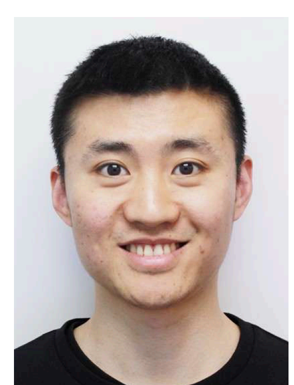
Discreteness in Neural Natural Language Processing

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

1. Why do we need this tutorial?
2. What can you learn from this tutorial?

Why this tutorial?

- Deep learning has almost dominated NLP these years.

Why this tutorial?

- Deep learning has almost dominated NLP these years
- Different from speech and images, natural language units (word, sentence, paragraph, etc.) are **discrete**
- May cause problems in neural NLP

What could we learn from this tutorial?

- We will give examples of discreteness in neural NLP, including input, latent and output spaces.

What could we learn from this tutorial?

- We will give examples of discreteness in neural NLP, including input, latent and output spaces.
- We will introduce advanced techniques to address the discreteness problem.

What could we learn from this tutorial?

- We will give examples of discreteness in neural NLP, including input, latent and output spaces.
- We will introduce advanced techniques to address the discreteness problem.
- Cases will be finally studied to show how we can use these techniques to solve practical problems.

Outline

- Tutorial Introduction
  - Ubiquitous discreteness in natural language processing
  - Challenges of dealing with discreteness in neural NLP
- Discrete Input Space
  - Mapping discrete symbols to distributed representation
- Discrete Latent Space
  - Addressing the non-differential problem in back-propagation of discrete variables
- Discrete Output Space
  - Learning and inference in exponential hypothesis space
  - Training without maximum likelihood estimation
- Take Away

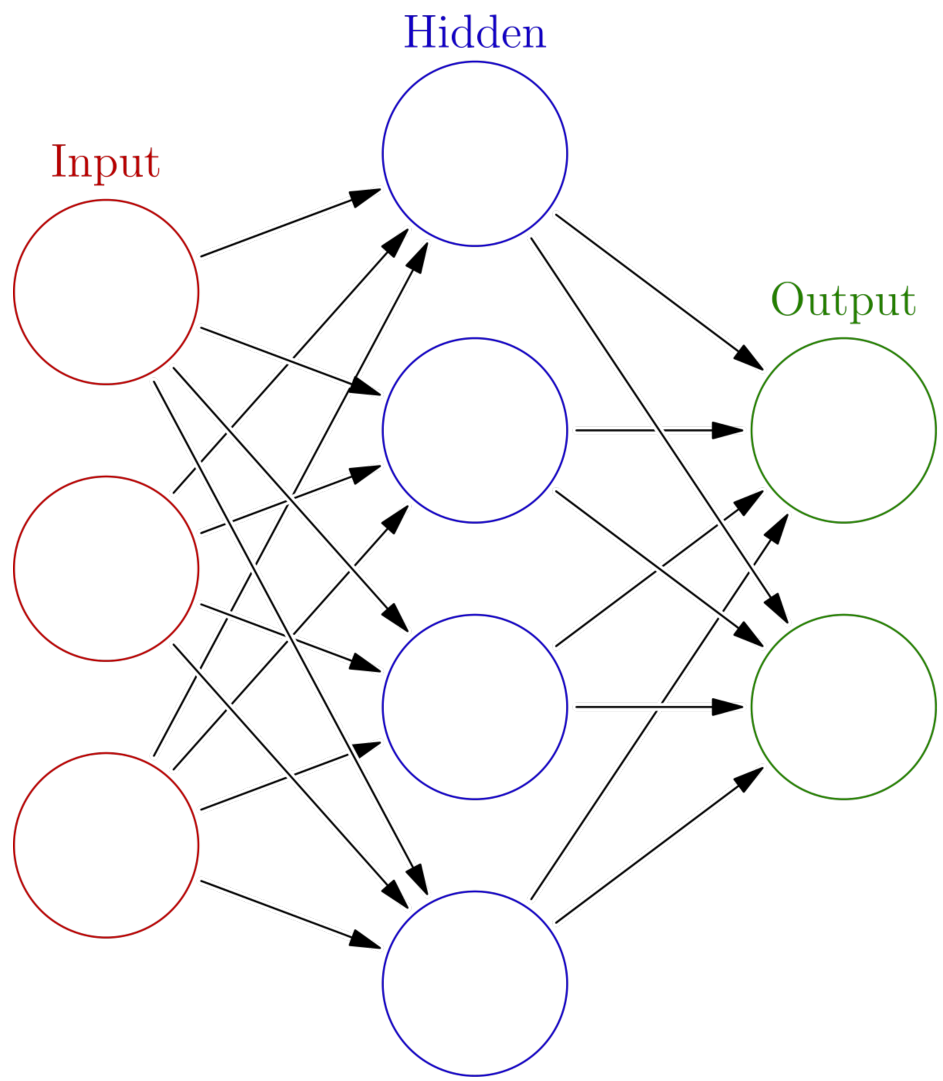
Part I: Introduction

Roadmap

- The role of distributed representation in deep learning
- Ubiquitous discreteness in natural language processing
- Challenges of dealing with discreteness in deep learning-based NLP
  - Continuous/distributed representation
  - Non-differentiability
  - Exponential search space

Neural Networks

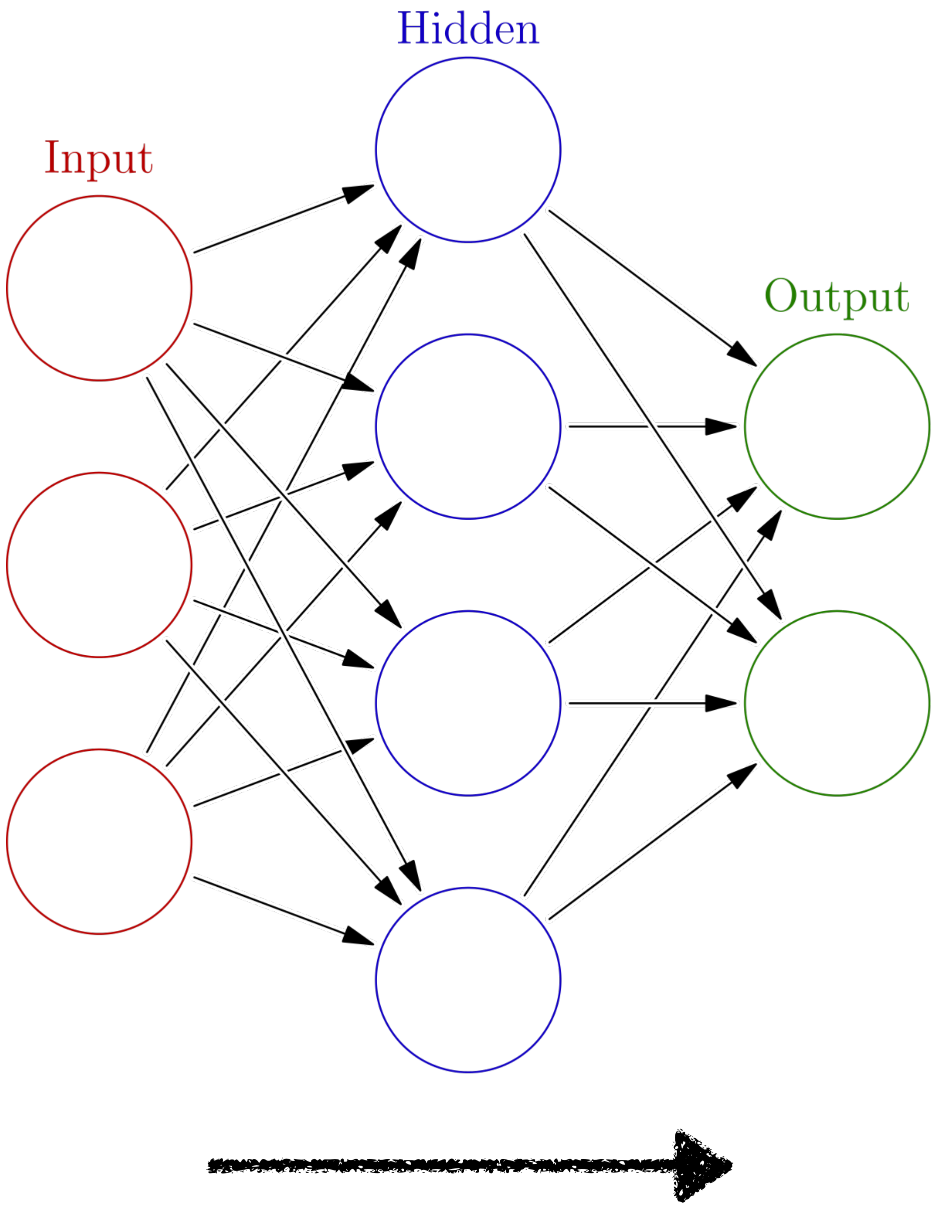
Input layers take data signals



Output layers give task oriented predictions

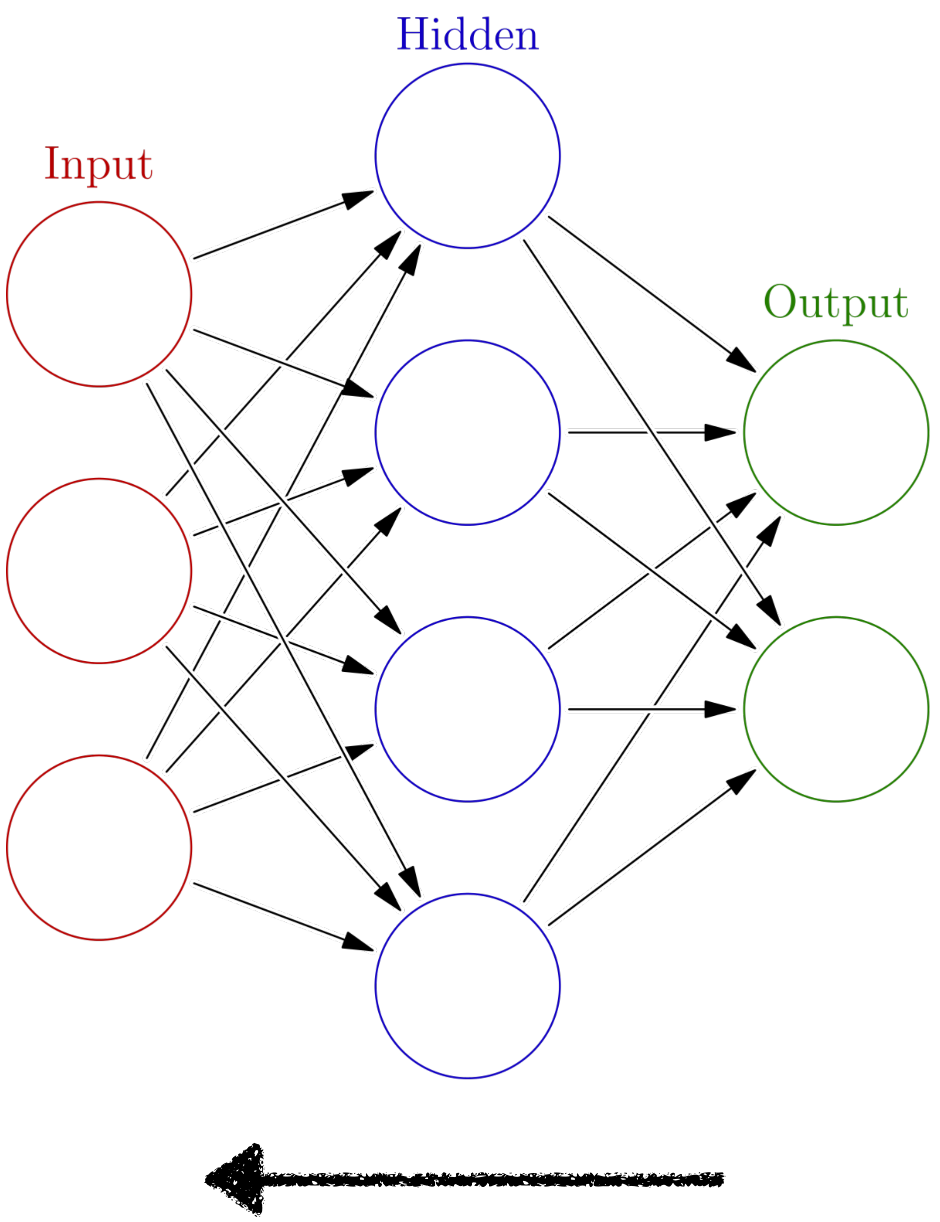
Hidden layers perform non-linear transformation

Forward



Obtaining predictions by forward propagation

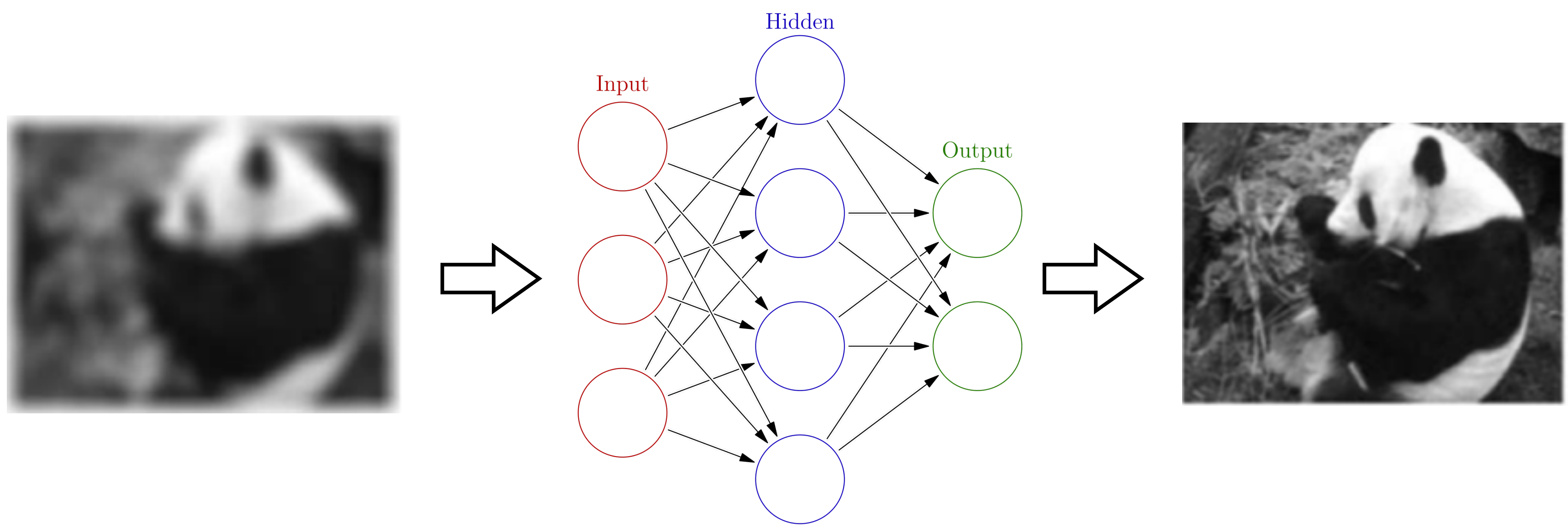
Backward



Updating parameters by backward propagation

DL is suitable for continuous variables

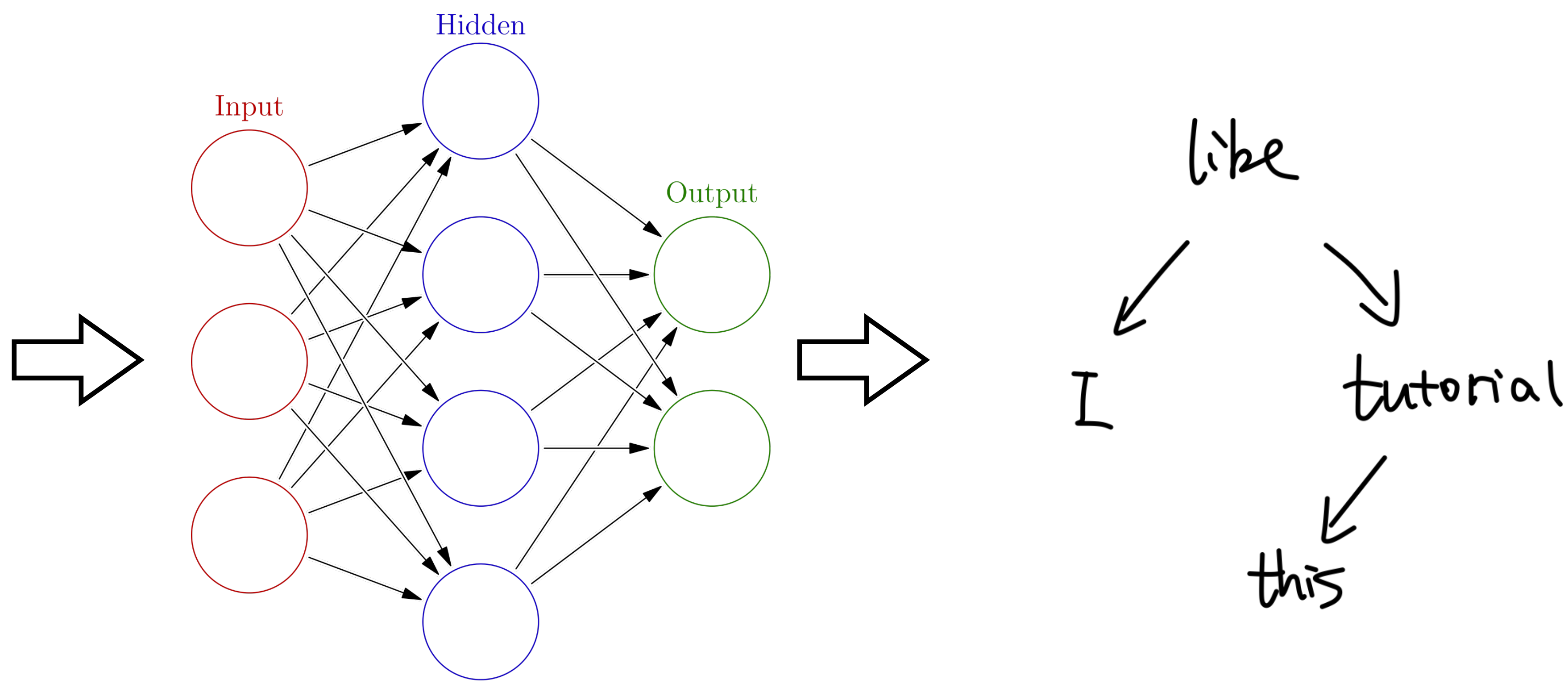
- For speech and images, the input and output spaces are always continuous, which are straightforward for forward and backward propagations in neural networks.



Ubiquitous Discreteness in NLP

- Natural Language is discrete
  - input space, latent space, output space

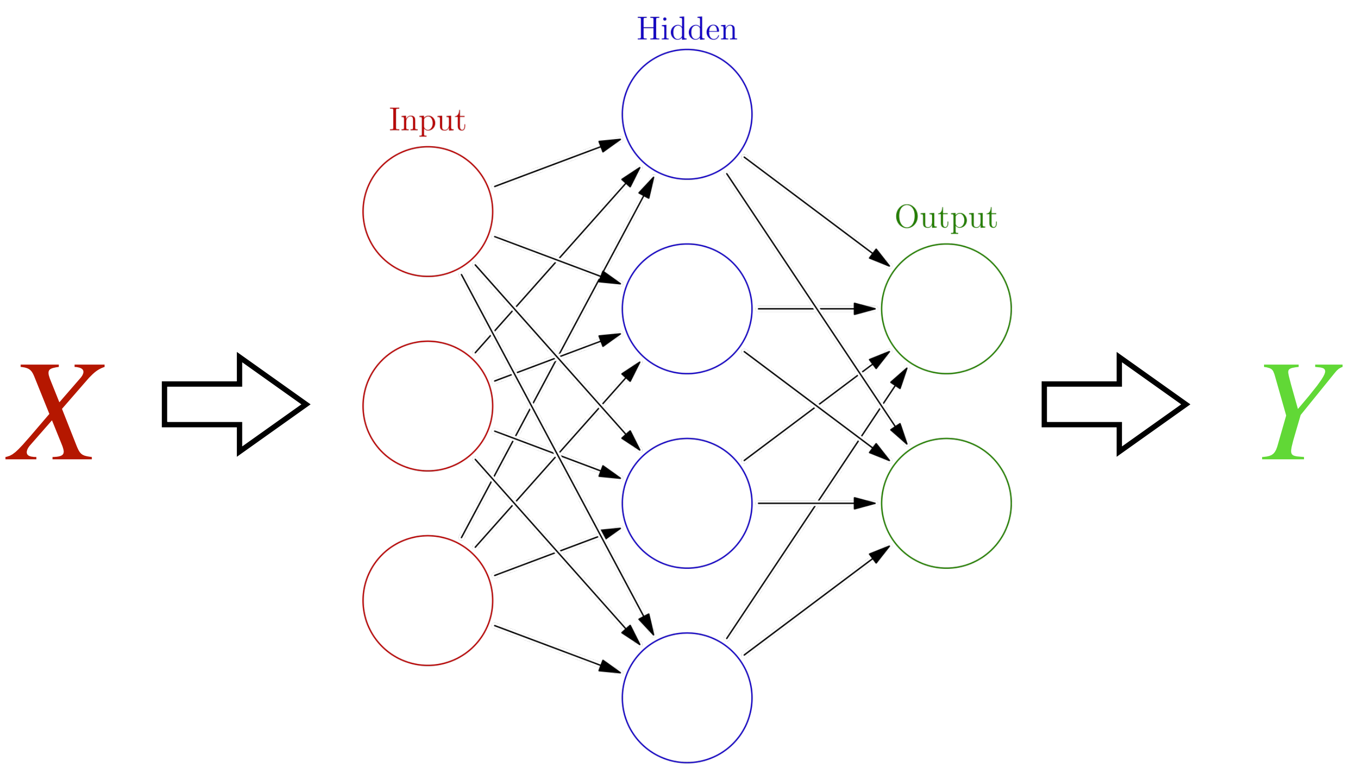
I like this tutorial



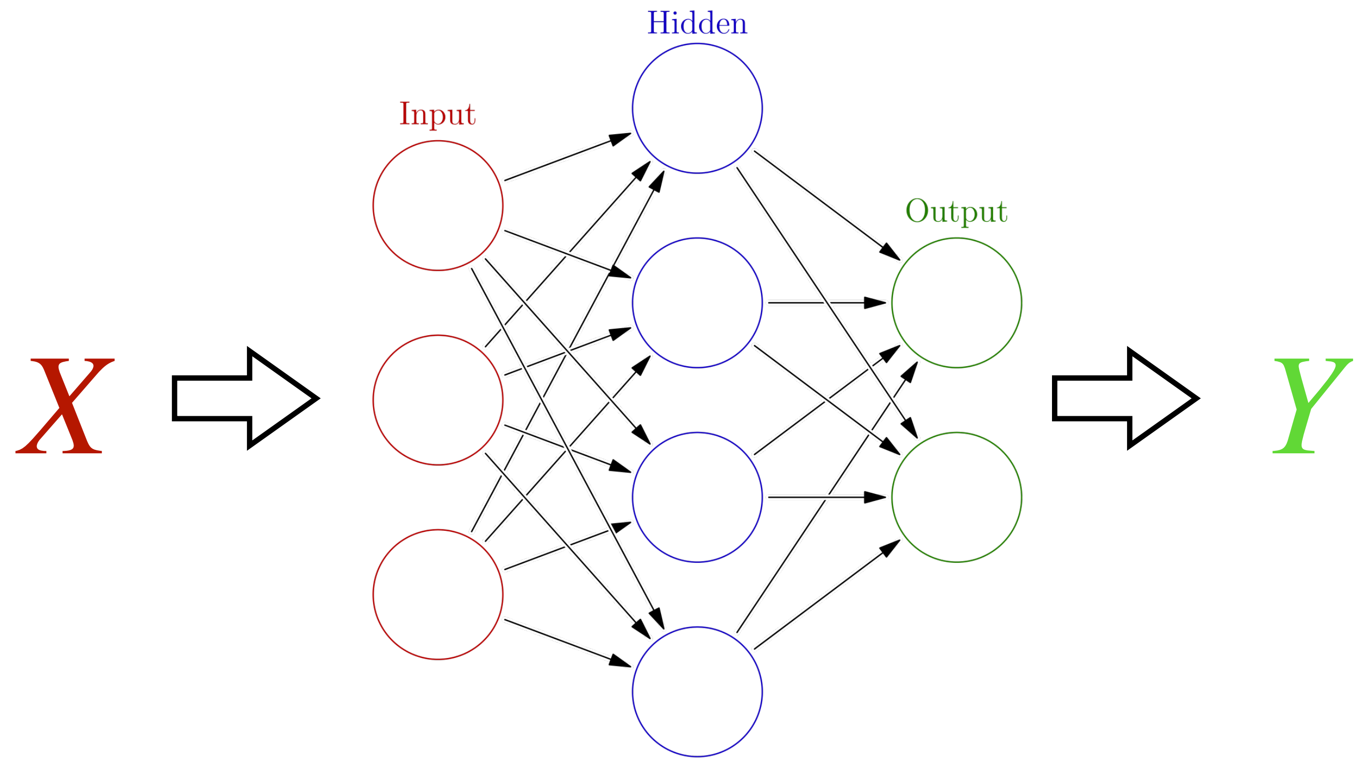
sentence

parse tree

From Input to Output

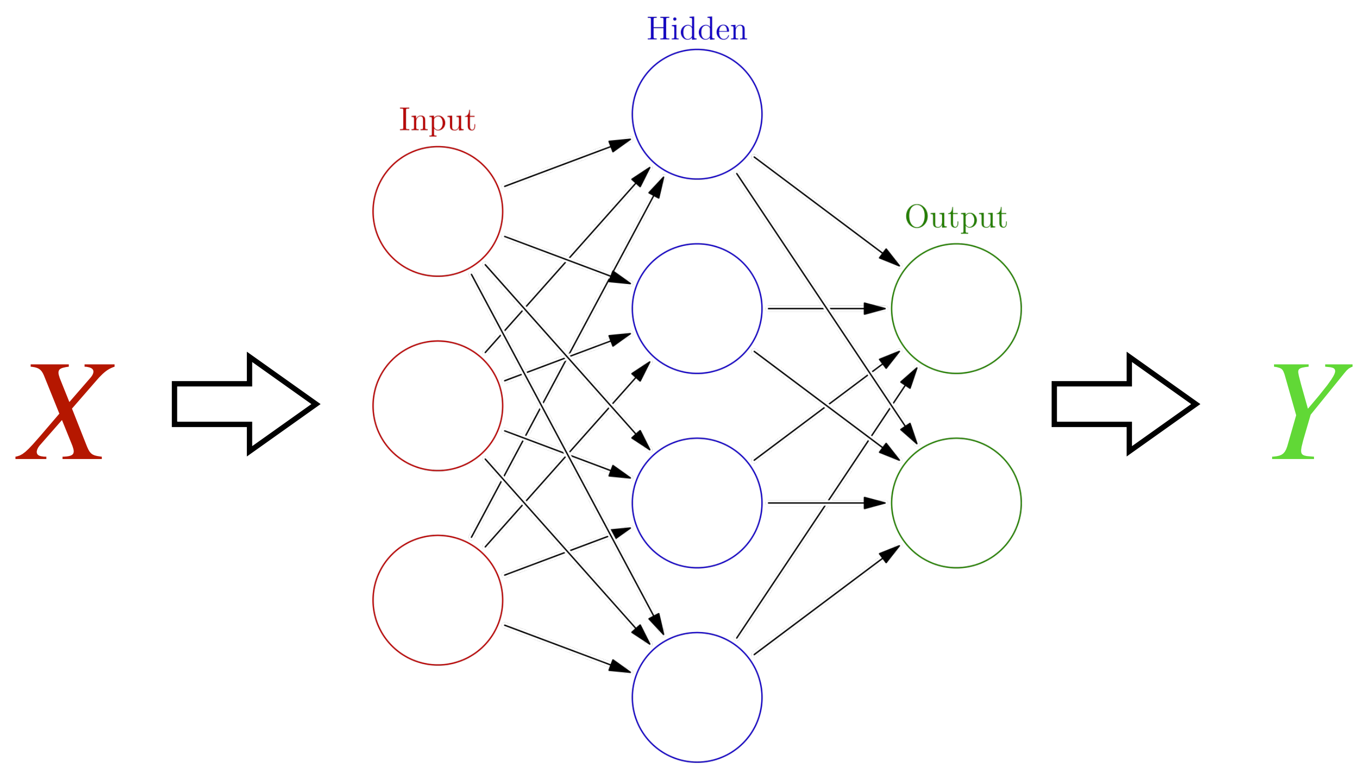


From Input to Output



$$X \xrightarrow{\text{neural networks}} Y$$

From Input to Output



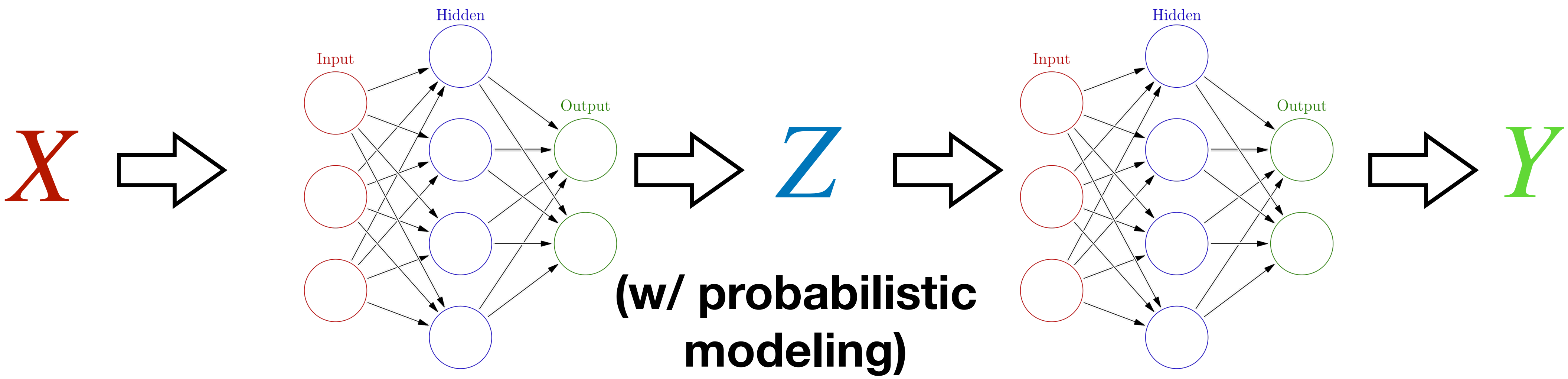
Word
Sentence
Tree
Graph
...

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

Word
Sentence
Tree
Graph
...

Input and Output Spaces may be discrete in NLP !

$P(Y|X)$ with Latent Variable Z



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

$$P(Y|X) = \sum_{Z} P(Z|X)P(Y|Z,X)$$

$P(Y|X)$ with Latent Variable Z

Topic
Sentence
Tree
Graph
...

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

$$P(Y|X) = \sum_{Z} P(Z|X)P(Y|Z,X)$$

Latent space may also be discrete in NLP !

Non-Trivial to Deal with Discreteness in Neural Networks

Input, latent and output of NLP tasks are oftentimes discrete symbols or structures.

Challenges of Discreteness

- Input Space
  - How to get good distributed representation?

Challenges of Discreteness

- Input Space
  - How to get good distributed representation?
- Latent Space
  - Difficult for Backpropagation

Challenges of Discreteness

- Input Space
  - How to get good distributed representation?
- Latent Space
  - Hard for Backpropagation
- Output Space
  - Exponential Search Space, hard for learning and inference.
  - Besides MLE, hard for training.

Challenges of Discreteness

- Input Space
  - How to get good distributed representation?
- Latent Space
  - Hard for Backpropagation
- Output Space
  - Exponential Search Space, hard for learning and inference.
  - Besides MLE, hard for training.

More challenges will be introduced in following parts.

What's Next?

In following parts, we will introduce techniques to alleviate above problems for **input**, **latent** and **output** spaces, respectively.

Outline

- Tutorial Introduction
  - Ubiquitous discreteness in natural language processing
  - Challenges of dealing with discreteness in neural NLP
- Discrete Input Space
  - Mapping discrete symbols to distributed representation
- Discrete Latent Space
  - Addressing the non-differential problems in BP for discrete variables
- Discrete Output Space
  - Learning and inference in exponential hypothesis space
  - Training without maximum likelihood estimation
- Take Away

Part II: Discrete Input Space

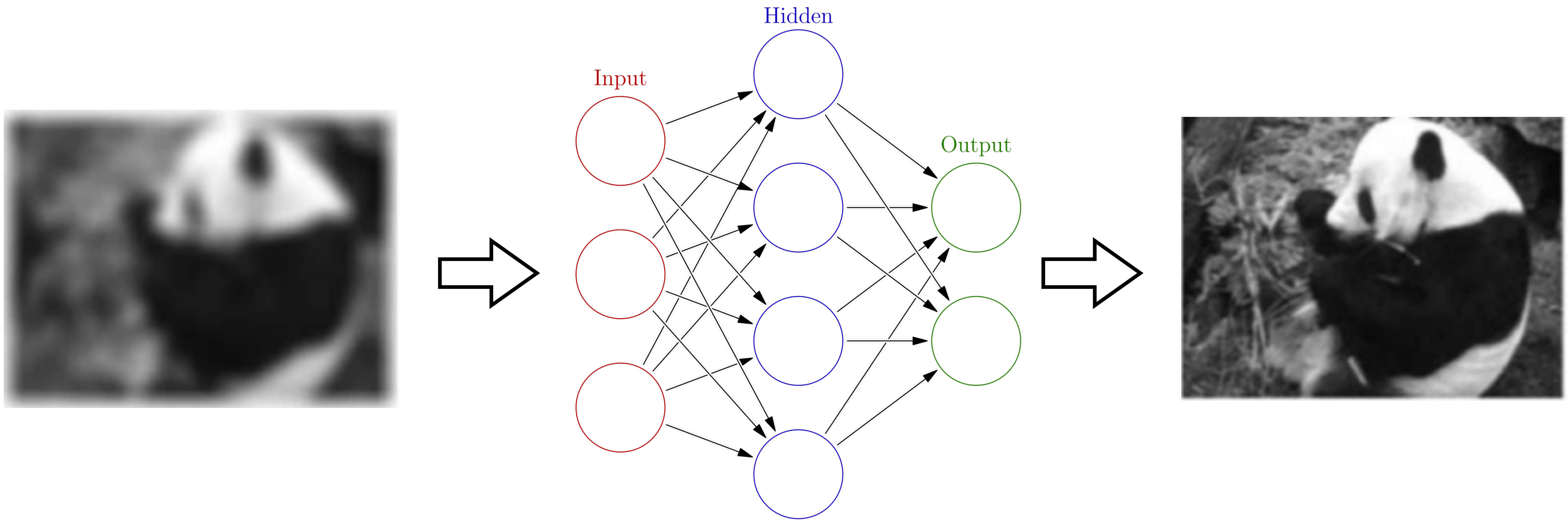
Outline

- Tutorial Introduction
  - Ubiquitous discreteness in natural language processing
  - Challenges of dealing with discreteness in neural NLP
- Discrete Input Space
  - Mapping discrete symbols to distributed representation
- Discrete Latent Space
  - Addressing the non-differential problems in BP for discrete variables
- Discrete Output Space
  - Learning and inference in exponential hypothesis space
  - Training without maximum likelihood estimation
- Take Away

Roadmap

- Examples of discrete input space
  - Sentence, graph, tree, relation, etc.
- Embedding discrete input as distributed vectors
  - From one-hot to distributed representations
  - From context-independent to context dependent representations
- Incorporating discrete structures into neural architectures.

Image as Inputs



Images are continuous signals which can be fed into neural networks directly as distributed representations.

How about Text ?

Word
Sentence
Tree
Graph
Relation
…

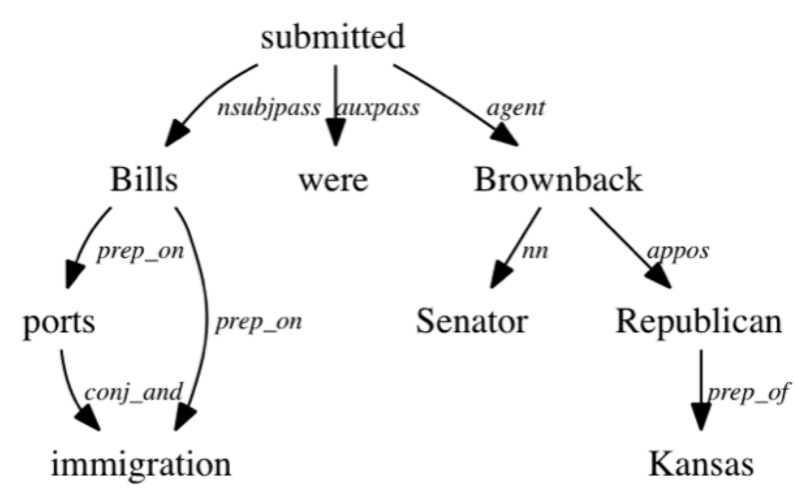
$$X \xrightarrow{\text{neural networks}} Y$$

Dealing with Di

- For example, for sentence classification task, we can input the **sentences**, **syntax tree** of the sentence or even extra **knowledge graph** into neural networks to get final predictions.

- Inputs can be **discrete** symbols/structures, which can not be fed to neural networks directly.

"I am visiting Hong Kong"



Dealing with Discrete Symbols

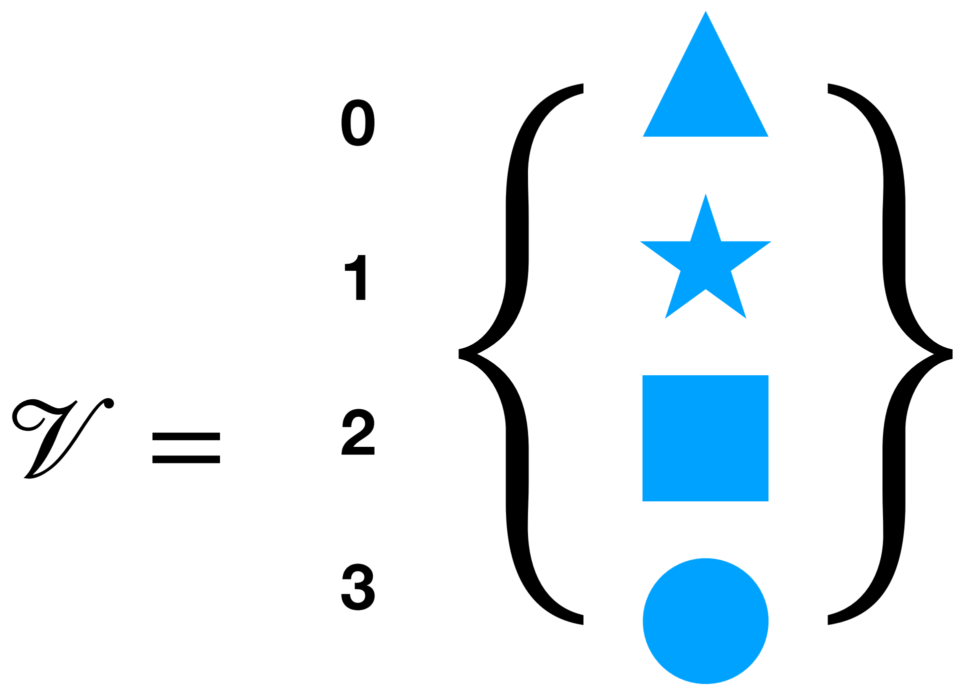
- The raw input of language

"I am visiting Hong Kong"

- Problem: **Words are discrete tokens**!

ID/One-Hot Rerepresentation

Index representation

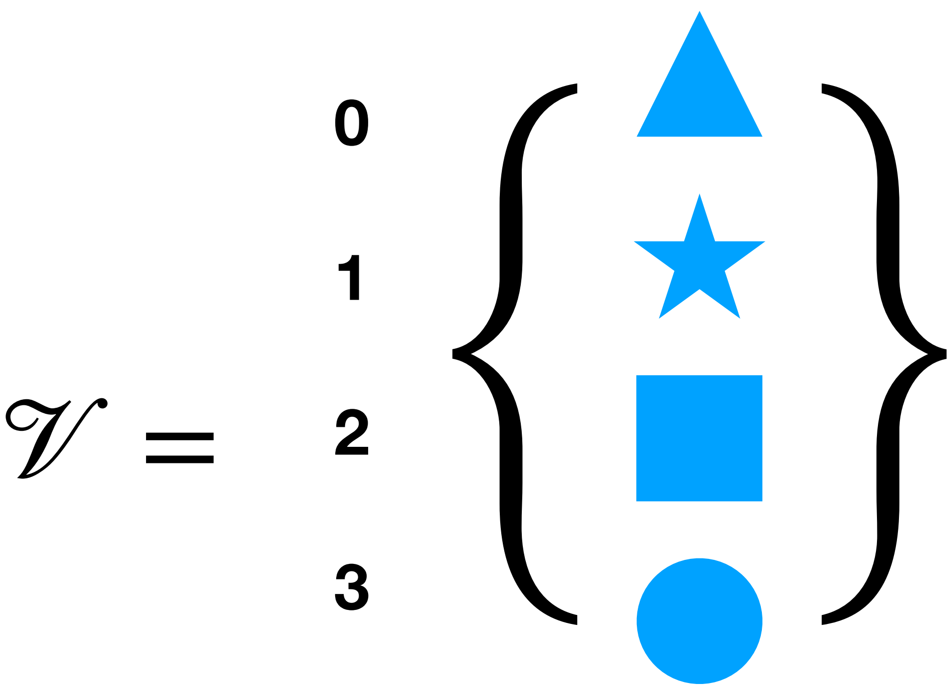


One-hot representation

$$\begin{pmatrix}0\\1\\0\\0\end{pmatrix} \quad \begin{pmatrix}0\\0\\0\\1\end{pmatrix} \quad \begin{pmatrix}0\\0\\1\\0\end{pmatrix} \quad \begin{pmatrix}1\\0\\0\\0\end{pmatrix}$$

One Hot Vectors

Index representation



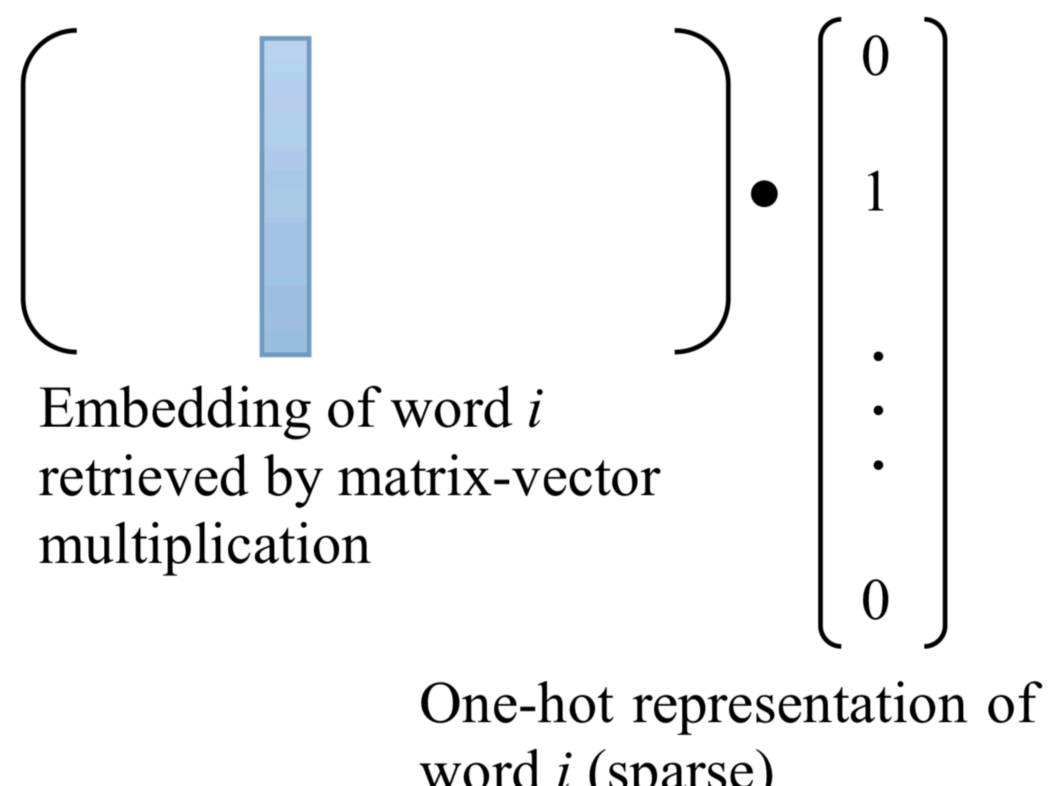
One-hot representation

$$\begin{pmatrix}0\\1\\0\\0\end{pmatrix} \quad \begin{pmatrix}0\\0\\0\\1\end{pmatrix} \quad \begin{pmatrix}0\\0\\1\\0\end{pmatrix} \quad \begin{pmatrix}1\\0\\0\\0\end{pmatrix}$$

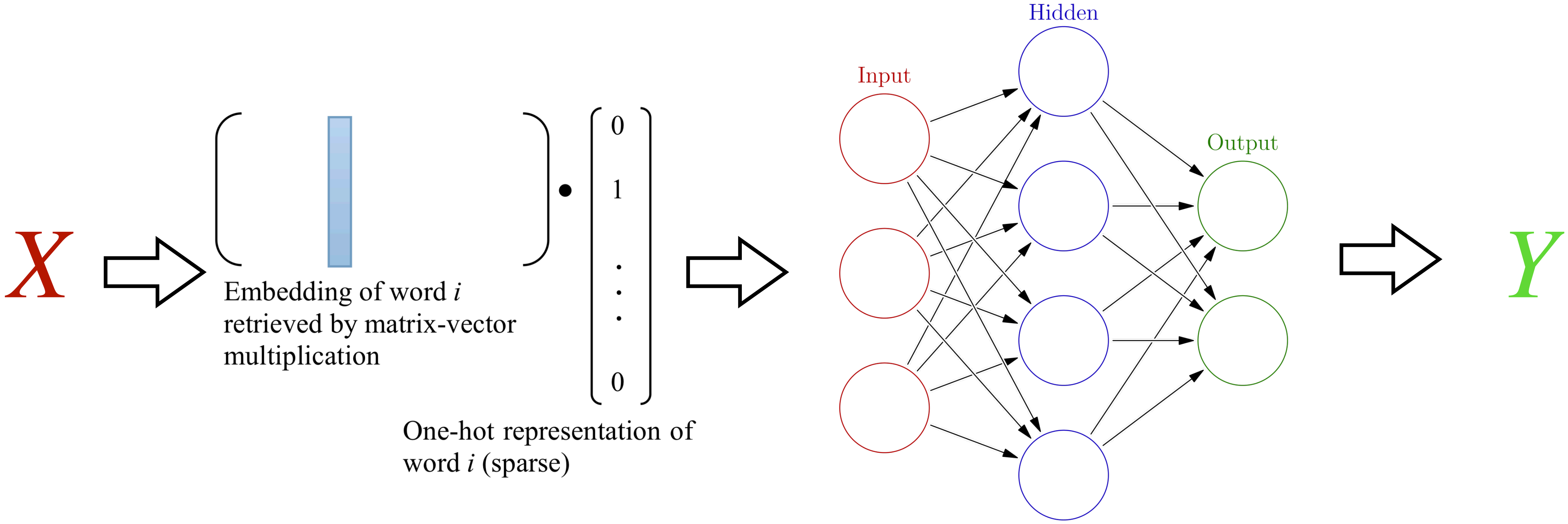
Both representations ignore
the relations between words.

Embeddings

- Map a word to a low-dimensional space
  - Not as low as one-dimensional ID representation
  - Not as high as $|\mathscr{V}|$-dimensional one-hot representation
- Word vector representation (a.k.a., word embeddings)
  - Mapping a word to a vector
  - Equivalent to linear tranformation of one-hot vector

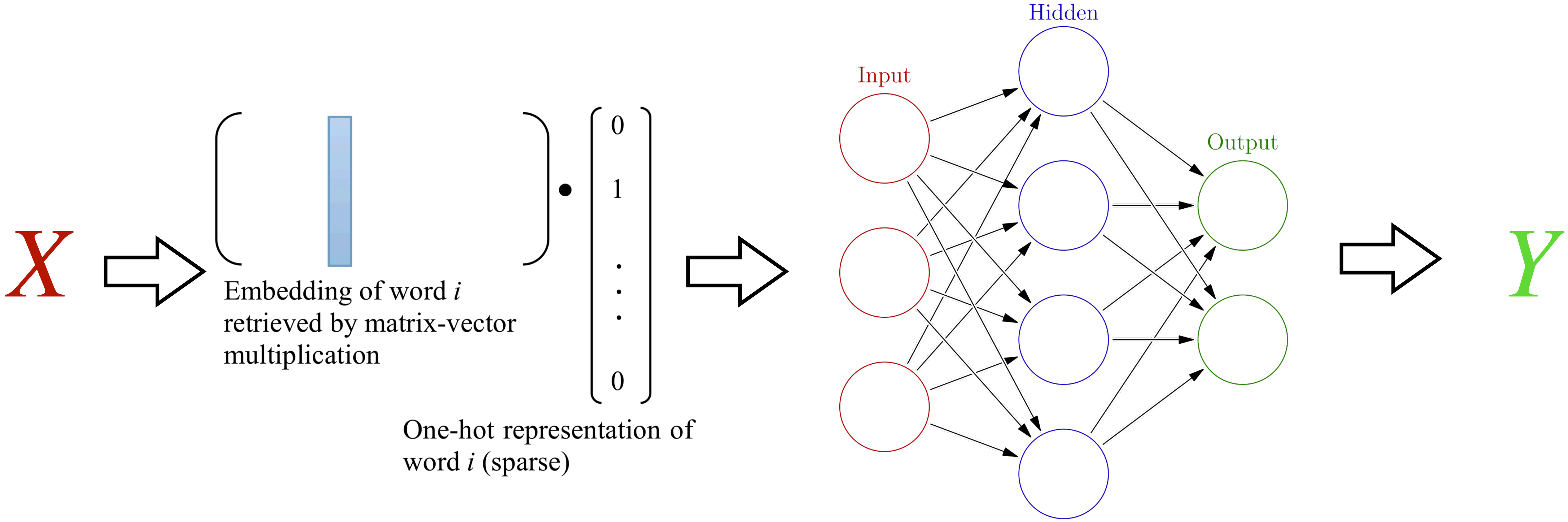


Embedding by Table Lookup



Transforming discrete symbols to distributed representations by table lookup.

Embedding by Table Lookup



The embedding matrix will be updating during the whole neural network training.

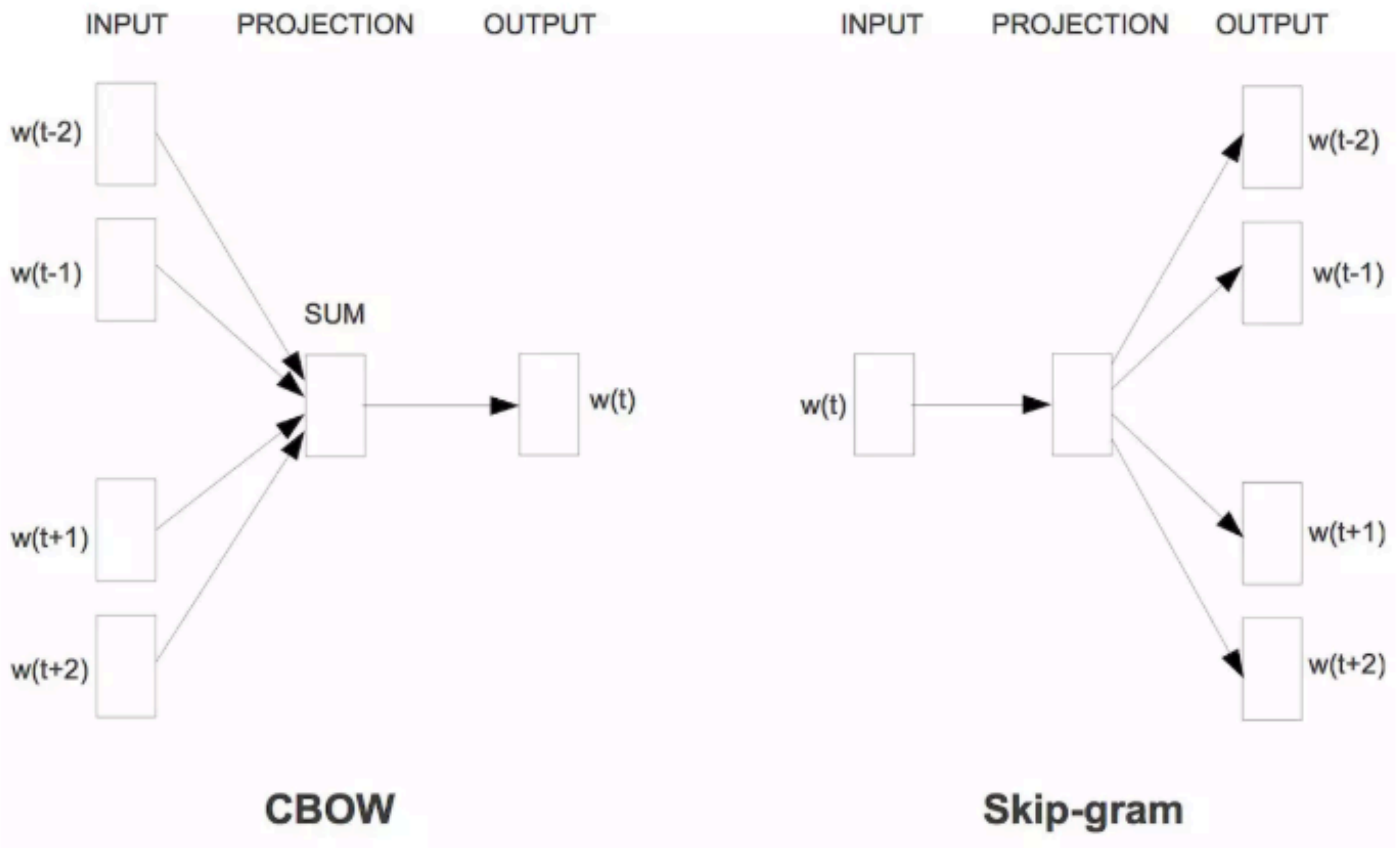
Question

Is the learned embedding in a specific task the best?

Pretraining Embedding

Pretraining word embedding in large-scale corpora, and then fine tuning in downstream tasks.

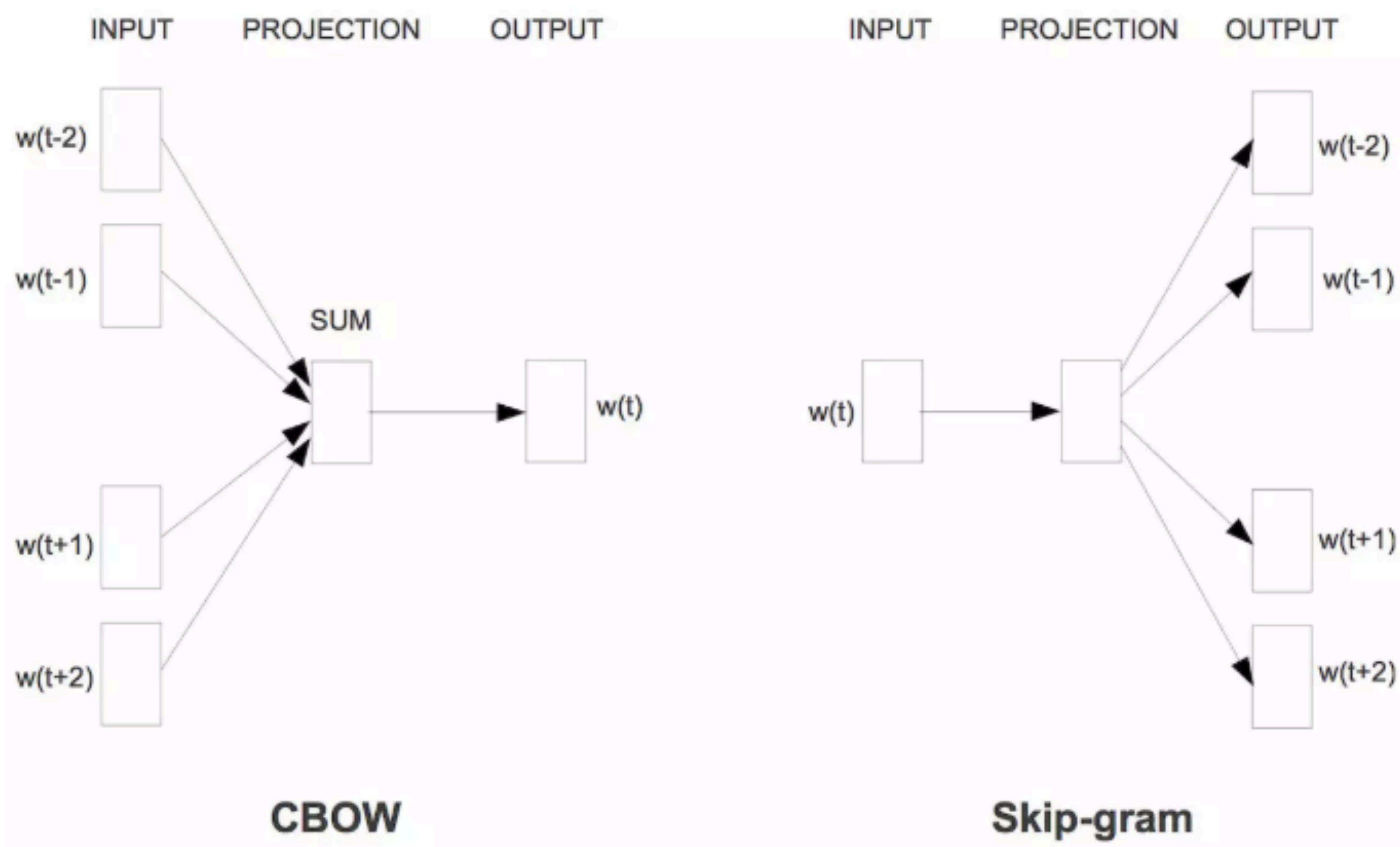
Word2Vec



distributed semantics: similar context leads to similar semantics.

Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality. In NIPS, 2013.

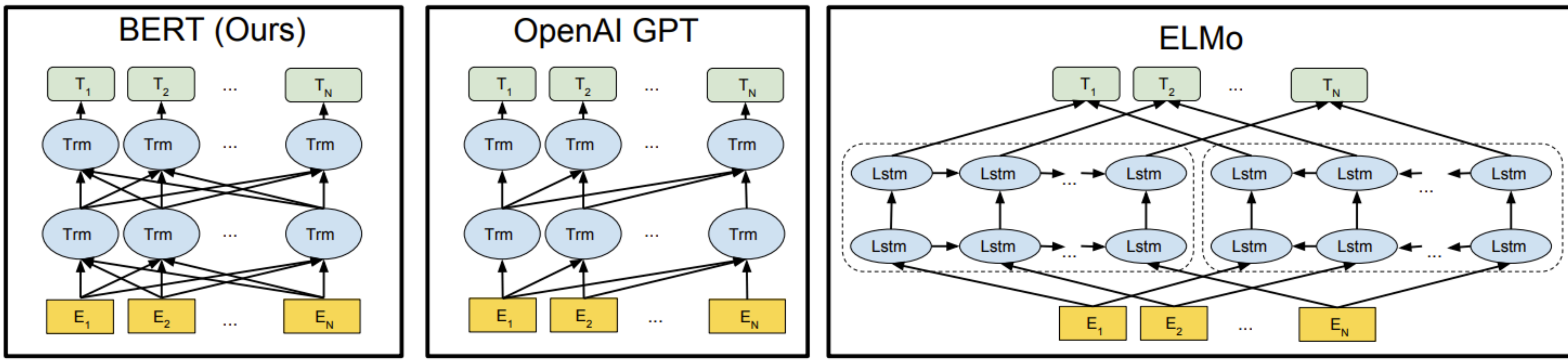
Word2Vec



distributed semantics: similar context leads to similar semantics.

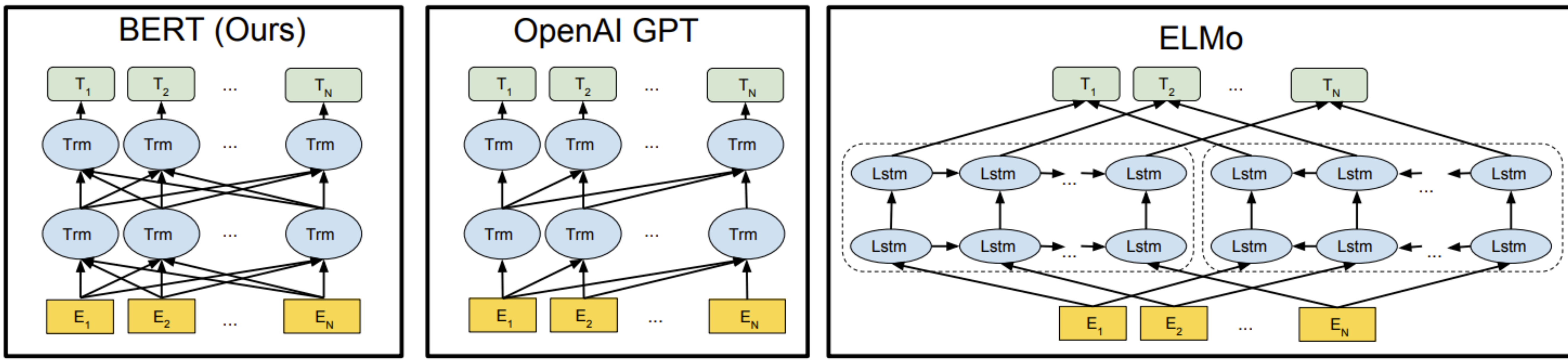
$\boxed{\text{Context Independent!}}$

Context Dependent Representation



Peters M E, Neumann M, Iyyer M, et al. Deep contextualized word representations, in NAACL, 2018.
Radford A, Wu J, Child R, et al., Language models are unsupervised multitask learners. In ICML, 2019.
Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding, in NAACL, 2019.

Context Dependent Representation



Word Embedding are related to its context of observed sentences.

One word has different embeddings

Peters M E, Neumann M, Iyyer M, et al. Deep contextualized word representations, In NAACL, 2018.

Context Dependent VS. Context Independent Embeddings

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

Advanced Representations

- Input
    - rich and rightful feature (context, order, etc.) [Roberta, XLNet, K-Bert]
- Model
    - feature aggregation
    - utilize more context [XLNet]
- Pre-training Objective [MASS, ERINE,SpanBert, T5, etc.]
    - Bert style, Mass style, …

Rich context

- The context of a word can be viewed as the feature for the word in concern.
- The more the valid context/feature, the better it can represent the word in concern.

Rich context

- In pre-training, we need sufficient number of masks to be computationally efficient.
- The mask in Bert is actually the introduced noise feature for representing a real word.
- There is a trade-off between Quantity and Quality.
- Recent papers Roberta, XLNet and K-Bert can be thought of as having enriched context and getting rid of noise.

Roberta

- Change wrt. Bert
  - Removed NSP task, (SpanBert did the same thing)
    - two segment from different document is noise feature to each other
  - All words in a training example came from the same document
  - Dynamic masking with large batch (up to 8K) and some lr increment and momentum change(beta_2: 0.999 —> 0.98)
  - More training data (up to 160G, 10 times compared to data used in Bert)

[Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. "RoBERTa: A Robustly Optimized BERT Pretraining Approach". In: arXiv:1907.11692 (2019)]

XLNet

- Change wrt. Bert
    - two-stream self-attention, latter words will have more words to observe.

$$\mathcal{J}_{\text{BERT}} = \log p(\text{New} \mid \text{is a city}) + \log p(\text{York} \mid \text{is a city}),$$

$$\mathcal{J}_{\text{XLNet}} = \log p(\text{New} \mid \text{is a city}) + \log p(\text{York} \mid \text{New}, \text{is a city}).$$

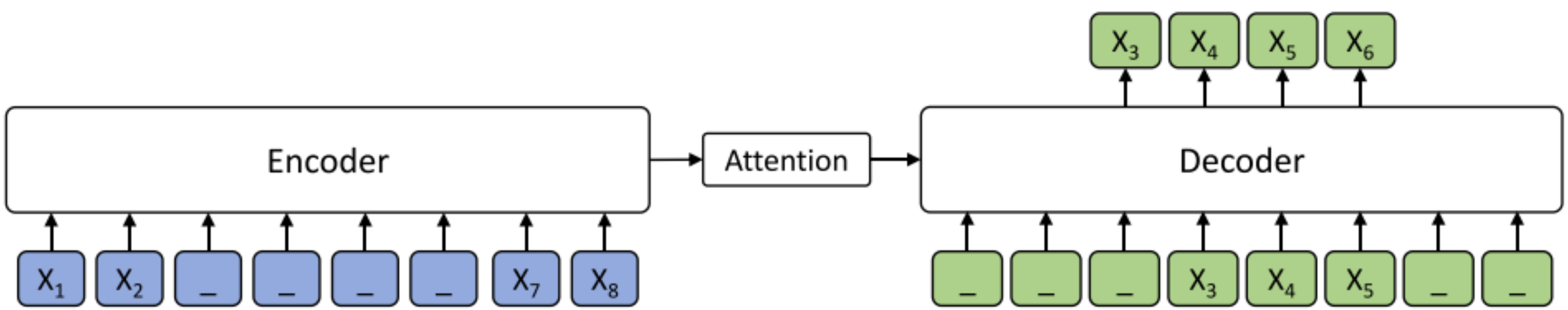
    - Caching mechanism, extend observable sequence up to 512+384, means even more feature.

[Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le. "XLNet: Generalized Autoregressive Pretraining for Language Understanding". In: arXiv:1906.08237 (2019)]

Model

- XLNet tries to utilize more context by devising a two-stream attention mechanism.
- XLNet also devised caching mechanism to utilizing even more context.
- Not much work on feature aggregation.

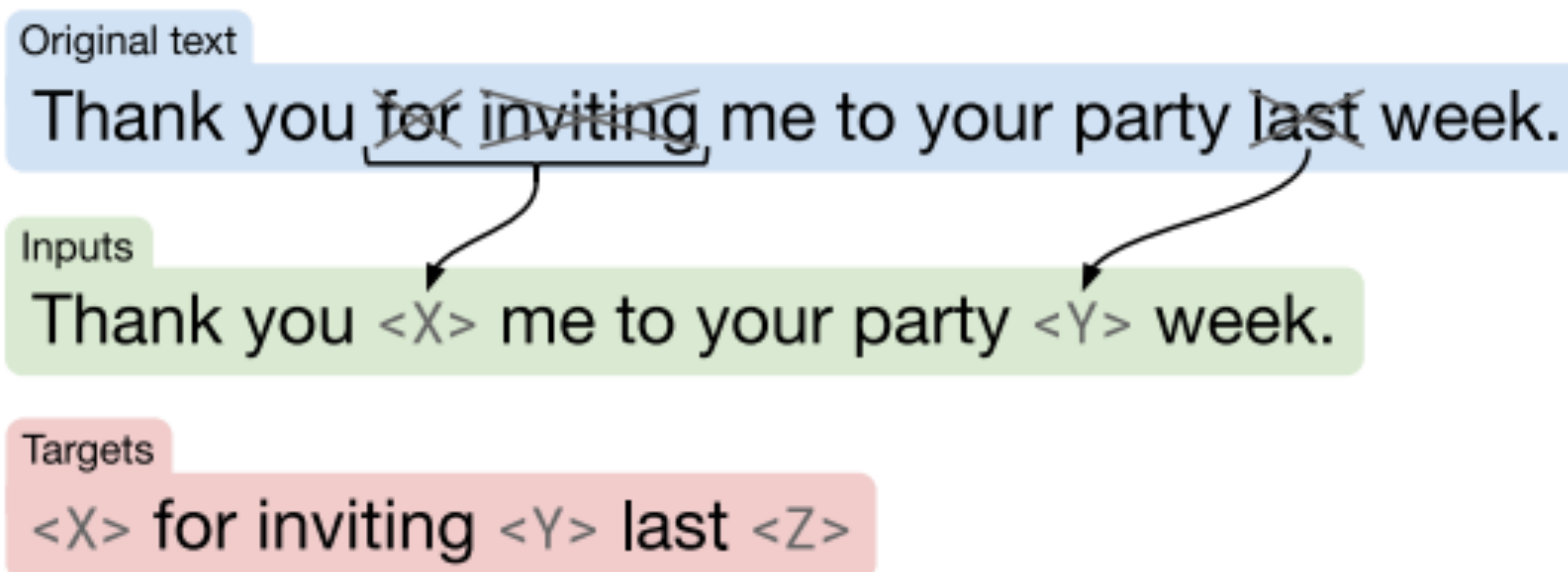
Pre-training objective: MASS

- Mass:
- 

Kaitao S, Xu T, Tao Q, Jianfeng L, and Tie-Yan L. MASS: Masked sequence to sequence pre-training for language generation. arXiv preprint arXiv:1905.02450, 2019.

Pre-training objective: T5

- **T5**: **Text-To-Text** Transfer Transformer

- 

[Raffel, C and Shazeer, Noam and Roberts, Adam and Lee, Katherine and Narang, Sharan and Matena, Michael and Zhou, Yanqi and Li, Wei and Liu, Peter J. arXiv preprint arXiv:1910.10683, 2019.]

Pre-training objective: SpanBert

- SpanBert:

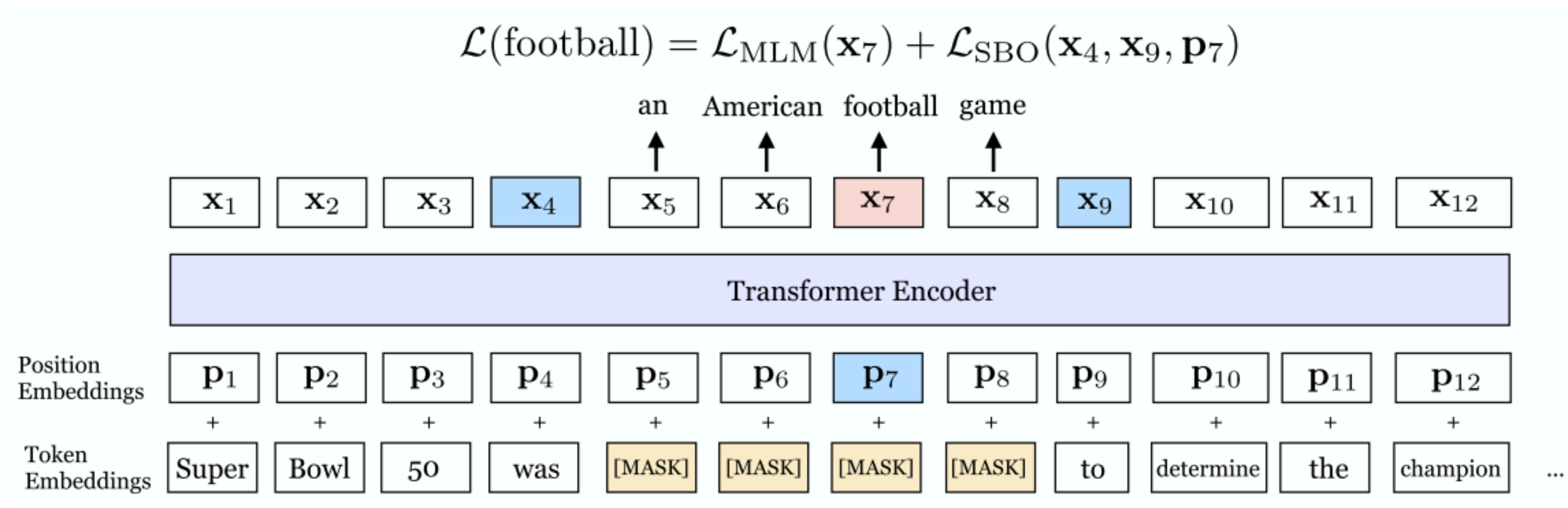


Figure 1: An illustration of SpanBERT. In this example, the span *an American football game* is masked. The span boundary objective then uses the boundary tokens *was* and *to* to predict each token in the masked span.

-

[Mandar J, Danqi C, Yinhan L, Daniel S, Luke Z, and Omer L. SpanBERT: Improving pre-training by representing and predicting spans. arXiv preprint arXiv:1907.10529, 2019]

Pre-training objective: ERNIE

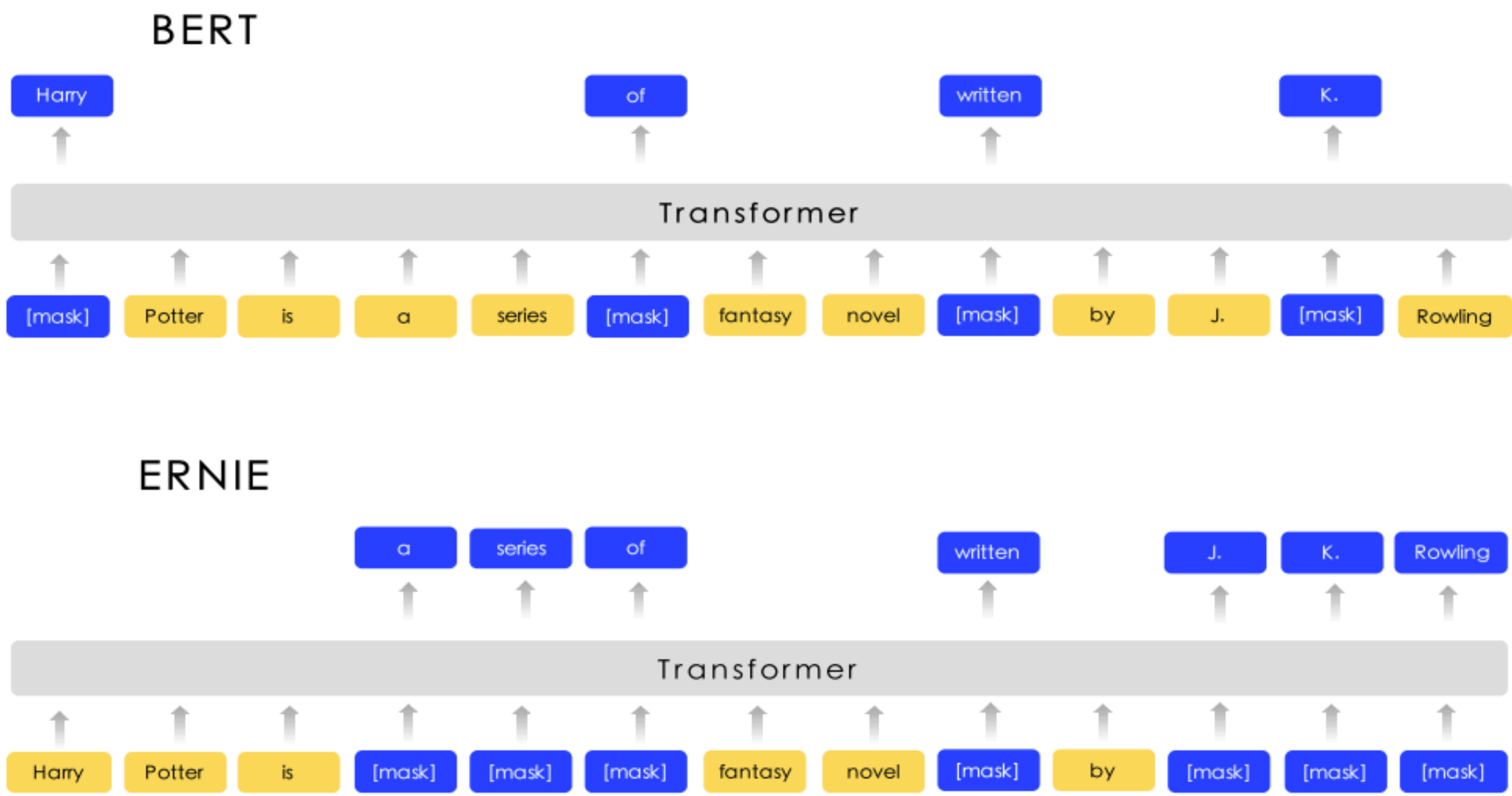
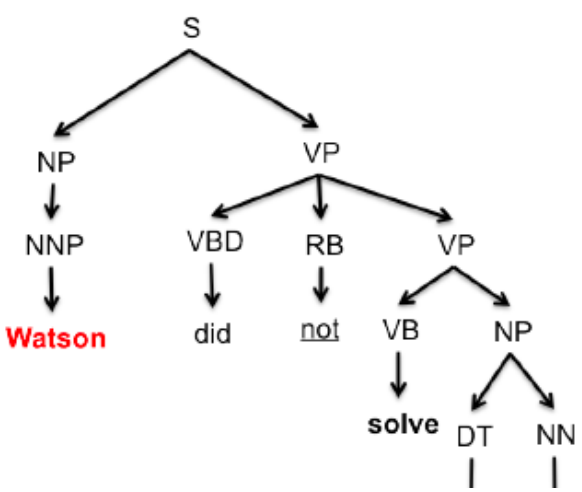
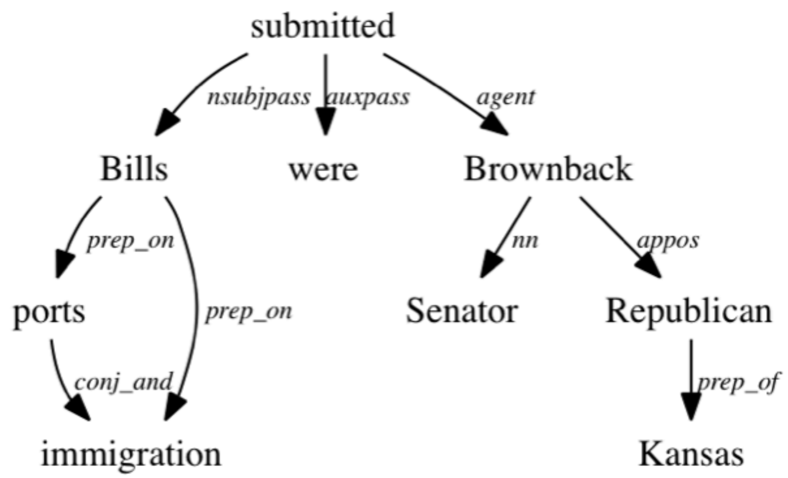


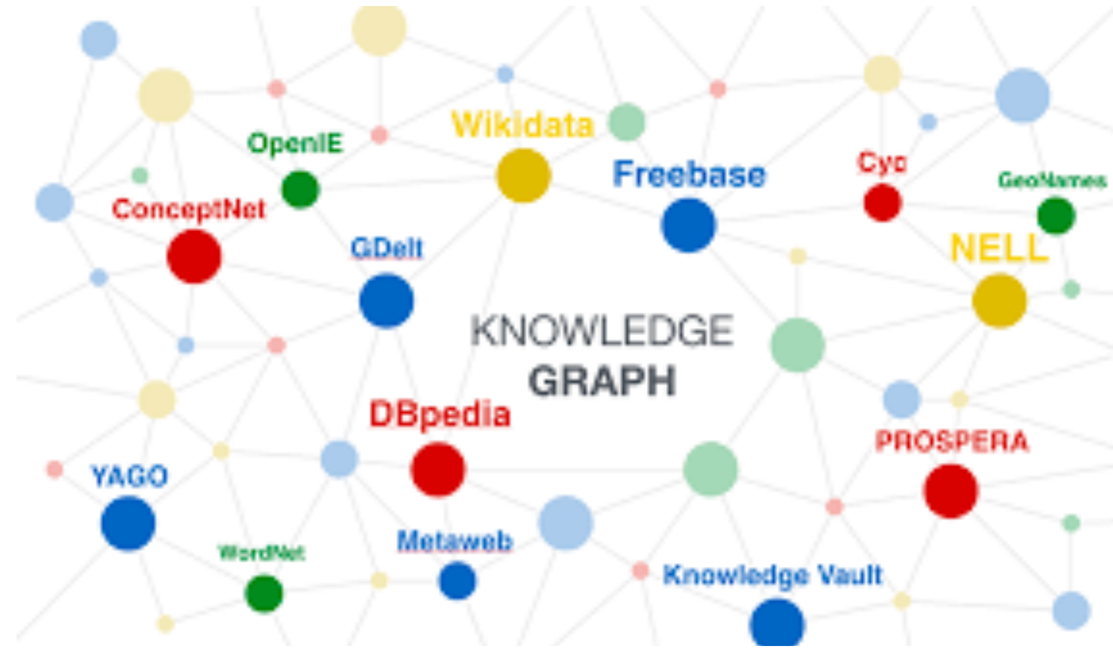
Figure 1: The different masking strategy between BERT and ERNIE

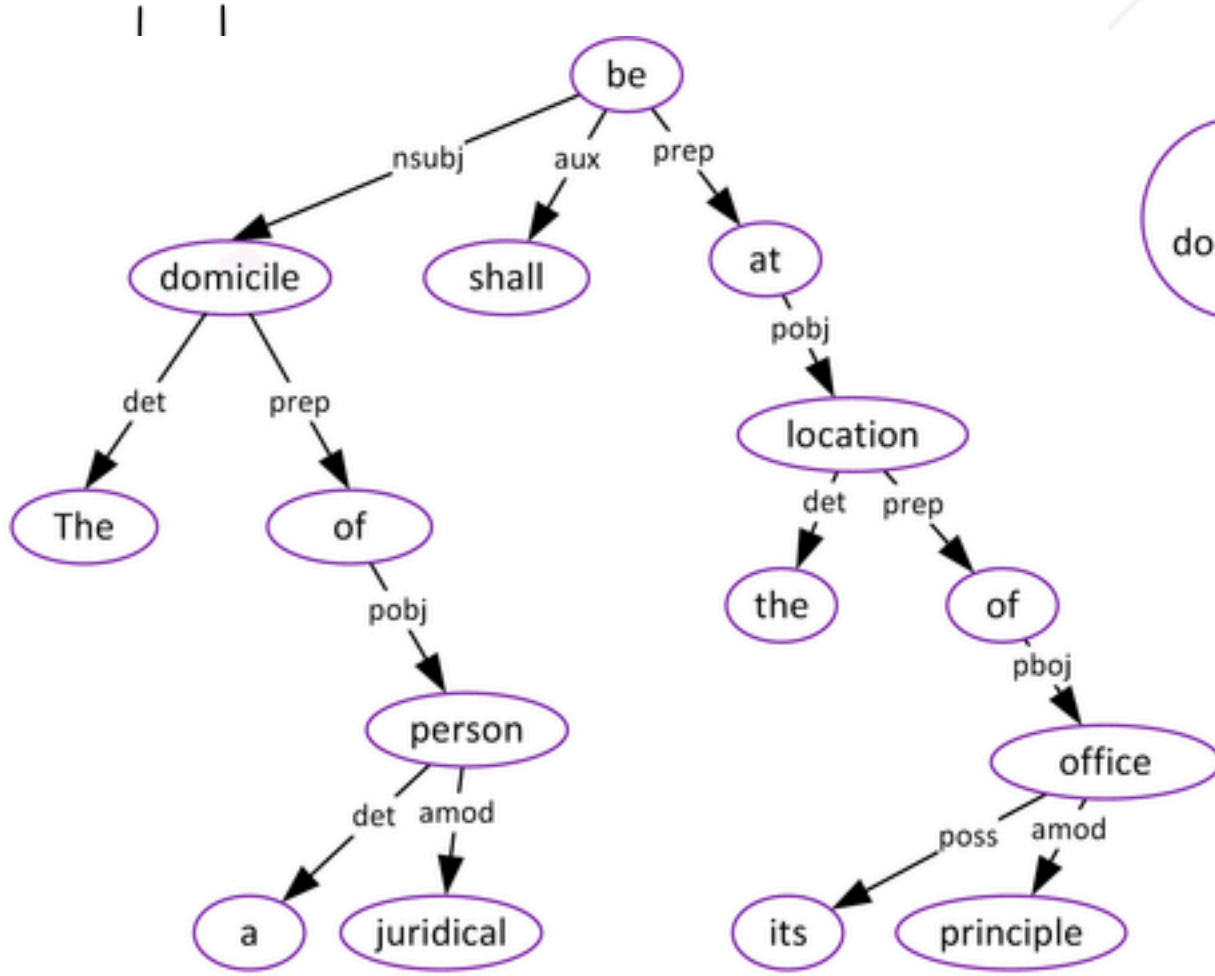
[Yu S, Shuohuan W, Yukun L, Shikun F, Xuyi C, Han Z, Xinlun T, Danxiang Z, Hao T, and HuaWu. ERNIE: En- hanced representation through knowledge integra- tion. arXiv preprint arXiv:1904.09223, 2019]

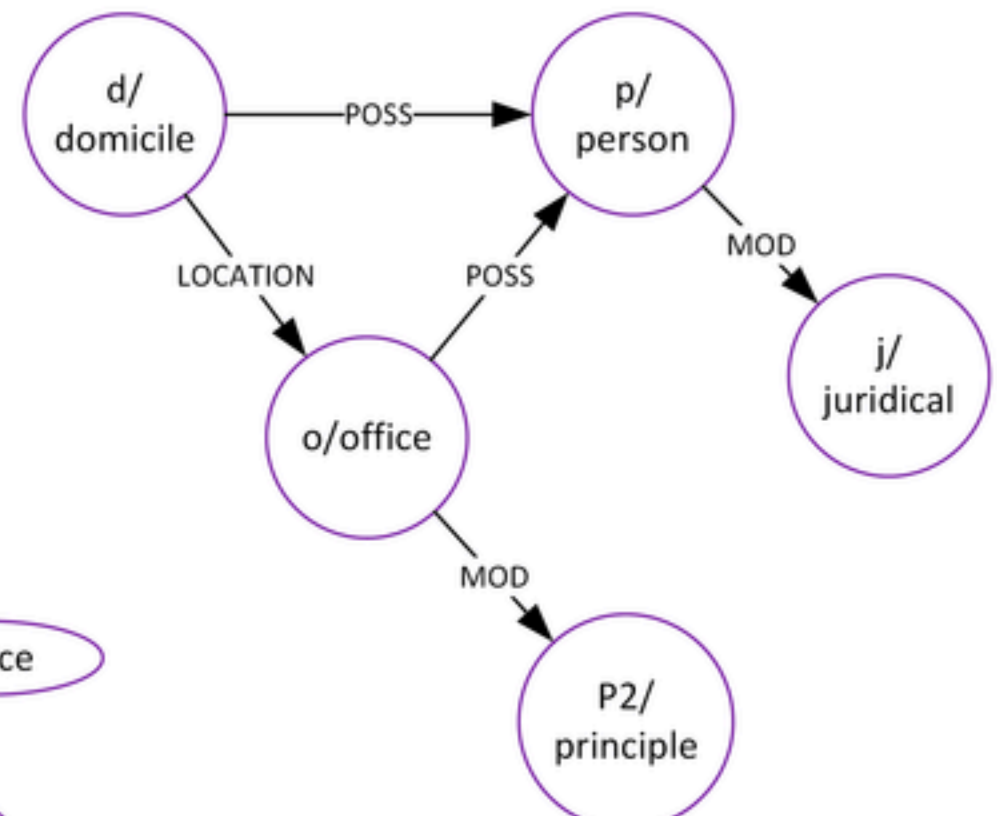
Besides Sentences







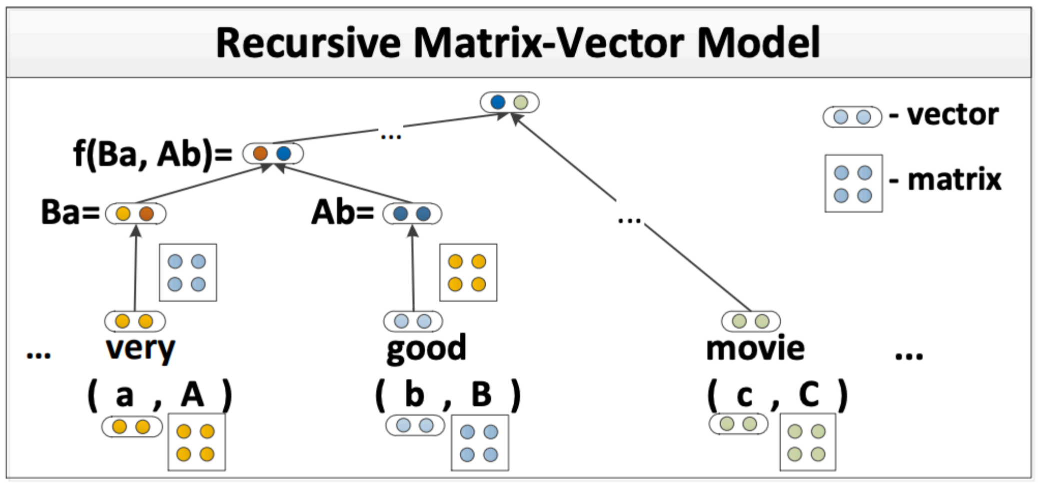


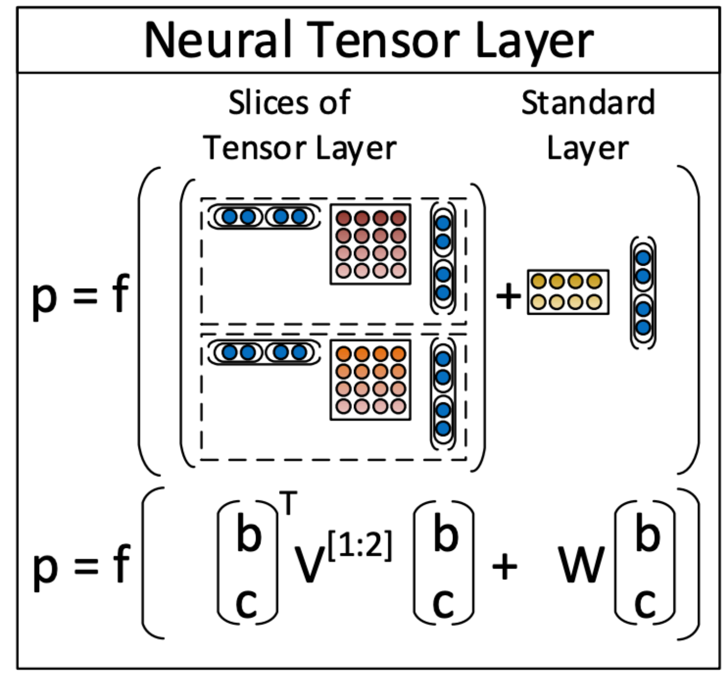


Tree

- Syntax tree structures are widely used in NLP, offering informative syntax information inside the tree structure, which is helpful to downstream task performance.
- Many related works study how to encode such tree structures.

Recursive NNs



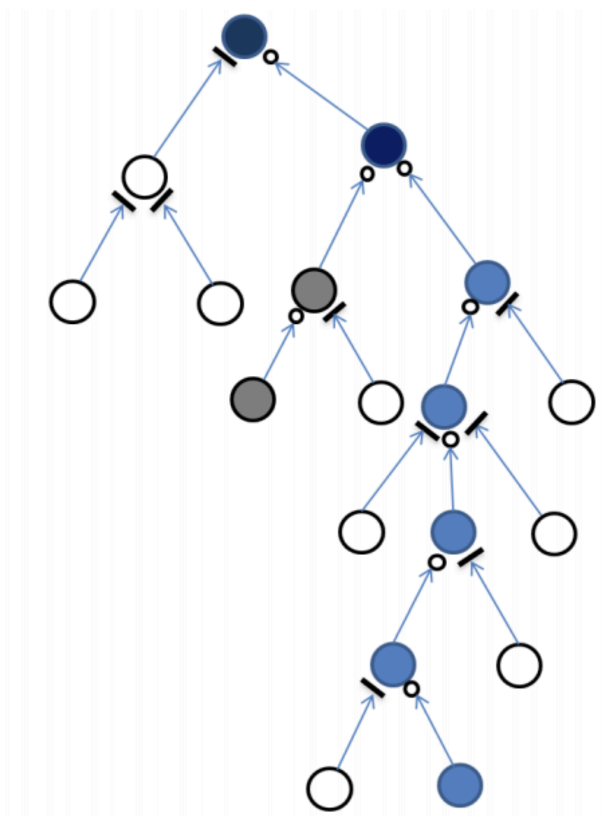


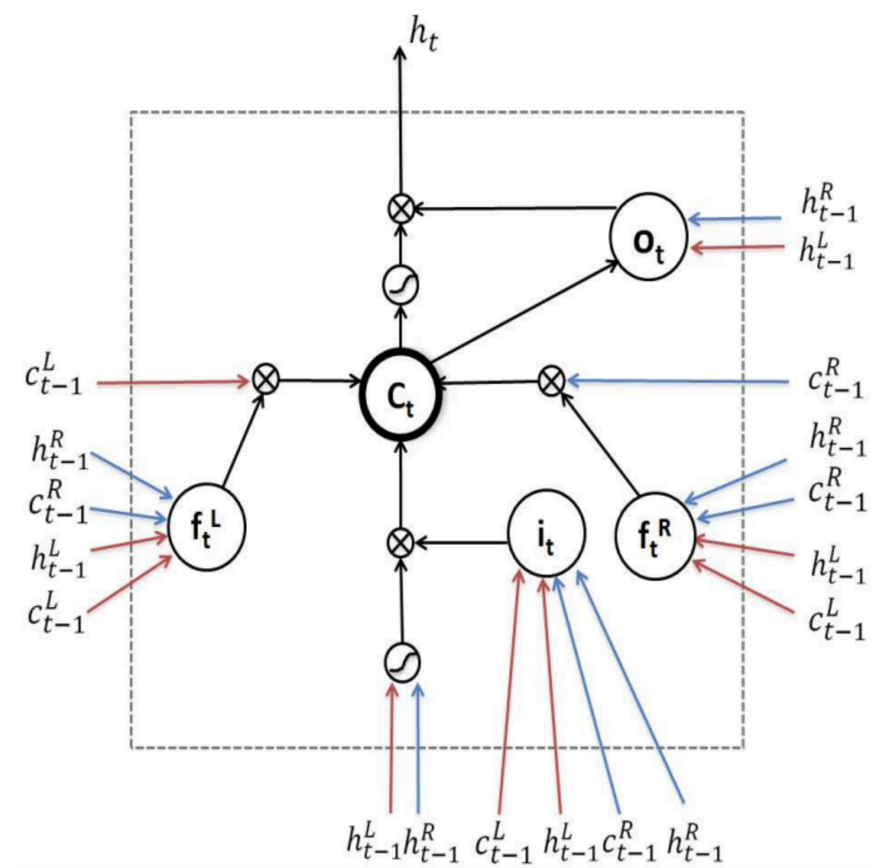
Recursive NNs like tree (instead of sequential) structure Recurrent NNs.

From leafs to root, encoding the whole tree from bottom to up.

Socher, Richard, et al. Parsing natural scenes and natural language with recursive neural networks, *in ICML,* 2011.

Tree LSTM





Tree LSTM likes tree (instead of sequential) structure LSTM.

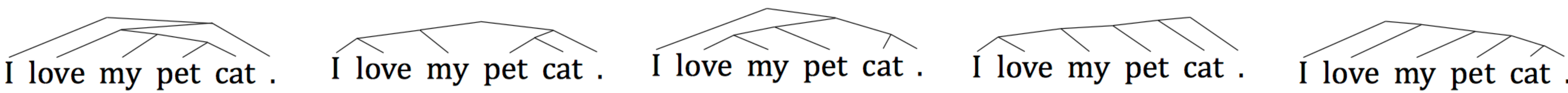
Zhu X, Sobihani P, Guo H. Long short-term memory over recursive structures, in ICML, 2015.

Tai, Kai Sheng, Richard Socher, and Christopher D. Manning. Improved semantic representations from tree-structured long short-term memory networks, *in* ACL, 2015.

On Tree Based Neural Sentence Modeling

However, tree structured NNs have been less useful in recent days.

- This paper studies to which extend tree-based encoders help downstream tasks.



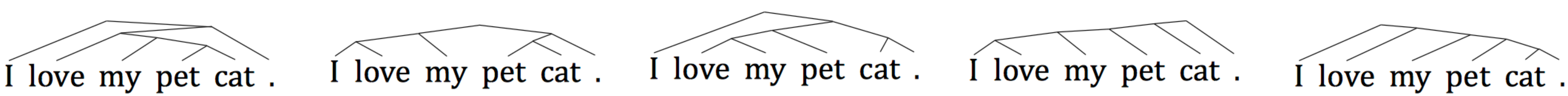
(a) Parsing tree. (b) Balanced tree. (c) Gumbel tree. (d) Left-branching tree. (e) Right-branching tree.

Haoyue Shi, Hao Zhou, Jiaze Chen, Lei Li. On Tree-Based Neural Sentence Modeling, in EMNLP, 2018.

On Tree Based Neural Sentence Modeling

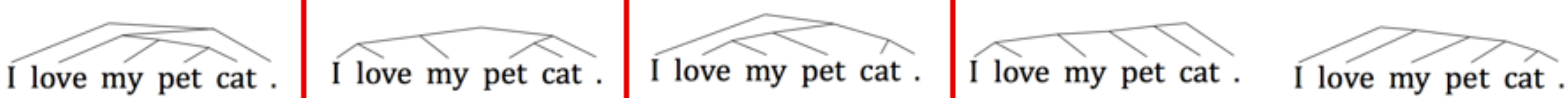
However, tree structured NNs have been less useful in recent days.

- This paper studies to which extend tree-based encoders help downstream tasks.



(a) Parsing tree. (b) Balanced tree. (c) Gumbel tree. (d) Left-branching tree. (e) Right-branching tree.

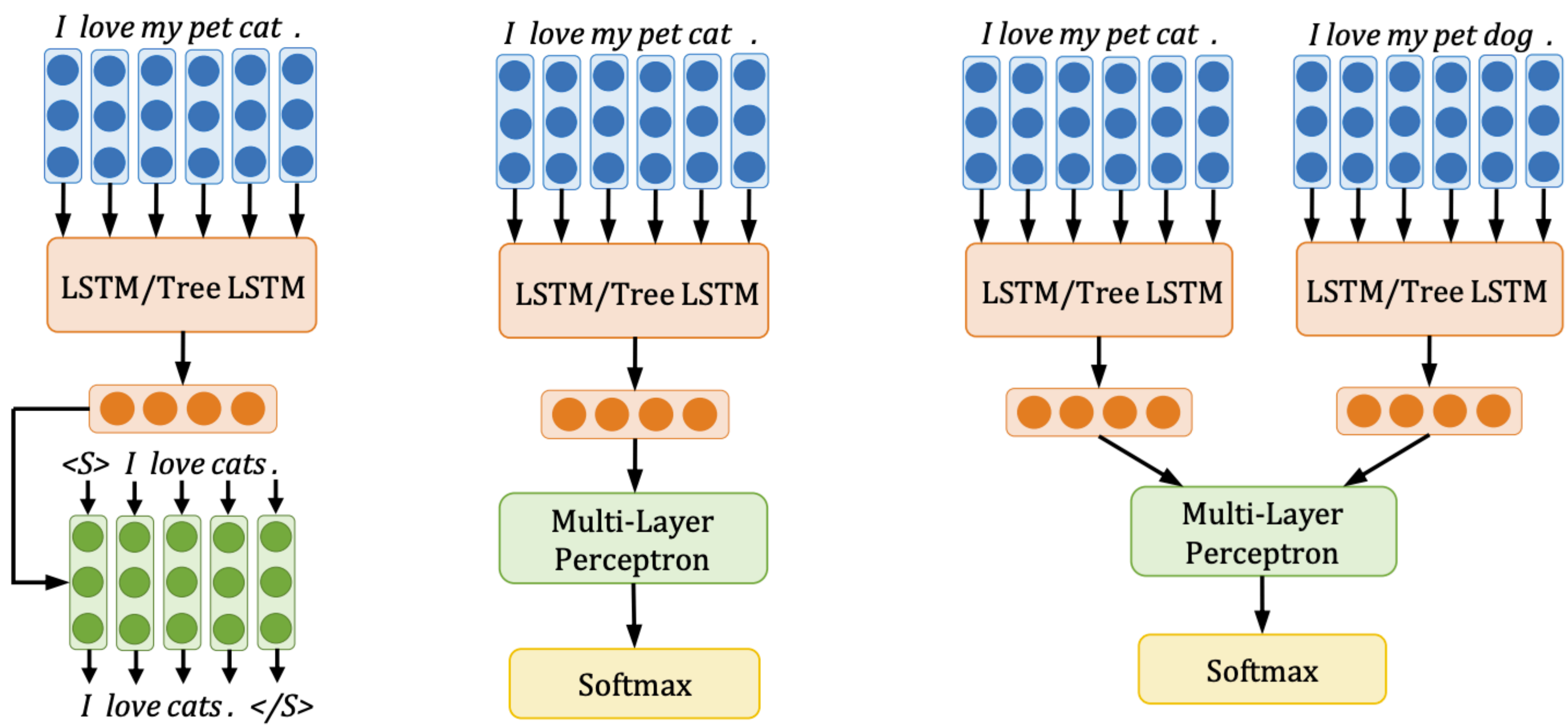
Candidates include 2 Trivial Trees without Syntax



(a) Parsing tree. (b) Balanced tree. (c) Gumbel tree. (d) Left-branching tree. (e) Right-branching tree.

Trivial Trees

Input Tree Representations into Different Tasks



(a) Encoder-decoder framework for sentence generation.

(b) Encoder-classifier framework for sentence classification.

(c) Siamese encoder-classifier framework for sentence relation classification.

Experimental Results

Model	*Sentence Classification*					*Sentence Relation*		*Sentence Generation*		
	AGN	**ARP**	**ARF**	**DBpedia**	**WSR**	**NLI**	**Conj**	**Para**	**MT**	**AE**
Latent Trees										
Gumbel	91.8	87.1	48.4	98.6	66.7	80.4	51.2	20.4	17.4	39.5
+bi-leaf-RNN	91.8	**88.1**	**49.7**	98.7	69.2	**82.9**	53.7	20.5	22.3	75.3
(Constituency) Parsing Trees										
Parsing	91.9	87.5	49.4	**98.8**	66.6	81.3	52.4	19.9	19.1	44.3
+bi-leaf-RNN	92.0	88.0	49.6	**98.8**	68.6	82.8	53.4	20.4	22.2	72.9
Trivial Trees										
Balanced	92.0	87.7	49.1	98.7	66.2	81.1	52.1	19.7	19.0	49.4
+bi-leaf-RNN	**92.1**	87.8	**49.7**	**98.8**	**69.6**	82.6	**54.0**	20.5	22.3	76.0
Left-branching	91.9	87.6	48.5	98.7	67.8	81.3	50.9	19.9	19.2	48.0
+bi-leaf-RNN	91.2	87.6	48.9	98.6	67.7	82.8	53.3	20.6	21.6	72.9
Right-branching	91.9	87.7	49.0	**98.8**	68.6	81.0	51.3	20.4	19.7	54.7
+bi-leaf-RNN	91.9	87.9	49.4	98.7	68.7	82.8	53.5	**20.9**	**23.1**	**80.4**
Linear Structures										
LSTM	91.7	87.8	48.8	98.6	66.1	82.6	52.8	20.3	19.1	46.9
+bidirectional	91.7	87.8	49.2	98.7	67.4	82.8	53.3	20.2	21.3	67.0
Avg. Length	31.5	33.7	33.8	20.1	23.1	11.2	23.3	10.2	34.1	34.1

Experimental Results

Model	*Sentence Classification* AGN	ARP	ARF	DBpedia	WSR	*Sentence Relation* NLI	Conj	*Sentence Generation* Para	MT	AE
Latent Trees										
Gumbel	91.8	87.1	48.4	98.6	66.7	80.4	51.2	20.4	17.4	39.5
+*bi-leaf-RNN*	91.8	**88.1**	**49.7**	98.7	69.2	**82.9**	53.7	20.5	22.3	75.3
(Constituency) Parsing Trees										
Parsing	91.9	87.5	49.4	**98.8**	66.6	81.3	52.4	19.9	19.1	44.3
+*bi-leaf-RNN*	92.0	88.0	49.6	**98.8**	68.6	82.8	53.4	20.4	22.2	72.9
Trivial Trees										
Balanced	92.0	87.7	49.1	98.7	66.2	81.1	52.1	19.7	19.0	49.4
+*bi-leaf-RNN*	**92.1**	87.8	**49.7**	**98.8**	**69.6**	82.6	**54.0**	20.5	22.3	76.0
Left-branching	91.9	87.6	48.5	98.7	67.8	81.3	50.9	19.9	19.2	48.0
+*bi-leaf-RNN*	91.2	87.6	48.9	98.6	67.7	82.8	53.3	20.6	21.6	72.9
Right-branching	91.9	87.7	49.0	**98.8**	68.6	81.0	51.3	20.4	19.7	54.7
+*bi-leaf-RNN*	91.9	87.9	49.4	98.7	68.7	82.8	53.5	**20.9**	**23.1**	**80.4**
Linear Structures										
LSTM	91.7	87.8	48.8	98.6	66.1	82.6	52.8	20.3	19.1	46.9
+*bidirectional*	91.7	87.8	49.2	98.7	67.4	82.8	53.3	20.2	21.3	67.0
Avg. Length	31.5	33.7	33.8	20.1	23.1	11.2	23.3	10.2	34.1	34.1

Trivial trees work better!!!

All experiments are conducted 5 times to get average results.

Visualization

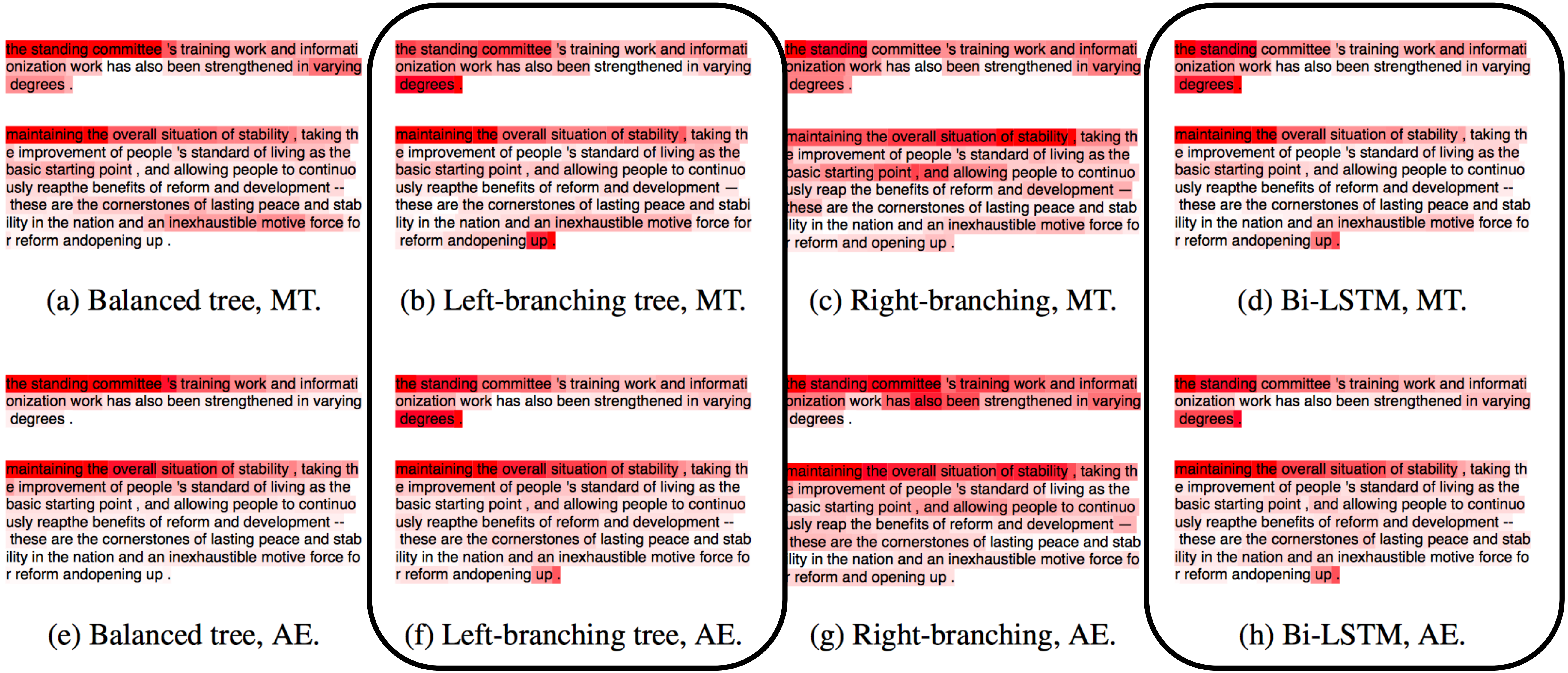


Figure 7: Saliency visualization of words in learned MT and AE models. Darker means more important to the sentence encoding.

Left-branching trees pay more attention to left words, but balanced trees treat all words fairly, and learns the weights by model.

Shallow Trees work Better

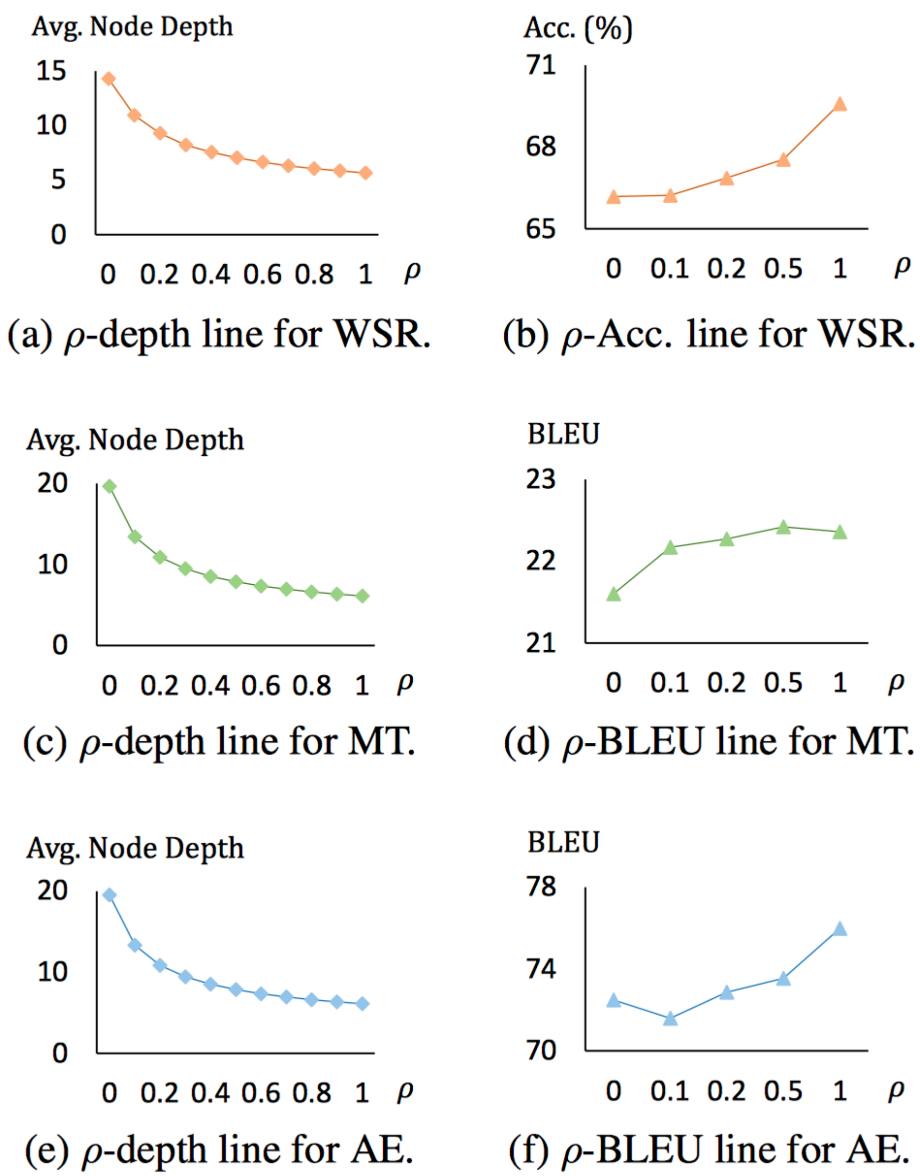
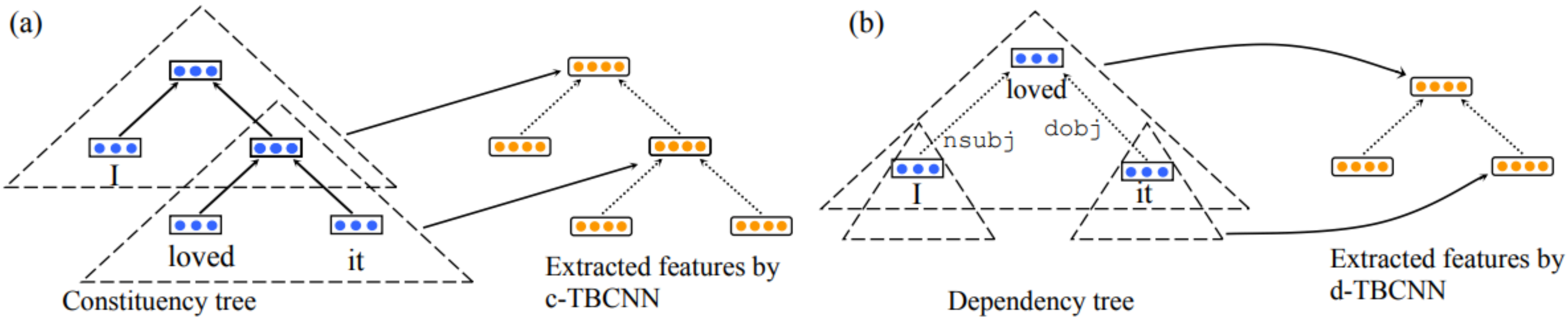


Figure 5: ρ-depth and ρ-performance lines for three tasks. There is a trend that the depth drops and the performance raises with the growth of ρ.

Constructing balanced trees with varying depth.

Shallow trees leads to better performances.

Tree-Based Convolution



Constituency tree **Dependency tree**

Lili Mou, Hao Peng, Ge Li, Yan Xu, Lu Zhang, Zhi Jin. Discriminative neural sentence modeling by tree-based convolution. In *EMNLP*, 2015.

Graph Network

Spectral convolution

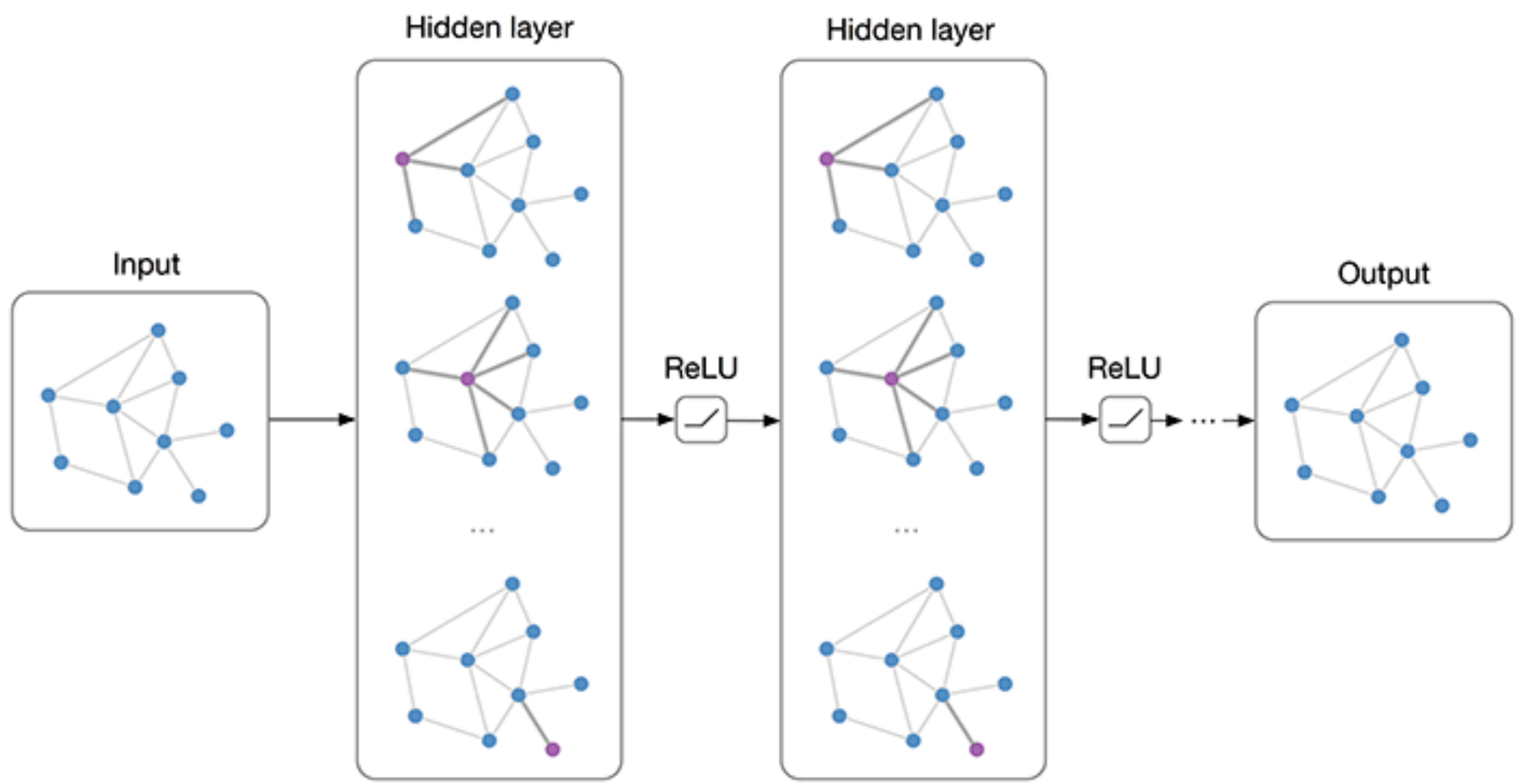
[Kipf & Welling, 2016]

Spatial convolution

[Duvenaud et al., 2015]

Other graph operations
E.g., attention

[Veličković et al., 2018]



Graph convolution neural networks can encode the graph structures as distributed representations.

Summary for Discrete Input Space

- Representing discrete tokens
  - Pretrained word embeddings by table lookup
  - Pretrained word embeddings within context
- Representing discrete structures
  - Trees, graphs, etc.
  - Structured CNN, RNN, attention, etc.

References

- Bishop CM. *Pattern Recognition and Machine Learning*. Springer, 2006.
- Welch G, Bishop G. An introduction to the Kalman filter.
- Guu K, Pasupat P, Liu EZ, Liang P. From language to programs: Bridging reinforcement learning and maximum marginal likelihood. In *ACL*, 2017.
- Kingma DP, Welling M. Auto-encoding variational Bayes. In *ICLR*, 2014.
- Sutton RS, Barto AG. Introduction to Reinforcement Learning. 1998.
- Jang E, Gu S, Poole B. Categorical reparameterization with Gumbel-softmax. In *ICLR*, 2017.
- Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.
- Socher, Richard, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *EMNLP*, 2011.
- Bowman, S.R., Gauthier, J., Rastogi, A., Gupta, R., Manning, C.D. and Potts, C., 2016. A fast unified model for parsing and sentence understanding. In *ACL*, 2016.
- Kipf, T.N. and Welling, M., 2016. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.
- Lili Mou, Hao Peng, Ge Li, Yan Xu, Lu Zhang, Zhi Jin. Discriminative neural sentence modeling by tree-based convolution. In EMNLP, 2015.
- Duvenaud DK, Maclaurin D, Iparraguirre J, Bombarell R, Hirzel T, Aspuru-Guzik A, Adams RP. Convolutional networks on graphs for learning molecular fingerprints. In *NIPS*, 2015.
- Veličković P, Cucurull G, Casanova A, Romero A, Lio P, Bengio Y. Graph attention networks. In *ICLR*, 2018.