8	Insert	Accept	BMW , the sports car of the future life
9	Delete	Reject	BMW , the sports car of the future life
10	Delete	Accept	BMW , the sports car of the future life
11	[Output]		BMW , the sports car of the future

Miao et al., CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling, in AAAI, 2019.

Sampling in Sentence Space

CGMH performs Metropolis-Hastings sampling directly in sentence space:

Step	Action	Acc/Rej	Sentences
0	[Input]		BMW sports

Miao et al., CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling, in AAAI, 2019.

Sampling in Sentence Space

CGMH performs Metropolis-Hastings sampling directly in sentence space:

Step	Action	Acc/Rej	Sentences
0	[Input]		BMW sports
1	Insert	Accept	BMW sports car

Miao et al., CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling, in AAAI, 2019.

Sampling in Sentence Space

CGMH performs Metropolis-Hastings sampling directly in sentence space:

Step	Action	Acc/Rej	Sentences
0	[Input]		BMW sports
1	Insert	Accept	BMW sports car
2	Insert	Accept	BMW the sports car

Miao et al., CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling, in AAAI, 2019.

Sampling in Sentence Space

CGMH performs Metropolis-Hastings sampling directly in sentence space:

Step	Action	Acc/Rej	Sentences
0	[Input]		BMW sports
1	Insert	Accept	BMW sports car
2	Insert	Accept	BMW the sports car
...	...	...	...

Miao et al., CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling, in AAAI, 2019.

Sampling in Sentence Space

CGMH performs Metropolis-Hastings sampling directly in sentence space:

Step	Action	Acc/Rej	Sentences
0	[Input]		BMW sports
1	Insert	Accept	BMW sports car
2	Insert	Accept	BMW the sports car
…	…	…	…
6	Insert	Accept	BMW , the sports car of daily life

Miao et al., CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling, in AAAI, 2019.

Sampling in Sentence Space

CGMH performs Metropolis-Hastings sampling directly in sentence space:

Step	Action	Acc/Rej	Sentences
0	[Input]		BMW sports
1	Insert	Accept	BMW sports car
2	Insert	Accept	BMW the sports car
…	…	…	…
6	Insert	Accept	BMW , the sports car of daily life
7	Replace	Accept	BMW , the sports car of future life

Miao et al., CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling, in AAAI, 2019.

Sampling in Sentence Space

CGMH performs Metropolis-Hastings sampling directly in sentence space:

Step	Action	Acc/Rej	Sentences
0	[Input]		BMW sports
1	Insert	Accept	BMW sports car
2	Insert	Accept	BMW the sports car
…	…	…	…
6	Insert	Accept	BMW , the sports car of daily life
7	Replace	Accept	BMW , the sports car of future life
8	Insert	Accept	BMW , the sports car of the future life

Miao et al., CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling, in AAAI, 2019.

Sampling in Sentence Space

CGMH performs Metropolis-Hastings sampling directly in sentence space:

Step	Action	Acc/Rej	Sentences
0	[Input]		BMW sports
1	Insert	Accept	BMW sports car
2	Insert	Accept	BMW the sports car
…	…	…	…
6	Insert	Accept	BMW , the sports car of daily life
7	Replace	Accept	BMW , the sports car of future life
8	Insert	Accept	BMW , the sports car of the future life
9	Delete	Reject	BMW , the sports car of the future life

Miao et al., CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling, in AAAI, 2019.

Sampling in Sentence Space

CGMH performs Metropolis-Hastings sampling directly in sentence space:

Step	Action	Acc/Rej	Sentences
0	[Input]		BMW sports
1	Insert	Accept	BMW sports car
2	Insert	Accept	BMW the sports car
…	…	…	…
6	Insert	Accept	BMW , the sports car of daily life
7	Replace	Accept	BMW , the sports car of future life
8	Insert	Accept	BMW , the sports car of the future life
9	Delete	Reject	BMW , the sports car of the future life
10	Delete	Accept	BMW , the sports car of the future life

Miao et al., CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling, in AAAI, 2019.

Sampling in Sentence Space

CGMH performs Metropolis-Hastings sampling directly in sentence space:

Step	Action	Acc/Rej	Sentences
0	[Input]		BMW sports
1	Insert	Accept	BMW sports car
2	Insert	Accept	BMW the sports car
…	…	…	…
6	Insert	Accept	BMW , the sports car of daily life
7	Replace	Accept	BMW , the sports car of future life
8	Insert	Accept	BMW , the sports car of the future life
9	Delete	Reject	BMW , the sports car of the future life
10	Delete	Accept	BMW , the sports car of the future life
11	[Output]		BMW , the sports car of the future

Miao et al., CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling, in AAAI, 2019.

Cases of Keyword to Sentences

Keyword(s)	**Generated Sentences**
friends	My good **friends** were in danger .
project	The first **project** of the scheme .
have, trip	But many people **have** never made the **trip** .
lottery, scholarships	But the **lottery** has provided **scholarships** .
decision, build, home	The **decision** is to **build** a new **home** .
attempt, copy, painting, denounced	The first **attempt** to **copy** the **painting** was **denounced** .

Paraphrase Generation

Model	BLEU-ref	BLEU-ori	NLL
Origin Sentence	30.49	100.00	7.73
VAE-SVG (100k)	22.50	-	-
VAE-SVG-eq (100k)	**22.90**	-	-
VAE-SVG (50k)	17.10	-	-
VAE-SVG-eq (50k)	17.40	-	-
Seq2seq (100k)	22.79	33.83	6.37
Seq2seq (50k)	20.18	27.59	**6.71**
Seq2seq (20k)	16.77	22.44	6.67
VAE (unsupervised)	9.25	27.23	7.74
CGMH *w/o matching*	18.85	50.28	7.52
w/ KW	20.17	53.15	7.57
w/ KW + WVA	20.41	53.64	7.57
w/ KW + WVM	20.89	54.96	7.46
w/ KW + ST	20.70	54.50	7.78

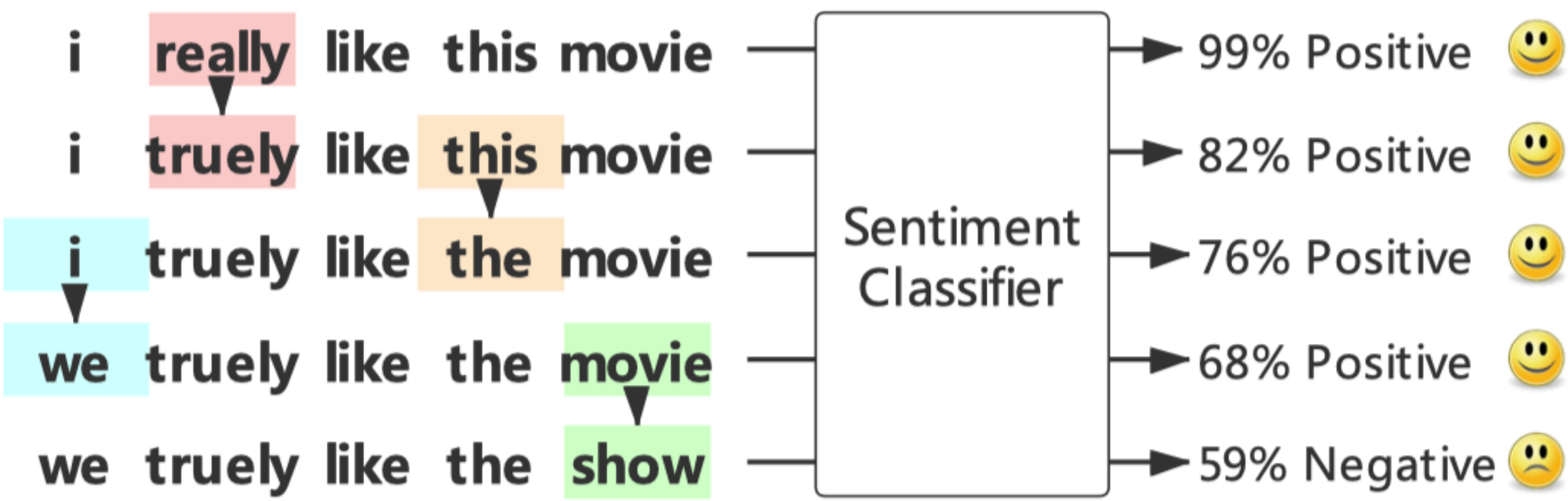
Type	Examples
Ori	what 's the best plan to lose weight
Ref	what is a good diet to lose weight
Gen	what 's the best way to slim down quickly
Ori	how should i control my emotion
Ref	how do i control anger and impulsive emotions
Gen	how do i control my anger
Ori	why do my dogs love to eat tuna fish
Ref	why do my dogs love eating tuna fish
Gen	why do some dogs like to eat raw tuna and raw fish

Adversarial Example for Text

Generating adversarial example for text is hard! Because the text space is discrete, which is non-trivial to apply adversarial gradients!

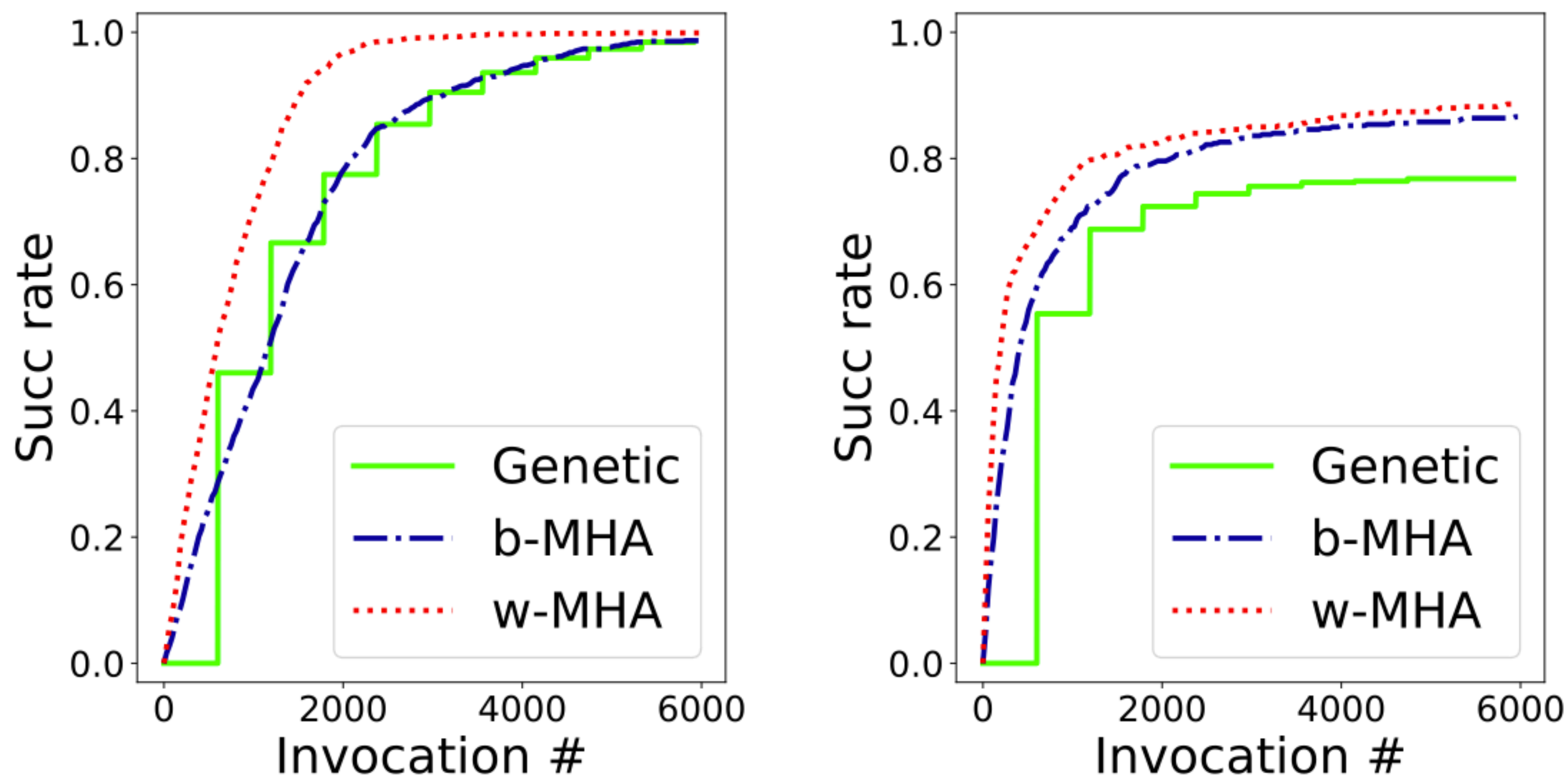
Zhang et al., Generating Fluent Adversarial Examples for Natural Languages, in ACL, 2019, short paper.

CGMH for Generating Fluent Adversarial Examples



Huangzhao Zhang, Hao Zhou, Ning Miao and Lei Li. Generating Fluent Adversarial Examples for Natural Languages, in ACL, 2019..

CGMH for Generating Fluent Adversarial Examples



(a) IMDB (b) SNLI

Huangzhao Zhang, Hao Zhou, Ning Miao and Lei Li. Generating Fluent Adversarial Examples for Natural Languages, in ACL, 2019..

CGMH for Generating Fluent Adversarial Examples

Task	Approach	Succ(%)	Invok#	PPL	α(%)
IMDB	Genetic	98.7	1427.5	421.1	–
	b-MHA	98.7	1372.1	385.6	17.9
	w-MHA	**99.9**	**748.2**	**375.3**	34.4
SNLI	Genetic	76.8	971.9	834.1	–
	b-MHA	86.6	681.7	358.8	9.7
	w-MHA	**88.6**	**525.0**	**332.4**	13.3

Huangzhao Zhang, Hao Zhou, Ning Miao and Lei Li. Generating Fluent Adversarial Examples for Natural Languages, in ACL, 2019..

Conclusion of the Tutorial

Conclusion of the Tutorial

- Neural networks are good
- Natural language is discrete (Input, latent, output spaces)
  - Representation learning
  - Non-differentiability
  - Exponential search space

Thank You