

Part I: Text Embedding and Language Models

KDD 2021 Tutorial

On the Power of Pre-Trained Text Representations: Models and Applications in Text Mining

Yu Meng, Jiaxin Huang, Yu Zhang, Jiawei Han

Computer Science, University of Illinois at Urbana-Champaign

August 14, 2021

Outline

- Introduction to text representations 
- Context-free embeddings
- Deep contextualized embeddings via neural language models
- Extend unsupervised embeddings to incorporate weak supervision

Overview of Text Representation Development

- Texts need to be represented as numbers/vectors so that computer programs can process them
- How were texts represented in history?

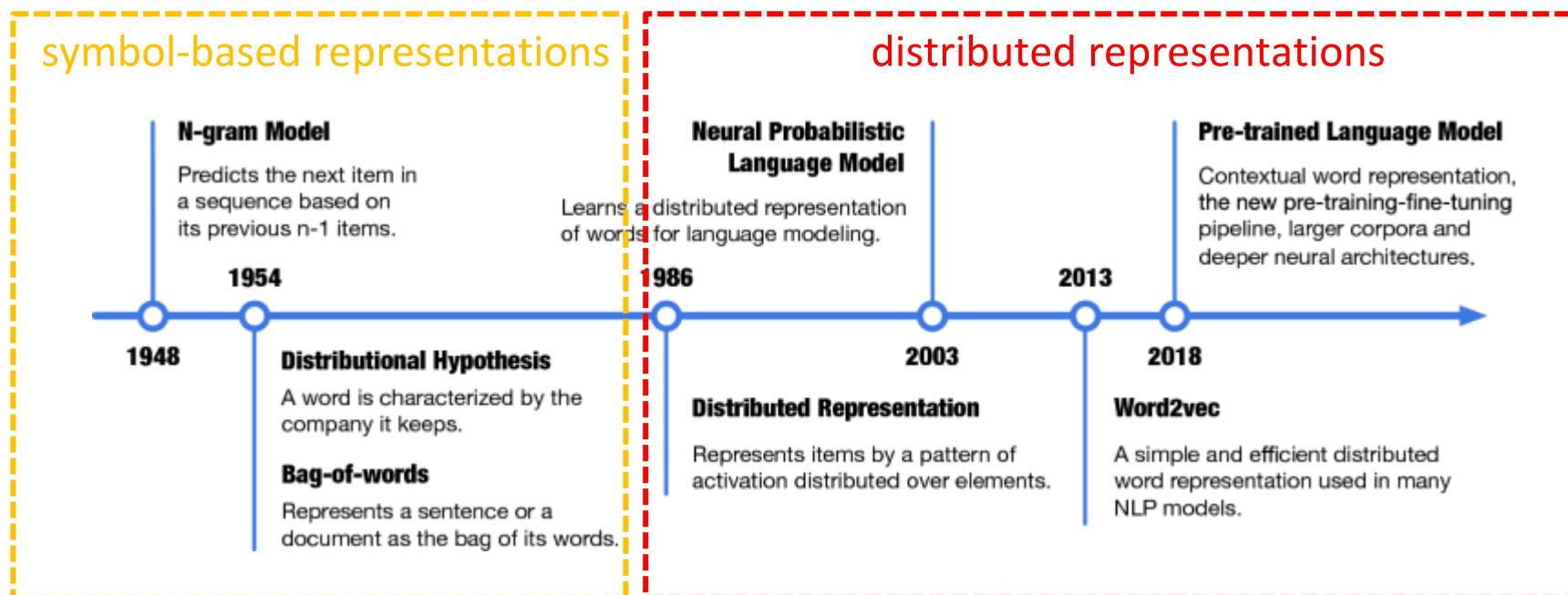


Figure from: Liu Z., Lin Y., Sun M. (2020) Representation Learning and NLP. In: Representation Learning for Natural Language Processing. Springer, Singapore.

Symbol-Based Text Representations

- One-to-one correspondence between text units and representation elements
- e.g., “dogs” = [1, 0, 0, 0, 0]; “cats” = [0, 1, 0, 0, 0]; “cars” = [0, 0, 1, 0, 0];
“like” = [0, 0, 0, 1, 0]; “I” = [0, 0, 0, 0, 1]
- Bag-of-words representation of documents: Describe a document according to which words are present, ignoring word ordering
 - e.g., “I like dogs” may be represented as [1, 0, 0, 1, 1]
 - Can further weigh words with Term Frequency (TF) and/or Inverse Document Frequency (IDF)
- Issues: Many dimensions needed (curse of dimensionality!); vectors do not reflect semantic similarity

Distributed Text Representations

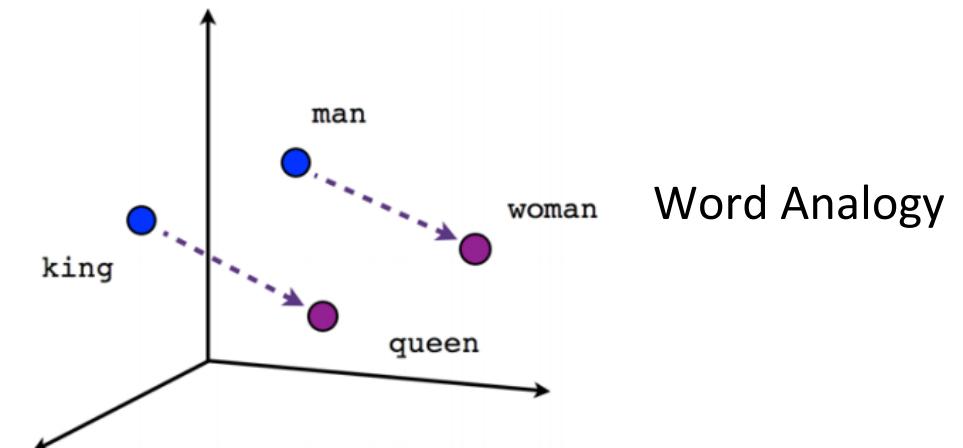
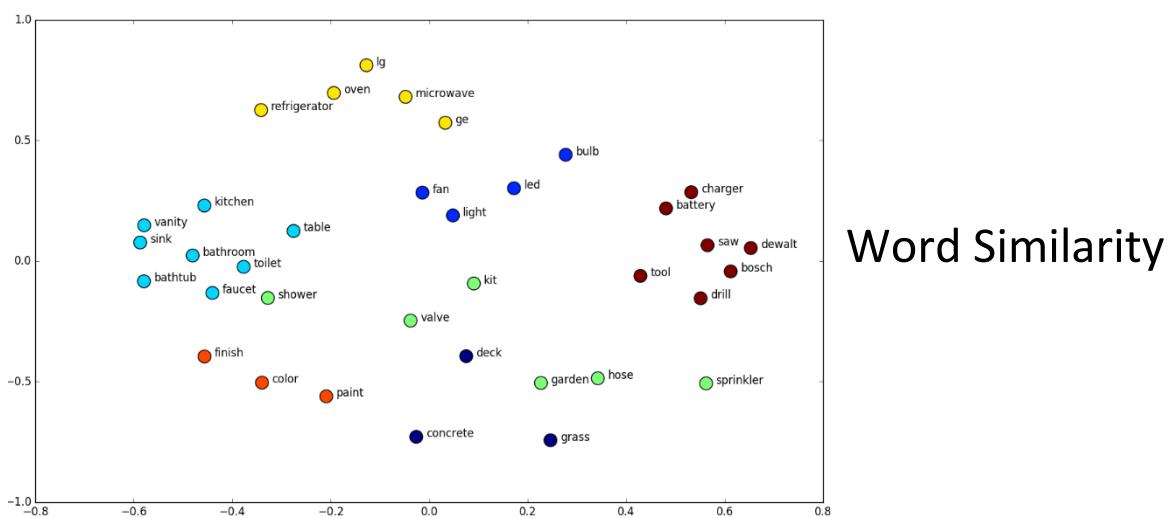
- ❑ The Distributional Hypothesis: “a word is characterized by the company it keeps”
 - ❑ words that are used and occur in the same contexts tend to purport similar meanings
- ❑ Distributed representations (i.e., embeddings)
 - ❑ The representation of any text unit is distributed over all vector dimensions as continuous values (instead of 0/1s)
 - ❑ Advantage: Vectors are dense and lower-dimensional, better at capturing semantic similarity
- ❑ Distributed representations are usually learned based on the distributional hypothesis—vector space similarity reflects semantic similarity
- ❑ We focus on distributed representations in this tutorial

Outline

- Introduction to text representations
- Context-free embeddings 
- Deep contextualized embeddings via neural language models
- Extend unsupervised embeddings to incorporate weak supervision

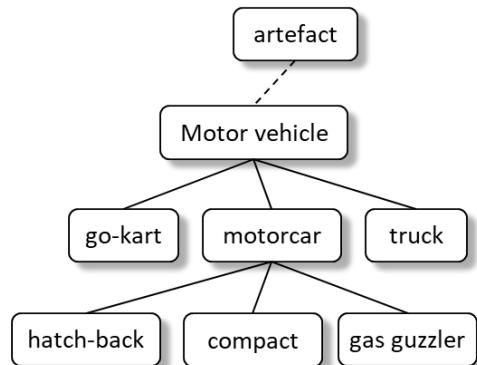
Introduction to Text Embeddings

- ❑ A milestone in NLP and ML:
 - ❑ Unsupervised learning of text representations—No supervision needed
 - ❑ Embed one-hot vectors into lower-dimensional space—Address “curse of dimensionality”
 - ❑ Word embedding captures useful properties of word semantics
 - ❑ Word similarity: Words with similar meanings are embedded closer
 - ❑ Word analogy: Linear relationships between words (e.g., king - queen = man - woman)

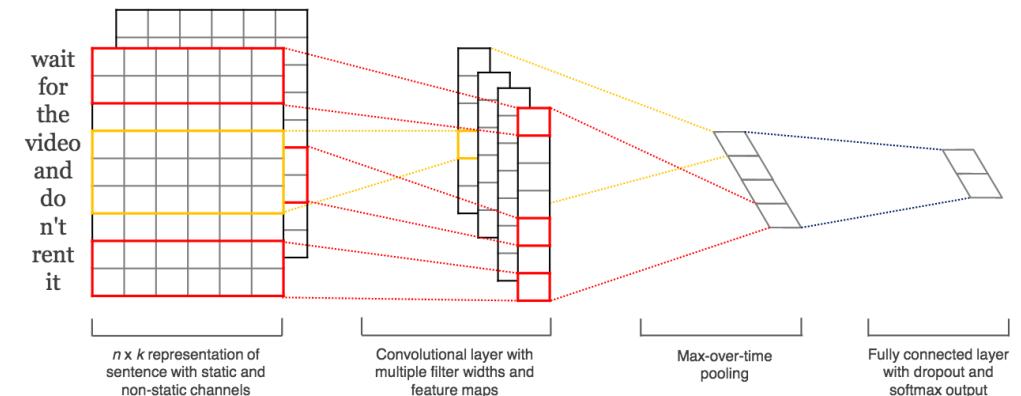


Applications of Text Embeddings

- ❑ Text embeddings can be used in a lot of downstream applications
 - ❑ Word/token/entity-level tasks
 - ❑ Keyword extraction/clustering
 - ❑ Taxonomy construction
 - ❑ Document/paragraph-level tasks
 - ❑ Document classification/clustering/retrieval
 - ❑ Question answering/text summarization



Taxonomy Construction



Document Classification

Outline

- ❑ Introduction to text representations
- ❑ Context-free embeddings
 - ❑ Euclidean space: Word2Vec, GloVe, fastText
 - ❑ Hyperbolic space: Poincaré embeddings
 - ❑ Spherical space: JoSE
- ❑ Deep contextualized embeddings via neural language models
- ❑ Extend unsupervised embeddings to incorporate weak supervision



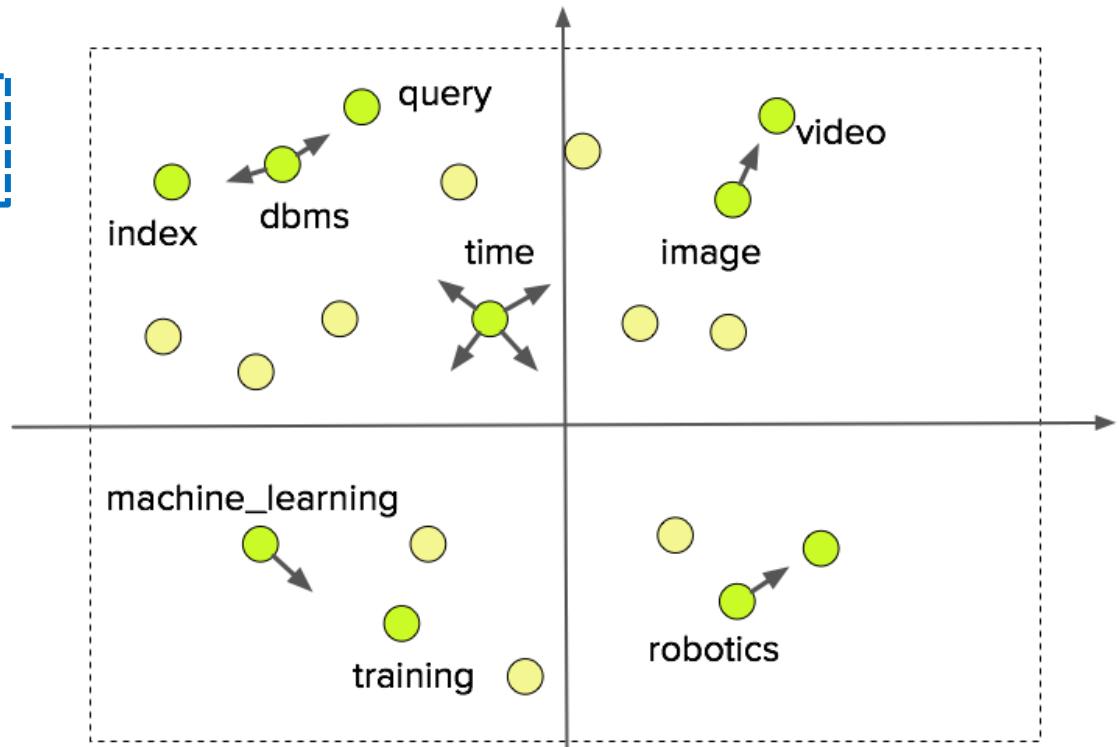
Word2Vec

- Many text embeddings are learned in the Euclidean space (without constraints on vectors)
- Word2Vec maximizes the probability of observing a word based on its local contexts
- As a result, semantically coherent terms are more likely to have close embeddings

Co-occurred words in a **local context window**

$$\frac{1}{T} \sum_{t=1}^T \sum_{-c \leq j \leq c, j \neq 0} \log p(w_{t+j} | w_t)$$

$$p(w_O | w_I) = \frac{\exp(v'_{w_O}^\top v_{w_I})}{\sum_{w=1}^W \exp(v'_{w'}^\top v_{w_I})}$$

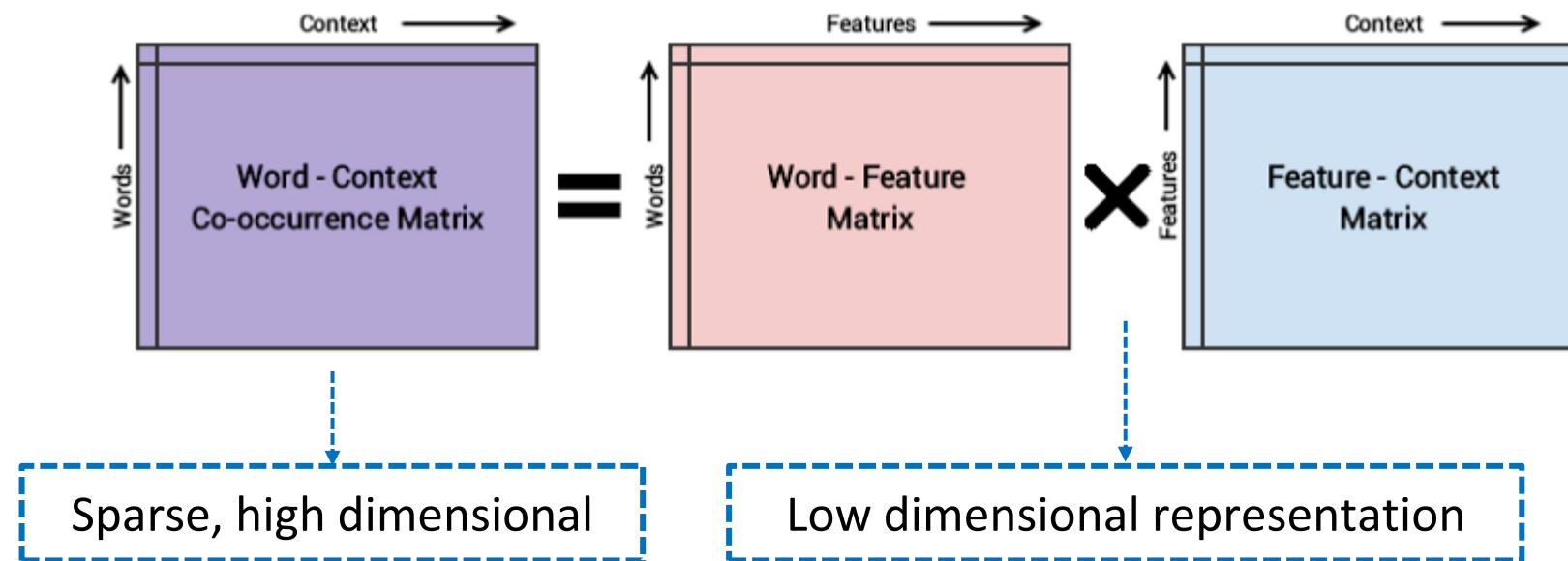


Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. NIPS.

GloVe

- ❑ GloVe factorizes a global co-occurrence matrix derived from the entire corpus
- ❑ Low-dimensional representations are obtained by solving a least-squares problem to “recover” the co-occurrence matrix

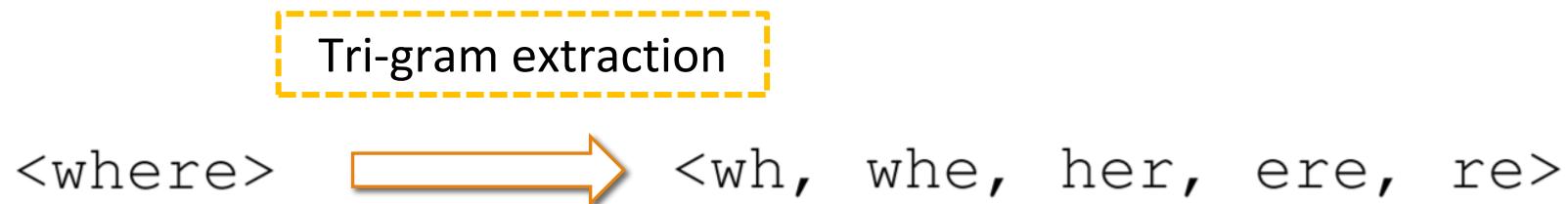
$$J = \sum_{i,j=1}^V f(X_{ij}) (w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij})^2$$



Pennington, J., Socher, R., & Manning, C.D. (2014). GloVe: Global Vectors for Word Representation. EMNLP.

fastText

- fastText improves upon Word2Vec by incorporating subword information into word embedding



- fastText allows sharing subword representations across words, since words are represented by the aggregation of their n-grams

Word2Vec probability expression

$$p(w_O|w_I) = \frac{\exp(v'_{w_O}^\top v_{w_I})}{\sum_{w=1}^W \exp(v'_{w'}^\top v_{w_I})}$$

N-gram embedding

Represent a word by the sum of the vector representations of its n-grams

Outline

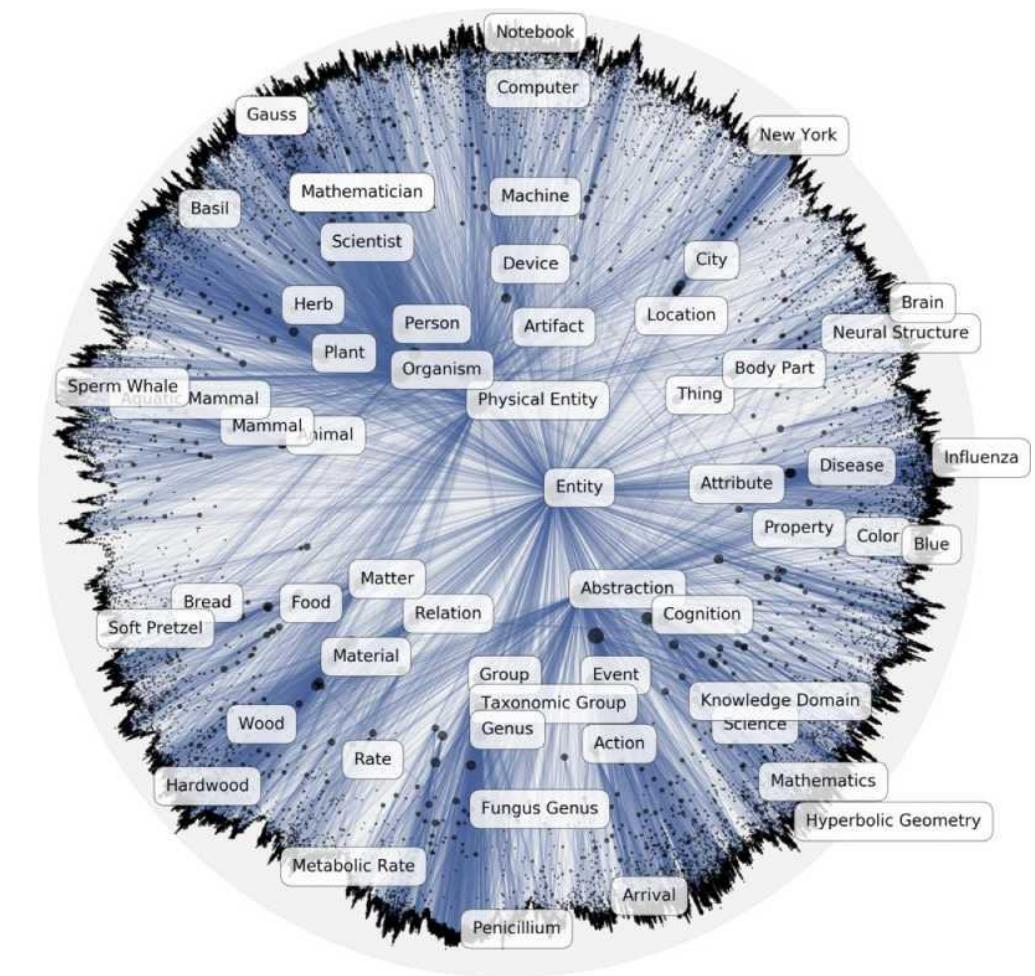
- ❑ Introduction to text representations
- ❑ Context-free embeddings
 - ❑ Euclidean space: Word2Vec, GloVe, fastText
 - ❑ Hyperbolic space: Poincaré embeddings
 - ❑ Spherical space: JoSE
- ❑ Deep contextualized embeddings via neural language models
- ❑ Extend unsupervised embeddings to incorporate weak supervision



Hyperbolic Embedding: Poincaré embedding

- Why non-Euclidean embedding space?
 - Data can have specific structures that Euclidean-space models struggle to capture
 - The hyperbolic space
 - Continuous version of trees
 - Naturally equipped to model hierarchical structures
 - Poincaré embedding
 - Learn hierarchical representations by pushing general terms to the origin of the Poincaré ball, and specific terms to the boundary

$$d(\mathbf{u}, \mathbf{v}) = \operatorname{arcosh} \left(1 + 2 \frac{\|\mathbf{u} - \mathbf{v}\|^2}{(1 - \|\mathbf{u}\|^2)(1 - \|\mathbf{v}\|^2)} \right)$$



Nickel, M., & Kiela, D. (2017). Poincaré Embeddings for Learning Hierarchical Representations. NIPS.

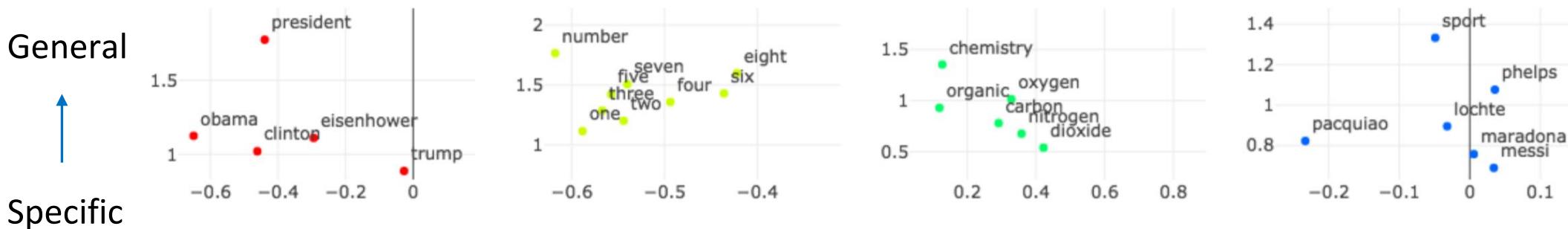
Texts in Hyperbolic Space: Poincaré GloVe

- GloVe in hyperbolic space
- Motivation: latent hierarchical structure of words exists among text
 - Hypernym-hyponym
 - Textual entailment
- Approach: use hyperbolic kernels!
- Effectively model generality/specificity

$$J = \sum_{i,j=1}^V f(X_{ij}) \left(\boxed{w_i^T \tilde{w}_j} + b_i + \tilde{b}_j - \log X_{ij} \right)^2 \quad \text{GloVe}$$

Hyperbolic metric

$$J = \sum_{i,j=1}^V f(X_{ij}) \left(\boxed{-h(d(w_i, \tilde{w}_j))} + b_i + \tilde{b}_j - \log X_{ij} \right)^2 \quad \text{Poincaré GloVe}$$



Tifrea, A., Bécigneul, G., & Ganea, O. (2019). Poincaré GloVe: Hyperbolic Word Embeddings. ICLR.

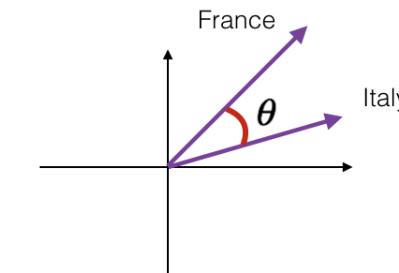
Outline

- ❑ Introduction to text representations
- ❑ Context-free embeddings
 - ❑ Euclidean space: Word2Vec, GloVe, fastText
 - ❑ Hyperbolic space: Poincaré embeddings
 - ❑ Spherical space: JoSE
- ❑ Deep contextualized embeddings via neural language models
- ❑ Extend unsupervised embeddings to incorporate weak supervision



Directional Analysis for Text Embeddings

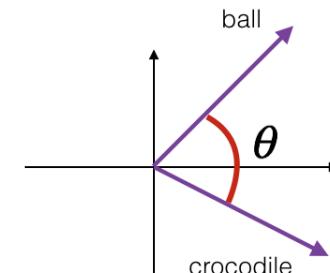
- How to use text embeddings? Mostly directional similarity (i.e., cosine similarity)
- Word similarity is derived using cosine similarity



France and Italy are quite similar

θ is close to 0°

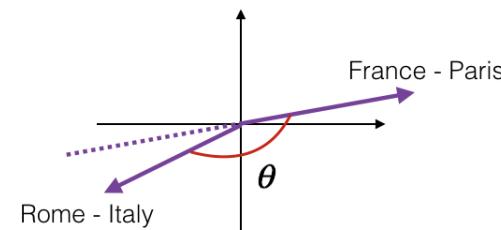
$\cos(\theta) \approx 1$



ball and crocodile are not similar

θ is close to 90°

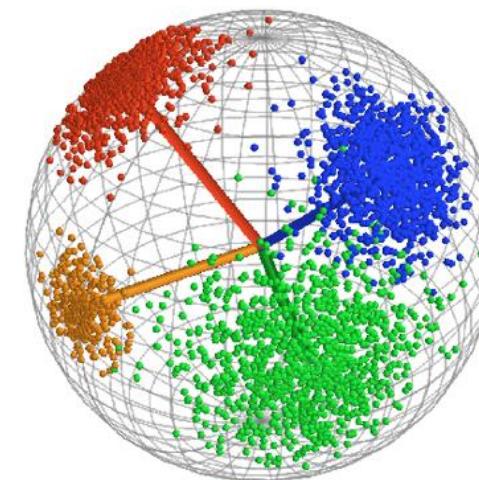
$\cos(\theta) \approx 0$



the two vectors are similar but opposite
the first one encodes (city - country)
while the second one encodes (country - city)

θ is close to 180°

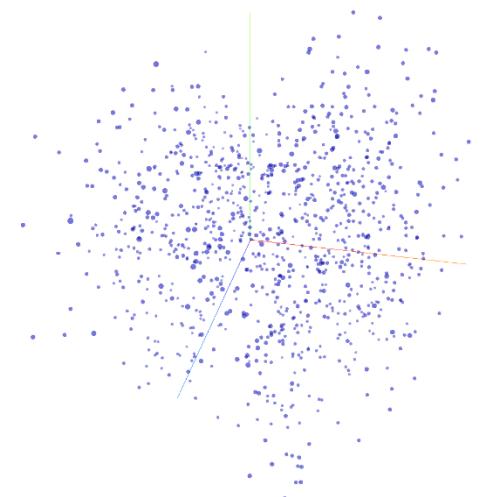
$\cos(\theta) \approx -1$



- Better clustering performances when embeddings are normalized, and spherical clustering algorithms are used (Spherical K-means)
- Vector direction is what actually matters!

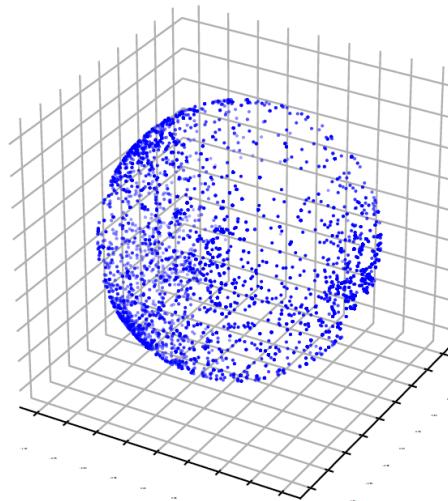
Issues with Previous Embedding Frameworks

- Although directional similarity has shown effective for various applications, previous embeddings (e.g., Word2Vec, GloVe, fastText) are trained in the Euclidean space
- A gap between training space and usage space: Trained in Euclidean space but used on sphere



Embedding Training in Euclidean Space

Post-processing
(Normalization)



Embedding Usage on the Sphere
(Similarity, Clustering, etc.)

Inconsistency Between Training and Usage

- ❑ The objective we optimize during training is not really the one we use
 - ❑ Regardless of the different training objective, Word2Vec, GloVe and fastText all optimize the embedding **dot product** during training, but **cosine similarity** is what used in applications

Embedding dot product is optimized during training

The diagram illustrates three neural network architectures: Word2Vec, GloVe, and fastText. Each architecture is shown with its corresponding loss function. A dashed blue circle highlights the term involving the dot product of embeddings, which is noted as being optimized during training.

Word2Vec:

$$p(w_O | w_I) = \frac{\exp(v'_{w_O}^\top v_{w_I})}{\sum_{w=1}^W \exp(v'_w{}^\top v_{w_I})}$$

GloVe:

$$J = \sum_{i,j=1}^V f(X_{ij}) \left(w_i^\top \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij} \right)^2$$

fastText:

$$s(w, c) = \sum_{g \in \mathcal{G}_w} \mathbf{z}_g^\top \mathbf{v}_c$$

Local Contexts May Not Be Enough

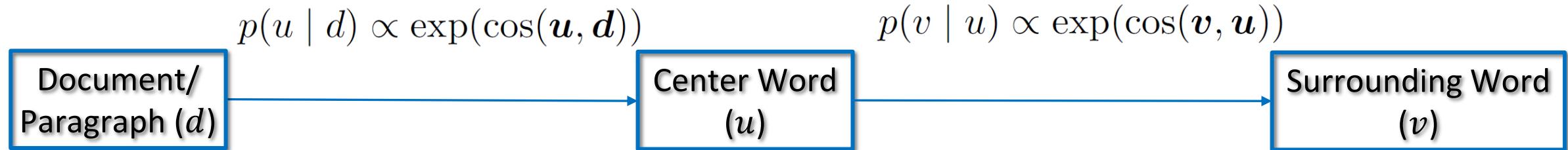
- ❑ Apart from the training/usage space inconsistency issue, previous embedding frameworks only leverage **local contexts** to learn word representations
- ❑ Local contexts can only partly define word semantics in unsupervised word embedding learning

If I hear someone screwing with my car (ie, setting off the **alarm**) and **taunting** me to come out, you can be very sure that my Colt Delta Elite will also be coming with me. It is not the screwing with the car that would get them **shot**, it is the potential physical **danger**. If they are **taunting** like that, it's very possible that they also intend to **rob** me and or do other physically **harmful** things. Here in Houston last year a woman heard the sound of someone ...

Local contexts of
“harmful”

Spherical Text Embedding: Generative Model

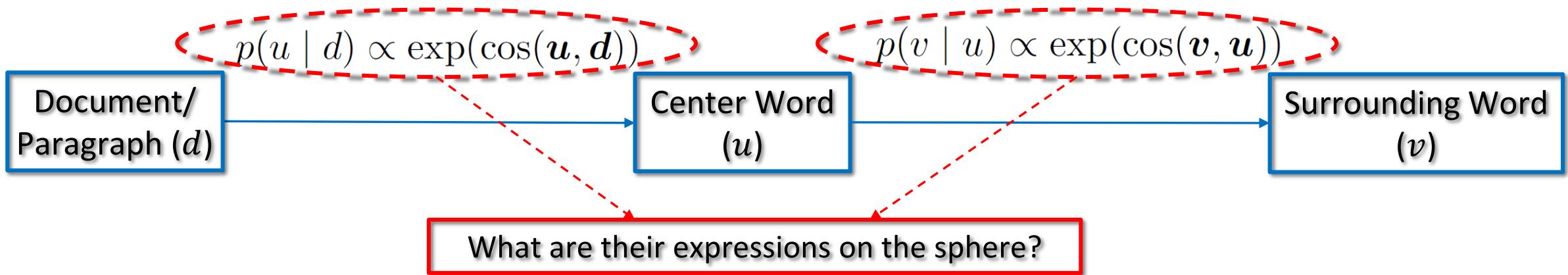
- We design a generative model on the sphere that follows how humans write articles:
 - We first have a general idea of the paragraph/document, and then start to write down each word in consistent with not only the paragraph/document, but also the surrounding words
 - Assume a two-step generation process:



Meng, Y., Huang, J., Wang, G., Zhang, C., Zhuang, H., Kaplan, L.M., & Han, J. (2019). Spherical Text Embedding. NeurIPS.

Spherical Text Embedding: Generative Model

- How to define the generative model in the spherical space?



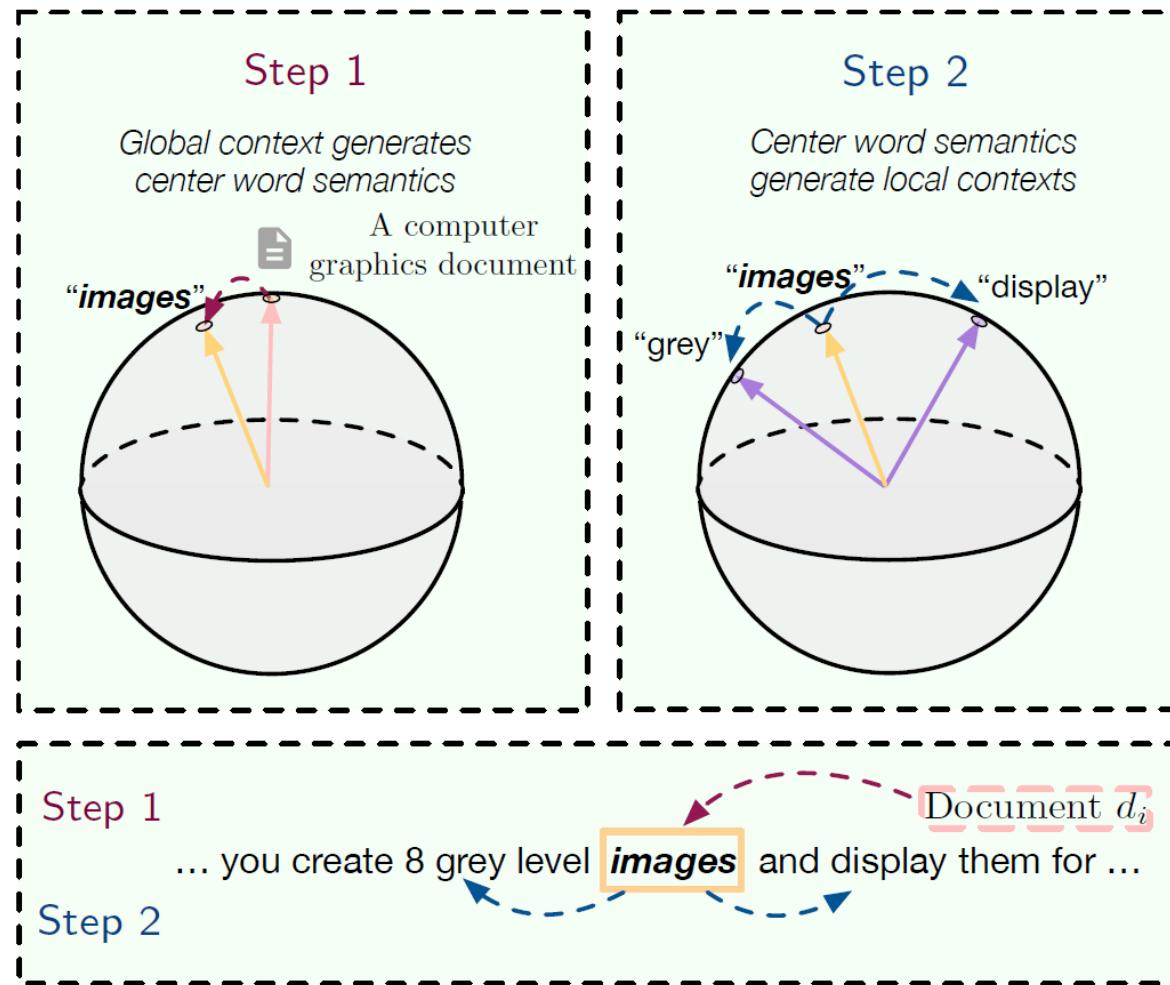
- We prove a theorem connecting the above generative model with a spherical probability distribution:

Theorem 1. When the corpus has infinite vocabulary, *i.e.*, $|V| \rightarrow \infty$, the analytic forms of $p(u | d) \propto \exp(\cos(u, d))$ and $p(v | u) \propto \exp(\cos(v, u))$ are given by the von Mises-Fisher (vMF) distribution with the prior embedding as the mean direction and constant 1 as the concentration parameter, *i.e.*,

$$\lim_{|V| \rightarrow \infty} p(v | u) = \text{vMF}_p(\mathbf{v}; \mathbf{u}, 1), \quad \lim_{|V| \rightarrow \infty} p(u | d) = \text{vMF}_p(\mathbf{u}; \mathbf{d}, 1).$$

Spherical Text Embedding: Illustration

□ Understanding the spherical generative model



Spherical Text Embedding: Objective

- The final generation probability:

$$p(v, u \mid d) = p(v \mid u) \cdot p(u \mid d) = \text{vMF}_p(\mathbf{v}; \mathbf{u}, 1) \cdot \text{vMF}_p(\mathbf{u}; \mathbf{d}, 1)$$

- Maximize the log-probability of a real co-occurred tuple (v, u, d) , while minimize that of a negative sample (v, u', d) , with a max-margin loss:

$$\begin{aligned} \mathcal{L}_{\text{joint}}(\mathbf{u}, \mathbf{v}, \mathbf{d}) &= \max \left(0, m - \log \left(c_p(1) \exp(\cos(\mathbf{v}, \mathbf{u})) \cdot c_p(1) \exp(\cos(\mathbf{u}, \mathbf{d})) \right) \right) \quad \text{Positive Sample} \\ &\quad + \log \left(c_p(1) \exp(\cos(\mathbf{v}, \mathbf{u}')) \cdot c_p(1) \exp(\cos(\mathbf{u}', \mathbf{d})) \right) \quad \text{Negative Sample} \end{aligned}$$
$$= \max (0, m - \cos(\mathbf{v}, \mathbf{u}) - \cos(\mathbf{u}, \mathbf{d}) + \cos(\mathbf{v}, \mathbf{u}') + \cos(\mathbf{u}', \mathbf{d})) ,$$

Optimization on the Sphere

- Riemannian optimization with Riemannian SGD:

- Riemannian gradient:

$$\text{grad } f(\mathbf{x}) := (I - \mathbf{x}\mathbf{x}^\top) \nabla f(\mathbf{x})$$

- Exponential mapping (maps from the tangent plane to the sphere):

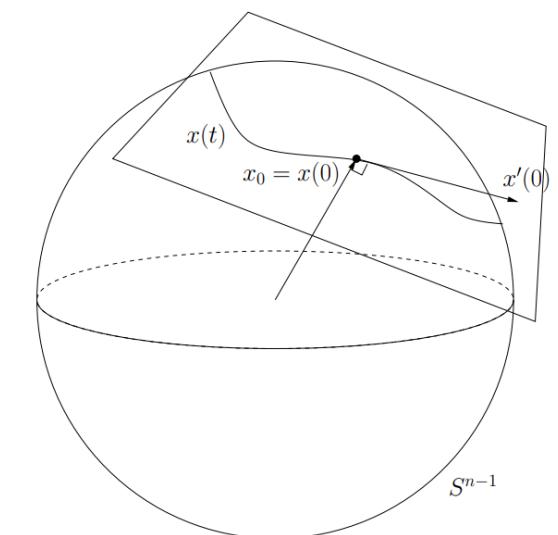
$$\exp_{\mathbf{x}}(z) := \begin{cases} \cos(\|z\|)\mathbf{x} + \sin(\|z\|)\frac{z}{\|z\|}, & z \in T_{\mathbf{x}}\mathbb{S}^{p-1} \setminus \{\mathbf{0}\}, \\ \mathbf{x}, & z = \mathbf{0}. \end{cases}$$

- Riemannian SGD:

$$\mathbf{x}_{t+1} = \exp_{\mathbf{x}_t}(-\eta_t \text{grad } f(\mathbf{x}_t))$$

- Retraction (first-order approximation of the exponential mapping):

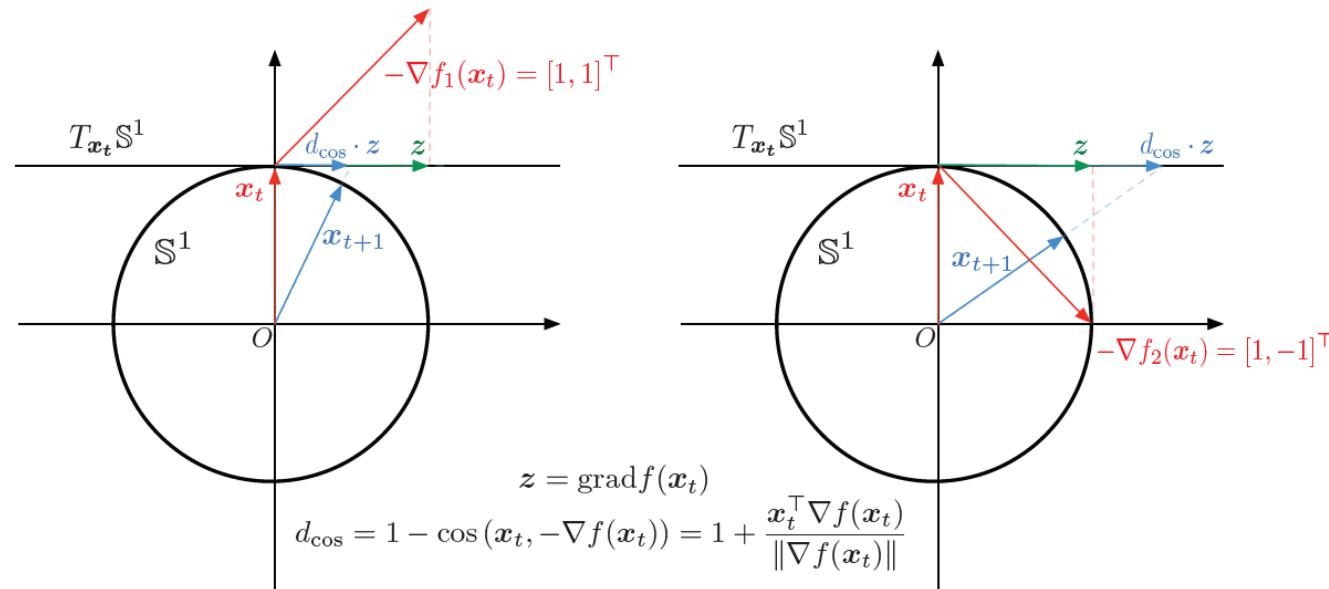
$$R_{\mathbf{x}}(z) := \frac{\mathbf{x} + z}{\|\mathbf{x} + z\|}$$



Optimization on the Sphere

- Training details:

- Incorporate angular distances into Riemannian optimization



- Multiply the Euclidean gradient with its angular distance from the current point

$$x_{t+1} = R_{x_t} \left(-\eta_t \left(1 + \frac{x_t^\top \nabla f(x_t)}{\|\nabla f(x_t)\|} \right) (I - x_t x_t^\top) \nabla f(x_t) \right).$$

Experiments

□ Word similarity results:

Table 1: Spearman rank correlation on word similarity evaluation.

Embedding Space	Model	WordSim353	MEN	SimLex999
Euclidean	Word2Vec	0.711	0.726	0.311
	GloVe	0.598	0.690	0.321
	fastText	0.697	0.722	0.303
	BERT	0.477	0.594	0.287
Poincaré	Poincaré GloVe	0.623	0.652	0.321
Spherical	JoSE	0.739	0.748	0.339

- Why does BERT fall behind on this task?
 - BERT learns contextualized representations, but word similarity is conducted in a context-free manner
 - BERT is optimized on specific pre-training tasks like predicting masked words and sentence relationships, which have no direct relation to word similarity

Experiments

□ Document clustering results:

Table 2: Document clustering evaluation on the 20 Newsgroup dataset.

Embedding	Clus. Alg.	MI	NMI	ARI	Purity
Avg. W2V	K-Means	1.299 ± 0.031	0.445 ± 0.009	0.247 ± 0.008	0.408 ± 0.014
	SK-Means	1.328 ± 0.024	0.453 ± 0.009	0.250 ± 0.008	0.419 ± 0.012
SIF	K-Means	0.893 ± 0.028	0.308 ± 0.009	0.137 ± 0.006	0.285 ± 0.011
	SK-Means	0.958 ± 0.012	0.322 ± 0.004	0.164 ± 0.004	0.331 ± 0.005
BERT	K-Means	0.719 ± 0.013	0.248 ± 0.004	0.100 ± 0.003	0.233 ± 0.005
	SK-Means	0.854 ± 0.022	0.289 ± 0.008	0.127 ± 0.003	0.281 ± 0.010
Doc2Vec	K-Means	1.856 ± 0.020	0.626 ± 0.006	0.469 ± 0.015	0.640 ± 0.016
	SK-Means	1.876 ± 0.020	0.630 ± 0.007	0.494 ± 0.012	0.648 ± 0.017
JoSE	K-Means	1.975 ± 0.026	0.663 ± 0.008	0.556 ± 0.018	0.711 ± 0.020
	SK-Means	1.982 ± 0.034	0.664 ± 0.010	0.568 ± 0.020	0.721 ± 0.029

- Embedding quality is generally more important than clustering algorithms:
- Using spherical K-Means only gives marginal performance boost over K-Means
- JoSE embedding remains optimal regardless of clustering algorithms

Experiments

- Training efficiency:

Table 4: Training time (per iteration) on the latest Wikipedia dump.

Word2Vec	GloVe	fastText	BERT	Poincaré GloVe	JoSE
0.81 hrs	0.85 hrs	2.11 hrs	> 5 days	1.25 hrs	0.73 hrs

- Why is JoSE training efficient?

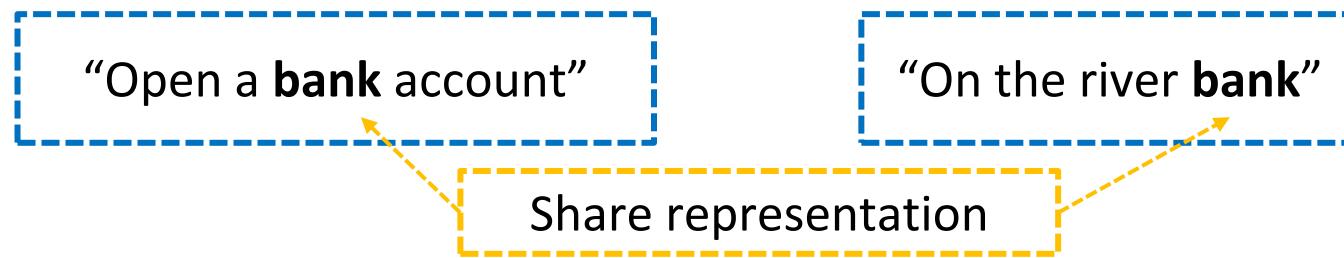
- Other models' objectives contain many non-linear operations (Word2Vec & fastText's objectives involve exponential functions; GloVe's objective involves logarithm functions), while JoSE only has linear terms in the objective

Outline

- Introduction to text representations
- Context-free embeddings
- Deep contextualized embeddings via neural language models 
- Extend unsupervised embeddings to incorporate weak supervision

From Context-Free Embedding to Contextualized Embedding

- ❑ Previous unsupervised word embeddings like Word2Vec and GloVe learn **context-free** word embedding
 - ❑ Each word has one representation regardless of specific contexts it appears in
 - ❑ E.g., “bank” is a polysemy, but only has one representation



- ❑ Deep neural language models overcome this problem by learning **contextualized** word semantics

Outline

- Introduction to text representations
- Context-free embeddings
- Deep contextualized embeddings via neural language models
 - Unidirectional LMs 
 - Bidirectional LMs
 - Applications
- Extend unsupervised embeddings to incorporate weak supervision

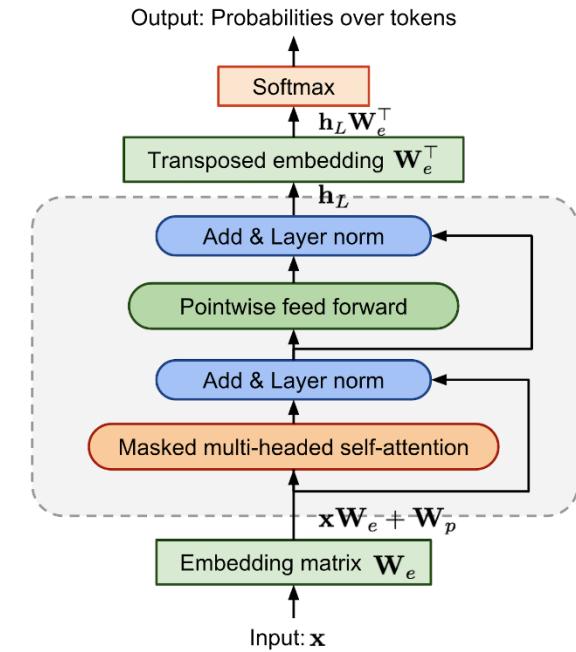
GPT-Style Pre-Training: Introduction

- Generative Pre-Training (GPT [1], GPT-2 [2], GPT-3 [3]):
- Leverage unidirectional context (usually left-to-right) for next token prediction (i.e., language modeling)

k previous tokens as context

$$\mathcal{L}_{LM} = - \sum_i \log p(x_i | x_{i-k}, \dots, x_{i-1})$$

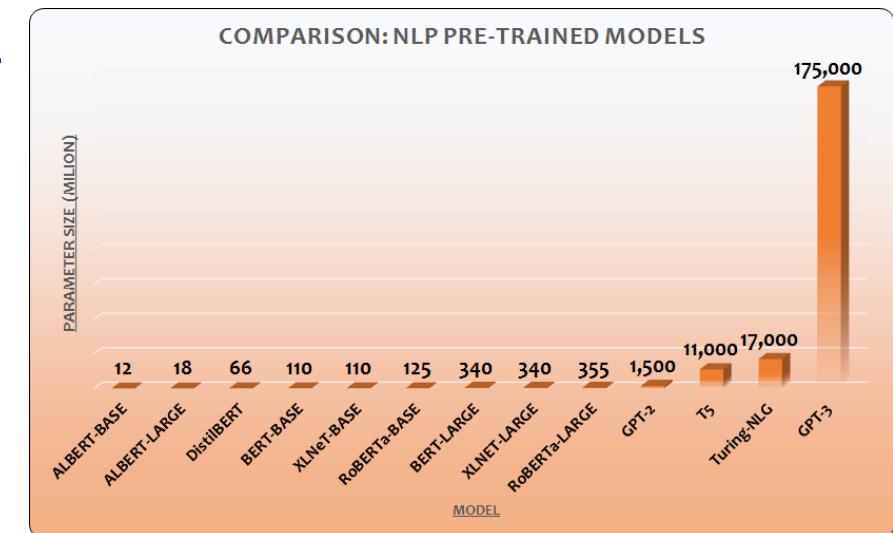
- The Transformer uses unidirectional attention masks (i.e., every token can only attend to previous tokens)



- [1] Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training. OpenAI blog
- [2] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. OpenAI blog, 1(8), 9.
- [3] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. NeurIPS.

GPT-Style Pre-Training: Text Generation

- ❑ Unidirectional LMs are commonly used for text generation tasks (e.g., summarization, translation, ...)
- ❑ They can be very, very large (GPT-3 has 175 Billion parameters!) and have very strong text generation abilities (e.g., generated articles make human evaluators difficult to distinguish from articles written by humans)
- ❑ A demo of real articles vs. generated texts by GPT-2 trained on 10K Nature Papers: <https://stefanzukin.com/enigma/>



GPT-Style Pre-Training: Few-Shot Learning

- ❑ GPT-3 also has strong few-shot learning ability (i.e., without fine-tuning on large task-specific training sets)
- ❑ Generate answers based on natural language descriptions and prompts

The three settings we explore for in-context learning

Zero-shot

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



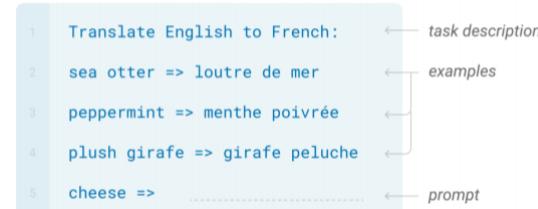
One-shot

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



Few-shot

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



Traditional fine-tuning (not used for GPT-3)

Fine-tuning

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



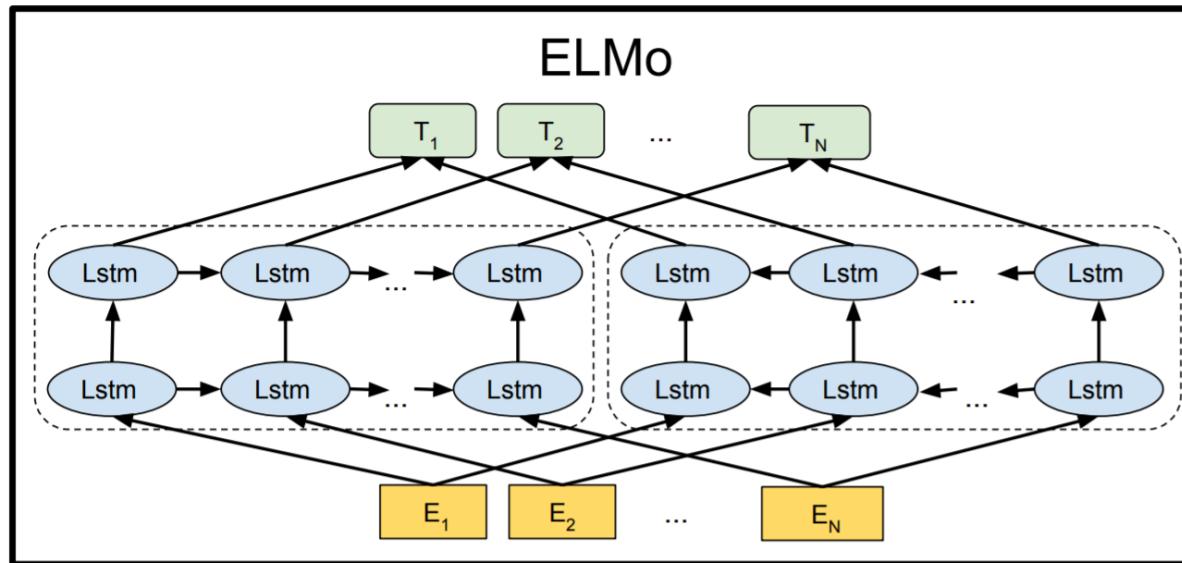
Outline

- Introduction to text representations
- Context-free embeddings
- Deep contextualized embeddings via neural language models
 - Unidirectional LMs
 - Bidirectional LMs
 - Applications
- Extend unsupervised embeddings to incorporate weak supervision



ELMo: Deep contextualized word representations

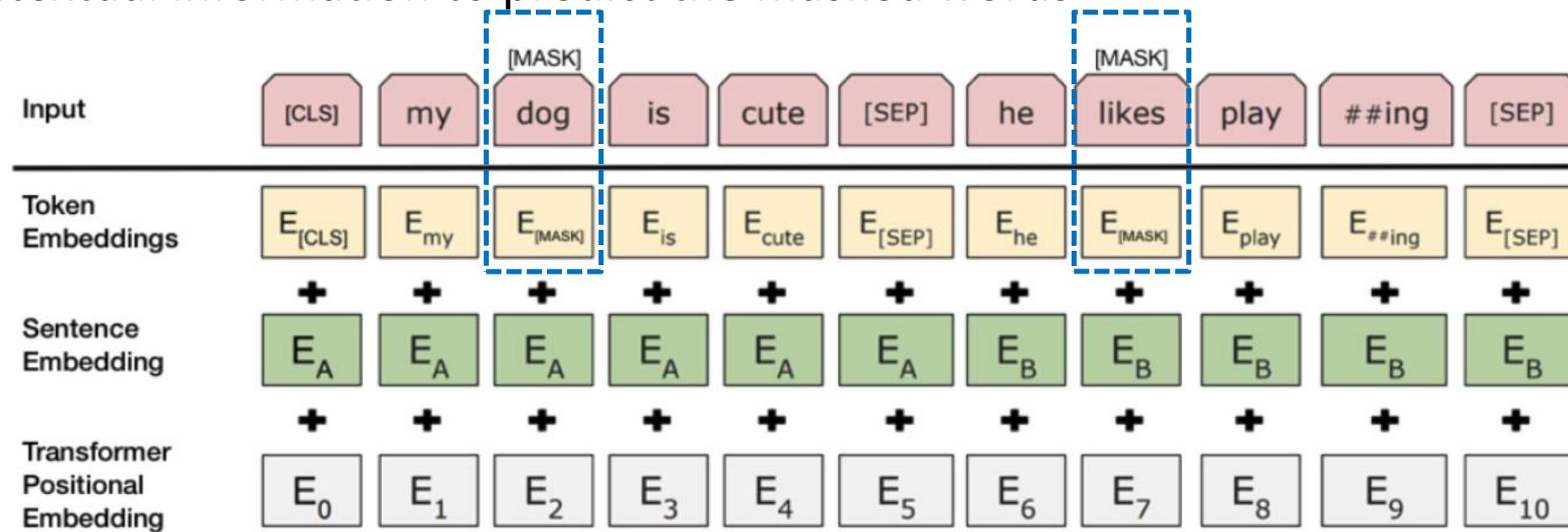
- Word representations are learned functions of the internal states of a deep bi-directional LSTMs
- Results in a pre-trained network that benefits several downstream tasks (e.g., Sentiment analysis, Named entity extraction, Question answering)
- However, left-to-right and right-to-left LSTMs are **independently** trained and concatenated



Peters, M.E., Neumann, M., Iyyer, M., Gardner, M.P., Clark, C., Lee, K., & Zettlemoyer, L.S. (2018). Deep contextualized word representations. NAACL.

BERT: Masked Language Modeling

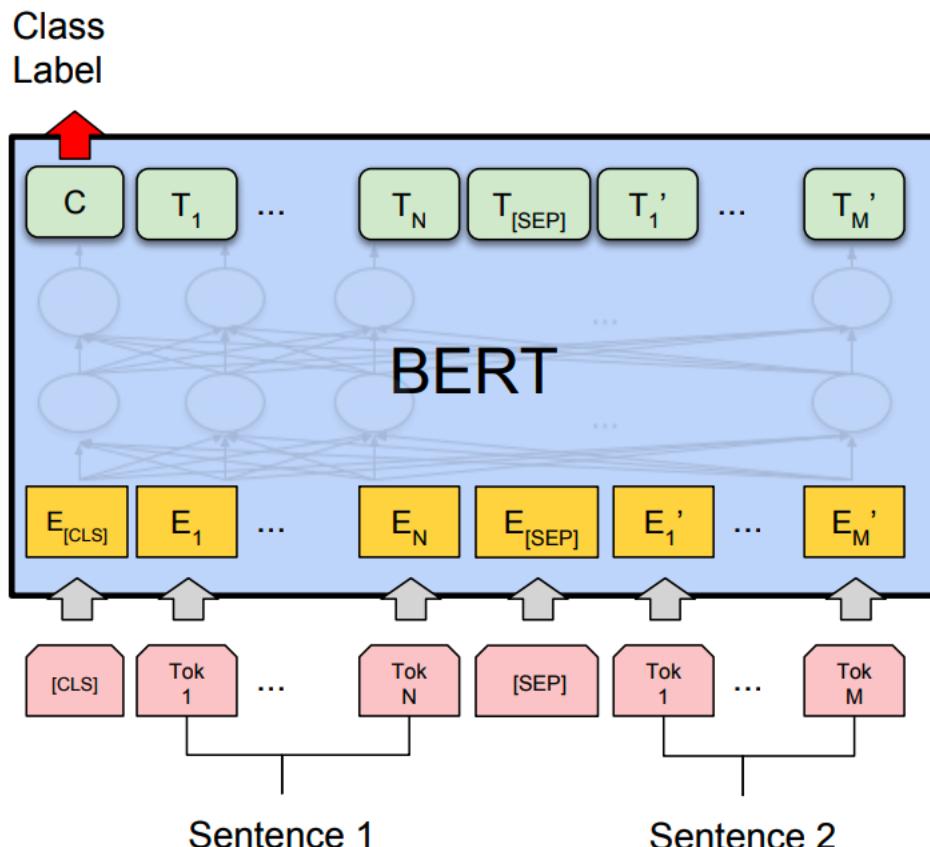
- Bidirectional: BERT leverages a Masked LM learning to introduce **real bidirectionality** training
- Masked LM: With 15% words randomly masked, the model learns bidirectional contextual information to predict the masked words



Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." NAACL (2019).

BERT: Next Sentence Prediction

- Next Sentence Prediction: learn to predict if the second sentence in the pair is the subsequent sentence in the original document



RoBERTa

- ❑ Several simple modifications that make BERT more **effective**:
 - ❑ train the model longer, with bigger batches over more data
 - ❑ remove the next sentence prediction objective
 - ❑ train on longer sequences
 - ❑ dynamically change the masking pattern applied to the training data

Model	data	bsz	steps	SQuAD (v1.1/2.0)	MNLI-m	SST-2
RoBERTa						
with BOOKS + WIKI	16GB	8K	100K	93.6/87.3	89.0	95.3
+ additional data (§3.2)	160GB	8K	100K	94.0/87.7	89.3	95.6
+ pretrain longer	160GB	8K	300K	94.4/88.7	90.0	96.1
+ pretrain even longer	160GB	8K	500K	94.6/89.4	90.2	96.4
BERT _{LARGE}						
with BOOKS + WIKI	13GB	256	1M	90.9/81.8	86.6	93.7
XLNet _{LARGE}						
with BOOKS + WIKI	13GB	256	1M	94.0/87.8	88.4	94.4
+ additional data	126GB	2K	500K	94.5/88.8	89.8	95.6

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). Roberta: A robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692.

ALBERT

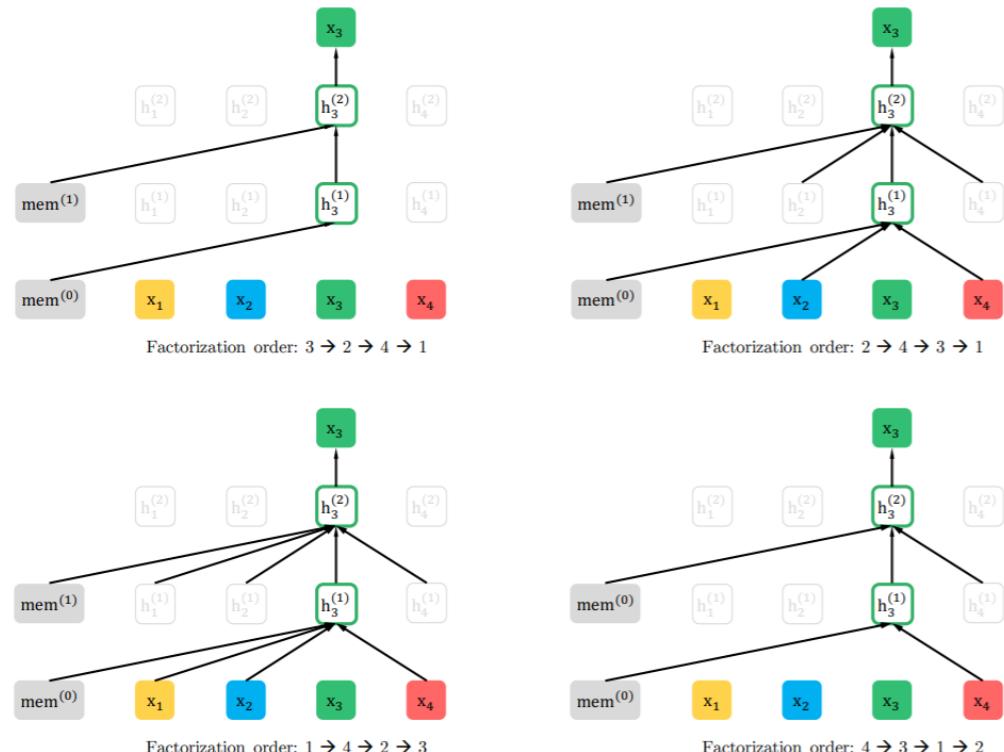
- ❑ Simple modifications that make BERT more **efficient**:
 - ❑ Factorized embedding parameterization: use lower-dimensional token embeddings; project token embeddings to hidden layer dimension
 - ❑ Cross-layer parameter sharing: Share feed-forward network parameters/attention parameters across layers
 - ❑ Inter-sentence coherence loss: change the next sentence prediction task to sentence order prediction

	Model	Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg	Speedup
BERT	base	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3	4.7x
	large	334M	92.2/85.5	85.0/82.2	86.6	93.0	73.9	85.2	1.0
ALBERT	base	12M	89.3/82.3	80.0/77.1	81.6	90.3	64.0	80.1	5.6x
	large	18M	90.6/83.9	82.3/79.4	83.5	91.7	68.5	82.4	1.7x
	xlarge	60M	92.5/86.1	86.1/83.1	86.4	92.4	74.8	85.5	0.6x
	xxlarge	235M	94.1/88.3	88.1/85.1	88.0	95.2	82.3	88.7	0.3x

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2020). Albert: A lite BERT for self-supervised learning of language representations. ICLR.

XLNet: Autoregressive Language Modeling

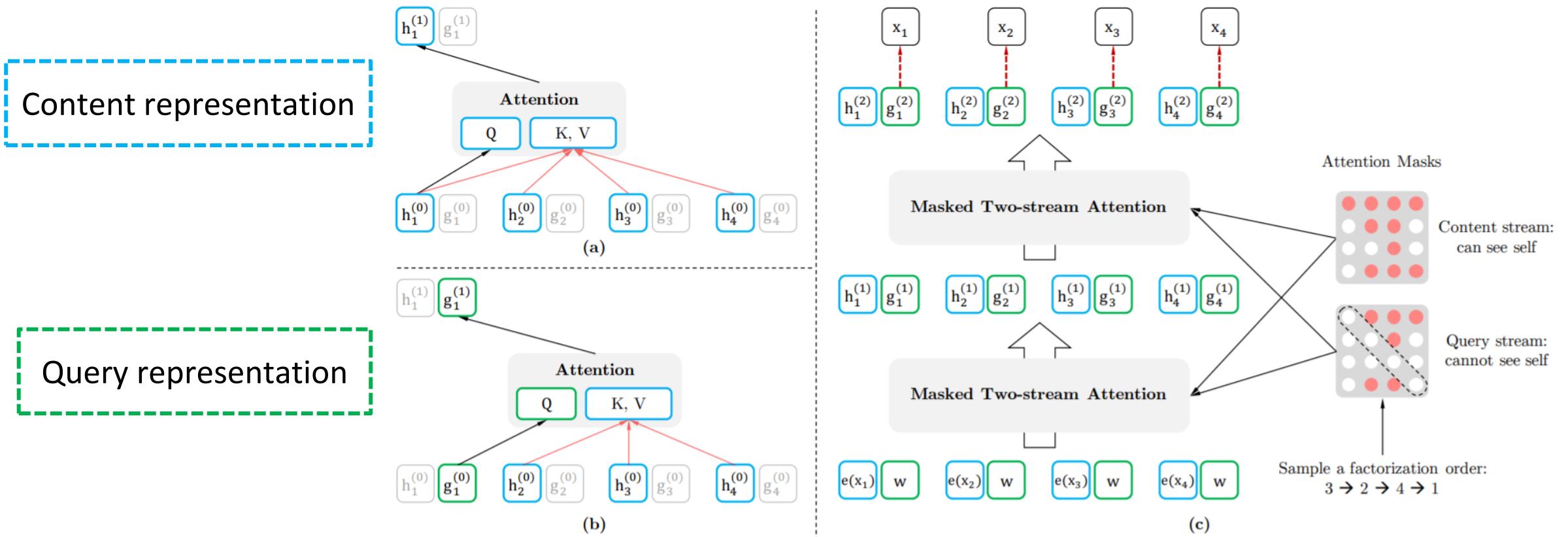
- ❑ Issues with BERT: Masked tokens are predicted independently, and [MASK] token brings discrepancy between pre-training and fine-tuning
- ❑ XLNet uses Permutation Language Modeling



- ❑ Permutes the text sequence and predicts the target word using the remaining words in the sequence
- ❑ Since words in the original sequence are permuted, both forward direction information and backward direction information are leveraged

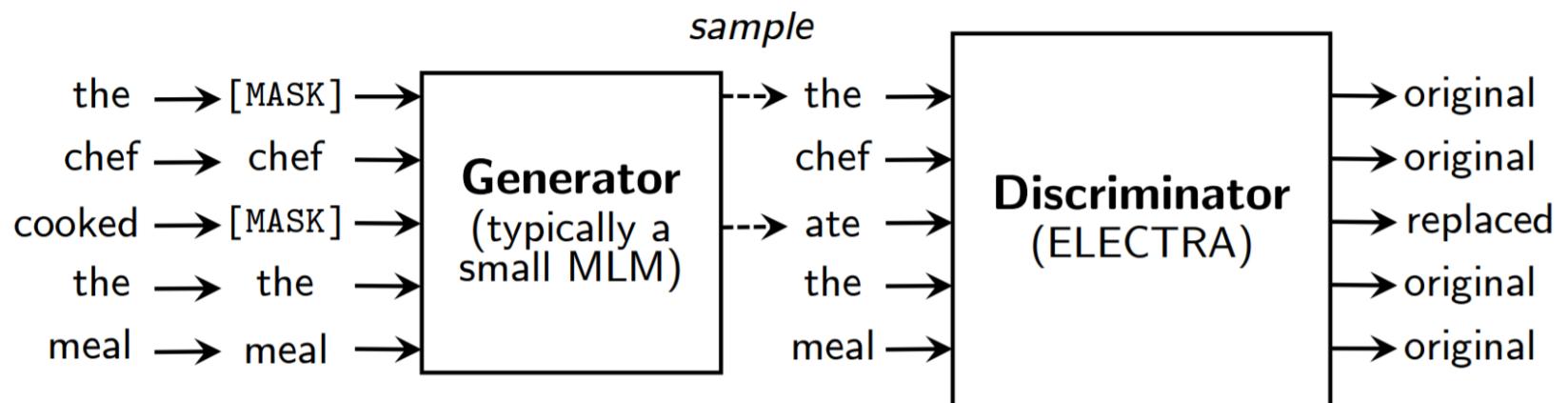
XLNet: Two-Stream Self-Attention

- Content representation: Encodes both token position as well as content
- Query representation: Encodes only token position



ELECTRA

- ❑ Change masked language modeling to a more sample-efficient pre-training task, **replaced token detection**
- ❑ Why more efficient:
 - ❑ Replaced token detection trains on all tokens, instead of just on those that are masked (15%)
 - ❑ The generator trained with MLM is small (parameter size is ~1/10 of discriminator)
 - ❑ The discriminator is trained with a binary classification task, instead of MLM (classification over the entire vocabulary)



Clark, K., Luong, M. T., Le, Q. V., & Manning, C. D. (2020). Electra: Pre-training text encoders as discriminators rather than generators. ICLR.

ELECTRA

- State-of-the-art GLUE (General Language Understanding Evaluation) test performance with the same compute (measured by Floating Point Operations)

Model	Train FLOPs	CoLA	SST	MRPC	STS	QQP	MNLI	QNLI	RTE	WNLI	Avg.*	Score
BERT	1.9e20 (0.06x)	60.5	94.9	85.4	86.5	89.3	86.7	92.7	70.1	65.1	79.8	80.5
RoBERTa	3.2e21 (1.02x)	67.8	96.7	89.8	91.9	90.2	90.8	95.4	88.2	89.0	88.1	88.1
ALBERT	3.1e22 (10x)	69.1	97.1	91.2	92.0	90.5	91.3	–	89.2	91.8	89.0	–
XLNet	3.9e21 (1.26x)	70.2	97.1	90.5	92.6	90.4	90.9	–	88.5	92.5	89.1	–
ELECTRA	3.1e21 (1x)	71.7	97.1	90.7	92.5	90.8	91.3	95.8	89.8	92.5	89.5	89.4

Outline

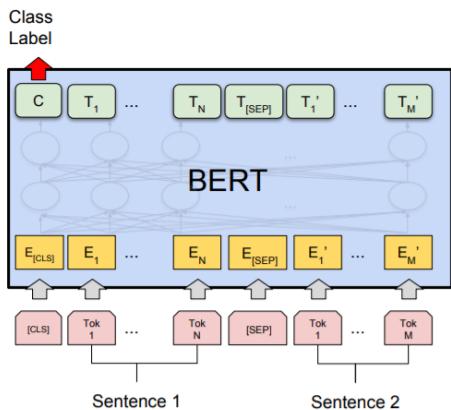
- Introduction to text representations
- Context-free embeddings
- Deep contextualized embeddings via neural language models
 - Unidirectional LMs
 - Bidirectional LMs
 - Applications 
- Extend unsupervised embeddings to incorporate weak supervision

Applications of Pre-Trained Language Models

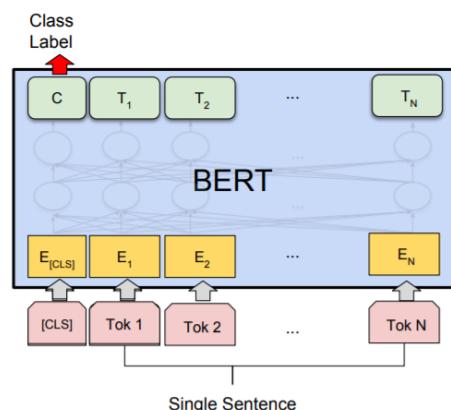
- ❑ Pre-trained language models (PLMs) are usually trained on large-scale general domain corpora to learn generic linguistic features that can be transferred to downstream tasks
- ❑ Common usages of PLMs in downstream tasks
 - ❑ Standard fine-tuning
 - ❑ Prompt-based fine-tuning
 - ❑ Inference without fine-tuning

Standard Fine-Tuning of PLMs

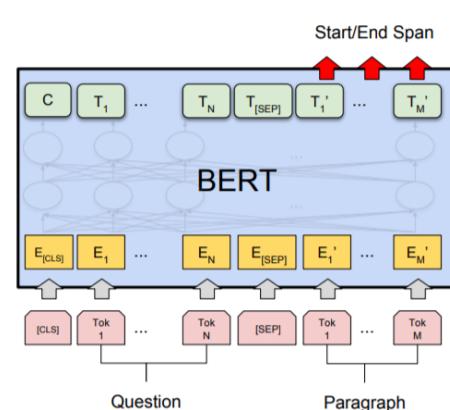
- Add task-specific layers (usually one or two linear layers) on top of the embeddings produced by the PLMs (sequence-level tasks use [CLS] token embeddings; token-level tasks use real token embeddings)
- Task-specific layers and the PLMs are jointly fine-tuned with task-specific training data



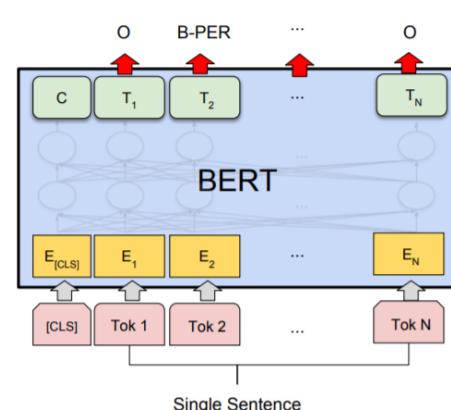
(a) Sentence Pair Classification Tasks:
MNLI, QQP, QNLI, STS-B, MRPC,
RTE, SWAG



(b) Single Sentence Classification Tasks:
SST-2, CoLA



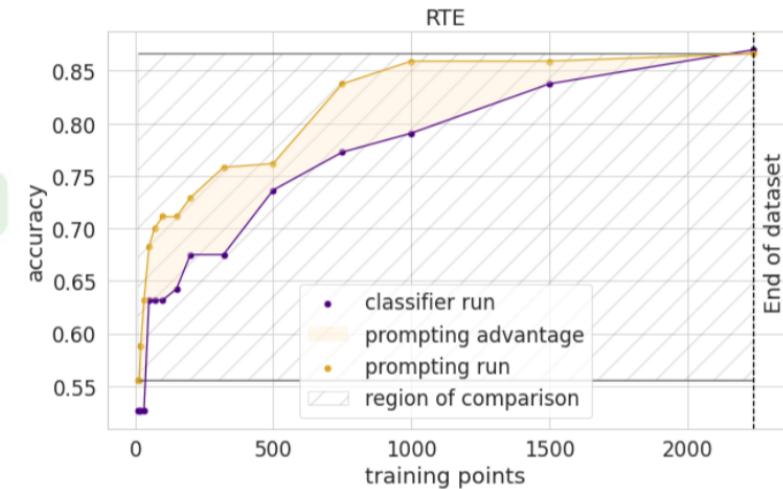
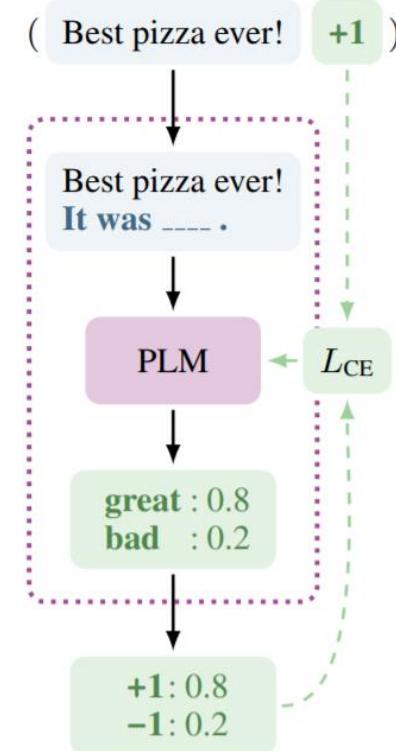
(c) Question Answering Tasks:
SQuAD v1.1



(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

Prompt-Based Fine-Tuning of PLMs

- Task descriptions are created to convert training examples to cloze questions
- Highly resemble the pre-training tasks (MLM) so that pre-training knowledge could be better leveraged
- Better than standard fine-tuning especially for few-shot settings



Schick, T., & Schütze, H. (2021). Exploiting cloze questions for few shot text classification and natural language inference. EACL.

Le Scao, T., & Rush, A. M. (2021). How many data points is a prompt worth? NAACL.

PLMs Inference Without Fine-Tuning

- ❑ Even without any training, knowledge can be extracted from PLMs through cloze patterns
- ❑ PLMs can serve as knowledge bases
 - ❑ Pros: require no schema engineering, and support an open set of queries
 - ❑ Cons: retrieved answers are not guaranteed to be accurate
- ❑ Could be used for unsupervised open-domain QA systems

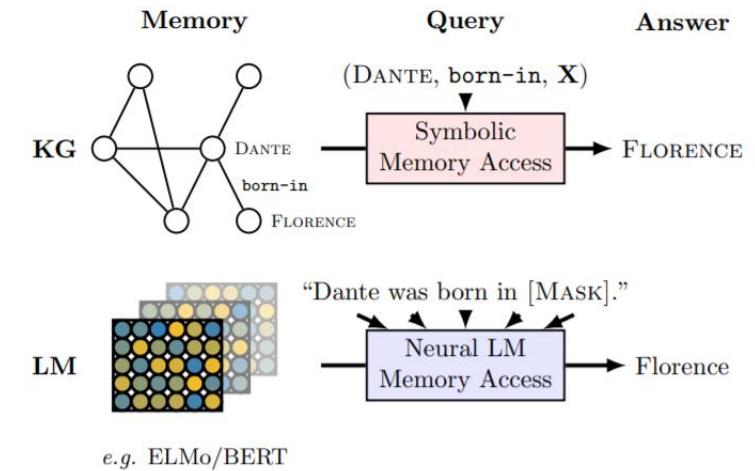


Figure 1: Querying knowledge bases (KB) and language models (LM) for factual knowledge.

Petroni, F., Rocktäschel, T., Lewis, P., Bakhtin, A., Wu, Y., Miller, A. H., & Riedel, S. (2019). Language models as knowledge bases? EMNLP.

Outline

- Introduction to text representations
- Context-free embeddings
- Deep contextualized embeddings via neural language models
- Extend unsupervised embeddings to incorporate weak supervision



From Unsupervised Embedding to Weakly-Supervised Embedding

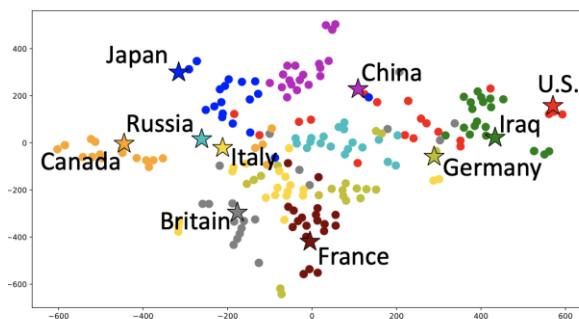
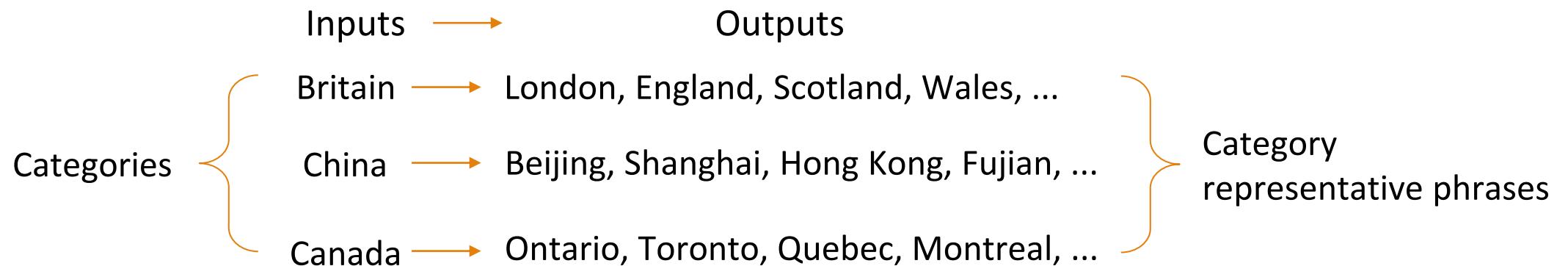
- Text embeddings/language models have strong representation power and can be generalized to many downstream tasks
- However, unsupervised word embeddings are **generic** word representations
 - Not yielding the best performance on downstream tasks (e.g., taxonomy construction, document classification)
 - Reason: Not incorporating **task-specific** information

good	bad
decent	<i>good</i> (x)
great	terrible
tasty	poor
yummy	horrible
<i>bad</i> (x)	awful
alright	<i>alright</i> (x)
fantastic	weird
impressive	frustrating
<i>weak</i> (x)	harsh
<i>disappointing</i> (x)	<i>decent</i> (x)

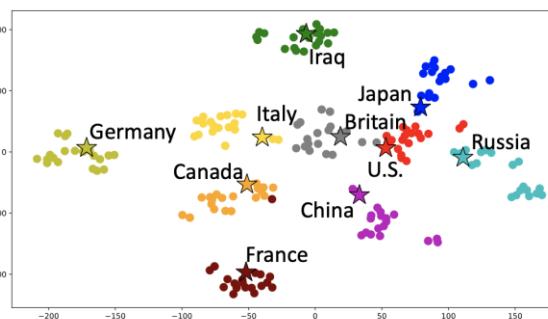
Unsupervised word embedding (Word2Vec) fails to discriminate opposite meaning words

Weakly-Supervised Embeddings for Topic Mining

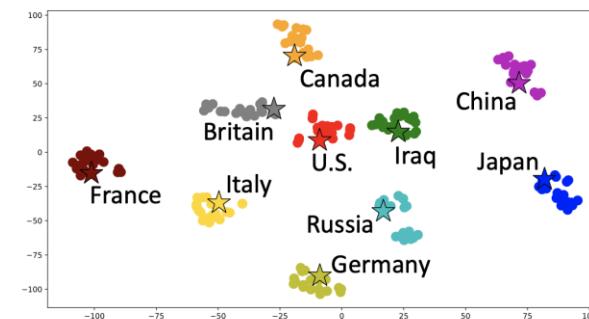
- What if a user is interested in comparative analyses with a specific set of categories (e.g., topics, sentiment, locations ...)?
- Train weakly-supervised (seed word guided) embeddings for discriminative representations w.r.t the categories



(a) Epoch 1



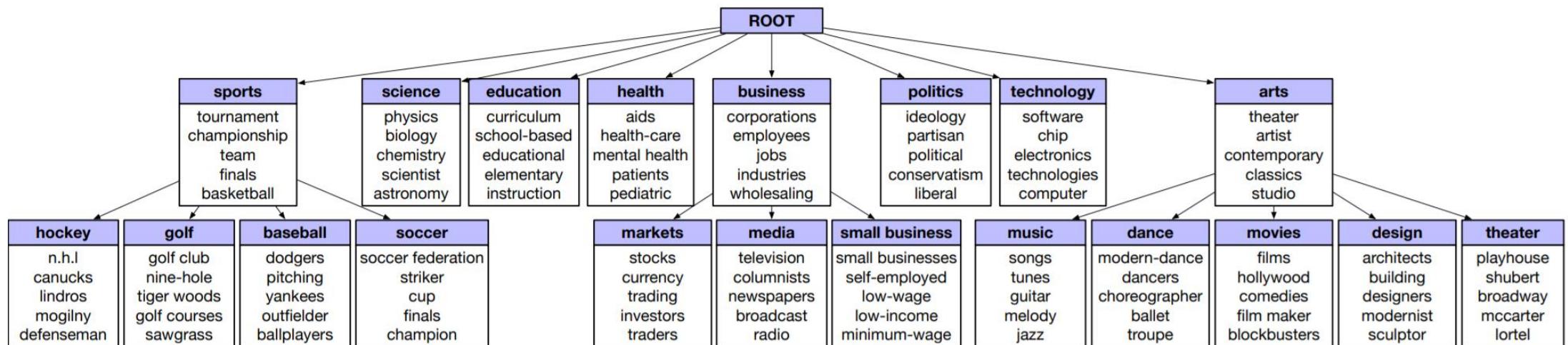
(b) Epoch 3



(c) Epoch 5

Weakly-Supervised Embeddings for (Hierarchical) Topic Mining

- Weakly-supervised embeddings may also leverage given taxonomy structures for coarse-to-fine topic mining
- To be introduced in detail in Part 3 of the tutorial



References

- Abu-El-Haija, S., Perozzi, B., Al-Rfou', R., & Alemi, A.A. (2018). Watch Your Step: Learning Node Embeddings via Graph Attention. NeurIPS.
- Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2016). Enriching Word Vectors with Subword Information. Transactions of the Association for Computational Linguistics, 5, 135-146.
- Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. NeurIPS.
- Clark, K., Luong, M. T., Le, Q. V., & Manning, C. D. (2020). Electra: Pre-training text encoders as discriminators rather than generators. ICLR.
- Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL-HLT.
- Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2020). Albert: A lite bert for self-supervised learning of language representations. ICLR.
- Le Scao, T., & Rush, A. M. (2021). How many data points is a prompt worth? NAACL.
- Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. NIPS.
- Mikolov, T., Chen, K., Corrado, G.S., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. CoRR, abs/1301.3781.

References (Continued)

- Meng, Y., Huang, J., Wang, G., Zhang, C., Zhuang, H., Kaplan, L.M., & Han, J. (2019). Spherical Text Embedding. NeurIPS.
- Nickel, M., & Kiela, D. (2017). Poincaré Embeddings for Learning Hierarchical Representations. NIPS.
- Nickel, M., & Kiela, D. (2018). Learning Continuous Hierarchies in the Lorentz Model of Hyperbolic Geometry. ICML.
- Pennington, J., Socher, R., & Manning, C.D. (2014). Glove: Global Vectors for Word Representation. EMNLP.
- Peters, M.E., Neumann, M., Iyyer, M., Gardner, M.P., Clark, C., Lee, K., & Zettlemoyer, L.S. (2018). Deep contextualized word representations. NAACL.
- Petroni, F., Rocktäschel, T., Lewis, P., Bakhtin, A., Wu, Y., Miller, A. H., & Riedel, S. (2019). Language models as knowledge bases? EMNLP.
- Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training. OpenAI blog.
- Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. OpenAI blog, 1(8), 9.
- Schick, T., & Schütze, H. (2021). Exploiting cloze questions for few shot text classification and natural language inference. EACL.
- Tifrea, A., Bécigneul, G., & Ganea, O. (2019). Poincare Glove: Hyperbolic Word Embeddings. ICLR.
- Turian, J.P., Ratinov, L., & Bengio, Y. (2010). Word Representations: A Simple and General Method for Semi-Supervised Learning. ACL.
- Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., & Le, Q. V. (2019). XLNet: Generalized Autoregressive Pretraining for Language Understanding. NeurIPS.

Q&A