Spherical Text Embedding

Yu Meng¹, Jiaxin Huang¹, Guangyuan Wang¹, Chao Zhang², Honglei Zhuang³, Lance Kaplan⁴ and Jiawei Han¹

University of Illinois at Urbana-Champaign¹, Georgia Institute of Technology², Google Research³, U.S. Army Research Laboratory⁴

Code: https://github.com/yumeng5/Spherical-Text-Embedding
Outline

- Preliminaries
- Motivations & Introduction
- Model: Spherical Text Embedding
- Method: Optimization on the Sphere
- Experiments
- Conclusions
Preliminaries: Text Embedding

- 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)
Preliminaries: Text Embedding

- 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
      \[ \theta \text{ is close to } 0^\circ \]
      \[ \cos(\theta) = 1 \]
    - ball and crocodile are not similar
      \[ \theta \text{ is close to } 90^\circ \]
      \[ \cos(\theta) = 0 \]
    - Rome - Italy
      \[ \theta \text{ is close to } 180^\circ \]
      \[ \cos(\theta) = -1 \]
  - Word clustering (e.g. TaxoGen) is performed on a sphere
  - Better document clustering performances when embeddings are normalized and spherical clustering algorithms are used
Outline

- Preliminaries
- Motivations & Introduction
- Model: Spherical Text Embedding
- Method: Optimization on the Sphere
- Experiments
- Conclusions
Motivations

- Issues with previous word 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.)
Motivations

- What is the consequence of the inconsistency between word embedding training and usage space?

  - 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 actually used in applications

\[
p(w_O | w_I) = \frac{\exp\left( v_{w_O}^T v_{w_I} \right)}{\sum_{w=1}^{W} \exp\left( v_{w}^T v_{w_I} \right)}
\]

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

\[
s(w, c) = \sum_{g \in \mathcal{G}_w} z_g^T v_c
\]

Word2Vec          GloVe          fastText
Motivations

- What is the consequence of the inconsistency between word embedding training and usage space?
  - The objective we optimize during training is not really the one we use
  - E.g. Consider two pairs of words, A: lover-quarrel; B: rock-jazz. Pair B has higher ground truth similarity than pair A in WordSim353 (a benchmark testset)
  - Word2Vec assigns higher dot product to pair B, but its cosine similarity is still smaller than pair A

<table>
<thead>
<tr>
<th>Metrics</th>
<th>A: lover-quarrel</th>
<th>B: rock-jazz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dot Product</td>
<td>5.284</td>
<td>&lt; 6.287</td>
</tr>
<tr>
<td>Cosine Similarity</td>
<td>0.637</td>
<td>&gt; 0.628</td>
</tr>
</tbody>
</table>
Motivations

- 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 …
Introduction

- In this work, we propose “Spherical Text Embedding”
  - "Spherical": Embeddings are trained on the unit sphere, where vector norms are ignored and directional similarity is directly optimized
  - "Text Embedding": Instead of training word embeddings only, we jointly train paragraph (document) embeddings with word embeddings to capture the local and global contexts in text embedding

Contributions:

- A spherical generative model that jointly exploits word-word (local) and word-paragraph (global) co-occurrence statistics
- An efficient optimization algorithm in the spherical space with convergence guarantee
- State-of-the-art performances on various text embedding applications
Outline

- Preliminaries
- Motivations & Introduction
- Model: Spherical Text Embedding
- Method: Optimization on the Sphere
- Experiments
- Conclusions
Model: Spherical Text Embedding

- 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:

\[
p(u \mid d) \propto \exp(\cos(u, d)) \quad \text{and} \quad p(v \mid u) \propto \exp(\cos(v, u))
\]
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., $|V| \to \infty$, the analytic forms of $p(u \mid d) \propto \exp(\cos(u, d))$ and $p(v \mid u) \propto \exp(\cos(v, u))$ 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.,

$$
\lim_{|V| \to \infty} p(v \mid u) = \text{vMF}_p(v; u, 1), \quad \lim_{|V| \to \infty} p(u \mid d) = \text{vMF}_p(u; d, 1).
$$
Model: Spherical Text Embedding

- Understanding the spherical generative model

**Step 1**
Global context generates center word semantics

**Step 2**
Center word semantics generate local contexts

... you create 8 grey level images and display them for ...
Training objective:

The final generation probability:

\[ p(v, u | d) = p(v | u) \cdot p(u | d) = \text{vMF}_p(v; u, 1) \cdot \text{vMF}_p(u; d, 1) \]

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:

\[
L_{\text{joint}}(u, v, d) = \max \left( 0, m - \log \left( c_p(1) \exp(\cos(v, u)) \cdot c_p(1) \exp(\cos(u, d)) \right) \right)
\]

\[
+ \log \left( c_p(1) \exp(\cos(v, u')) \cdot c_p(1) \exp(\cos(u', d)) \right)
\]

\[
= \max \left( 0, m - \cos(v, u) - \cos(u, d) + \cos(v, u') + \cos(u', d) \right)
\]
Outline

- Preliminaries
- Motivations & Introduction
- Model: Spherical Text Embedding
- Method: Optimization on the Sphere
- Experiments
- Conclusions
The constrained optimization problem:

\[
\min_{\Theta} L_{\text{joint}}(\Theta) \quad \text{s.t.} \quad \forall \theta \in \Theta : \|\theta\| = 1 \quad \Theta = \{u_i\}_{i=1}^{|V|} \cup \{v_i\}_{i=1}^{|V|} \cup \{d_i\}_{i=1}^{|D|}
\]

Challenge: Parameters must be always updated on the sphere, but Euclidean optimization methods (e.g. SGD) are not constrained on a curvature space.

Need to consider the nature of the spherical space.
Method: Optimization on the Sphere

- Riemannian manifold:
  - The sphere is a Riemannian manifold with constant positive curvature
Method: Optimization on the Sphere

- Riemannian optimization with Riemannian SGD:
  - Riemannian gradient:
    \[
    \text{grad} f(x) := (I - xx^T) \nabla f(x)
    \]
  - Exponential mapping (maps from the tangent plane to the sphere):
    \[
    \exp_x(z) := \begin{cases} 
    \cos(\|z\|) x + \sin(\|z\|) \frac{z}{\|z\|}, & z \in T_xS^{p-1}\setminus\{0\}, \\
    x, & z = 0.
    \end{cases}
    \]
  - Riemannian SGD:
    \[
    x_{t+1} = \exp_x(-\eta_t \text{grad} f(x_t))
    \]
  - Retraction (first-order approximation of the exponential mapping):
    \[
    R_x(z) := \frac{x + z}{\|x + z\|}
    \]
Method: Optimization on the Sphere

- Training details:
  - Incorporate angular distances into Riemannian optimization

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

\[
x_{t+1} = R_{x_t} \left( -\eta_t \left( 1 + \frac{x_t^T \nabla f(x_t)}{\|\nabla f(x_t)\|} \right) (I - x_t x_t^T) \nabla f(x_t) \right).
\]
Method: Optimization on the Sphere

- Convergence guarantee of the optimization procedure:

**Theorem 2.** When the update rule given by Equation (7) is applied to $\mathcal{L}(x)$, and the learning rate satisfies the usual condition in stochastic approximation, i.e., $\sum_t \eta_t^2 < \infty$ and $\sum_t \eta_t = \infty$, $x$ converges almost surely to a critical point $x^*$ and grad $\mathcal{L}(x)$ converges almost surely to 0, i.e.,

$$\Pr \left( \lim_{t \to \infty} \mathcal{L}(x_t) = \mathcal{L}(x^*) \right) = 1, \quad \Pr \left( \lim_{t \to \infty} \text{grad} \mathcal{L}(x_t) = 0 \right) = 1.$$
Outline

- Preliminaries
- Motivations & Introduction
- Model: Spherical Text Embedding
- Method: Optimization on the Sphere
- Experiments
- Conclusions
Experiments

- **Word similarity:**
  - Train 100-d word embedding on the latest Wikipedia dump (~13G)
  - Compute embedding **cosine similarity** between word pairs to obtain a ranking of similarity
  - Benchmark datasets contain human rated similarity scores
  - The more similar the two rankings are, the better embedding reflects human thoughts
  - **Spearman’s rank correlation** is used to measure the ranking similarity
Experiments

- **Word similarity baselines:**
  - Euclidian Space:
  - Word2Vec: Distributed representations of words and phrases and their compositionality. In NIPS, 2013
  - GloVe: Glove: Global vectors for word representation. In EMNLP, 2014
  - fastText: Enriching word vectors with subword information. TACL, 2017
  - BERT: Pre-training of deep bidirectional transformers for language understanding. In NAACL, 2019
  - Poincare Space:
  - Poincaré glove: Hyperbolic word embeddings. In ICLR, 2019
  - Spherical Space:
  - JoSE (our full model)
Experiments

- Word similarity results:

<table>
<thead>
<tr>
<th>Embedding Space</th>
<th>Model</th>
<th>WordSim353</th>
<th>MEN</th>
<th>SimLex999</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euclidean</td>
<td>Word2Vec</td>
<td>0.711</td>
<td>0.726</td>
<td>0.311</td>
</tr>
<tr>
<td></td>
<td>GloVe</td>
<td>0.598</td>
<td>0.690</td>
<td>0.321</td>
</tr>
<tr>
<td></td>
<td>fastText</td>
<td>0.697</td>
<td>0.722</td>
<td>0.303</td>
</tr>
<tr>
<td></td>
<td>BERT</td>
<td>0.477</td>
<td>0.594</td>
<td>0.287</td>
</tr>
<tr>
<td>Poincaré</td>
<td>Poincaré GloVe</td>
<td>0.623</td>
<td>0.652</td>
<td>0.321</td>
</tr>
<tr>
<td>Spherical</td>
<td>JoSE</td>
<td>0.739</td>
<td>0.748</td>
<td>0.339</td>
</tr>
</tbody>
</table>

- 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 downstream tasks like predicting masked words and sentence relationships, which have no direct relation to word similarity
Experiments

- Document Clustering:
  - Train document embedding on 20News dataset (20 classes)
  - Perform K-means and Spherical K-means (SK-means)
  - Metrics: Mutual Information (MI), Normalized Mutual Information (NMI), Adjusted Rand Index (ARI), and Purity
  - Run clustering 10 times and report the above metrics with mean and standard deviation
Experiments

- Document clustering baselines:
  - Euclidian Space:
    - Avg. Word2Vec: Use the averaged word embedding of Word2Vec as document embedding
    - SIF: Simple but tough-to-beat baseline for sentence embeddings. In ICLR, 2017
    - BERT: Pre-training of deep bidirectional transformers for language understanding. In NAACL, 2019
    - Doc2Vec: Distributed representations of sentences and documents. In ICML, 2014
  - Spherical Space:
    - JoSE (our full model)
Document clustering results:

- 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

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

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

- Document Classification:
  - Train document embedding on 20News (20 classes) and Movie review (2 classes) dataset
  - Perform k-NN classification (k=3)
  - Metrics: Macro-F1 & Micro-F1
  - Baselines: Same with document clustering
Experiments

- Document classification results:

<table>
<thead>
<tr>
<th>Embedding</th>
<th>20 Newsgroup</th>
<th>Movie Review</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Macro-F1</td>
<td>Micro-F1</td>
</tr>
<tr>
<td>Avg. W2V</td>
<td>0.630</td>
<td>0.631</td>
</tr>
<tr>
<td>SIF</td>
<td>0.552</td>
<td>0.549</td>
</tr>
<tr>
<td>BERT</td>
<td>0.380</td>
<td>0.371</td>
</tr>
<tr>
<td>Doc2Vec</td>
<td>0.648</td>
<td>0.645</td>
</tr>
<tr>
<td>JoSE</td>
<td><strong>0.703</strong></td>
<td><strong>0.707</strong></td>
</tr>
</tbody>
</table>

- We observe similar comparison results with \( k \) from [1, 10]

- \( k \)-NN is non-parametric and directly reflect how well the topology of the embedding space captures document-level semantics
Training efficiency:

<table>
<thead>
<tr>
<th></th>
<th>Word2Vec</th>
<th>GloVe</th>
<th>fastText</th>
<th>BERT</th>
<th>Poincaré GloVe</th>
<th>JoSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training time</td>
<td>0.81 hrs</td>
<td>0.85 hrs</td>
<td>2.11 hrs</td>
<td>&gt; 5 days</td>
<td>1.25 hrs</td>
<td>0.73 hrs</td>
</tr>
</tbody>
</table>

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

- Preliminaries
- Motivations & Introduction
- Model: Spherical Text Embedding
- Method: Optimization on the Sphere
- Experiments
- Conclusions
In this work, we introduce a spherical text embedding framework

- Address the discrepancy between the training procedure and the practical usage of text embedding
- Introduce a spherical generative model to jointly learn word and paragraph embedding
- Develop an efficient Riemannian optimization method to train text embedding on the unit hypersphere
- Achieves state-of-the-art performances on common text embedding applications

Future work:

- Exploit spherical embedding space for other tasks like lexical entailment
- Incorporate other signals such as subword information into spherical text embedding
- Benefit other supervised tasks: Word embedding is commonly used as the first layer in DNN
  - Add norm constraints to word embedding layer
  - Apply Riemannian optimization when fine-tuning the word embedding layer
Thanks!