Learning Embeddings into Entropic Wasserstein Spaces

Charlie Frohner1,2, Farzaneh Mirzazadeh2,3, Justin Solomon1,2
1 MIT CSAIL, 2 MIT-IBM Watson AI Laboratory, 3 IBM Research

Summary

- Wasserstein spaces can embed a wider variety of semantic structures than Euclidean spaces.
- Discrete Wasserstein embeddings can be learned via gradient descent on the Sinkhorn divergence.
- Learned embeddings represent complex networks with low distortion.
- Wasserstein word embeddings achieve competitive performance with smaller data than previous methods.
- Wasserstein embeddings can be directly visualized, unlike other high-dimensional embeddings.

Entropic Wasserstein Spaces

Wasserstein metric: Measures distance between probability measures \( \mu \) and \( \nu \).
- Defined by an optimal transport problem:
  \[
  W_p(\mu, \nu) = \inf_{\gamma \in \Pi(\mu, \nu)} \int_X d(x, y) \gamma(dx, dy)
  \]
  with \( d: X \times X \to \mathbb{R} \), the distance in the underlying domain and \( \Pi(\mu, \nu) \) the set of joint distributions having marginals \( \mu \) and \( \nu \).
- Measures the minimum cost of transporting the mass in \( \mu \) to match \( \nu \).

Discrete Wasserstein metric: The case where \( \mu \) and \( \nu \) are both discrete,
\[
\rho = \sum_{i} w_i \rho_i
\]
where
\[
W_p(\mu, \nu)^p = \min_{T \in \{0, 1\}^{|X|}} \sum \rho_i (T(i) - v_i)^p
\]
with \( \rho_i = \sum_{j \in X} \rho_{ij} \) the matrix of pairwise ground metric distances, with \( T \) a mapping of mass transported between \( \mu \) and \( \nu \).

Sinkhorn divergence: Add an entropic regularizer,
\[
W_p(\mu, \nu)^p = \frac{1}{\lambda} \log (\text{det}(\nabla^2 T)) + \lambda \left( T \mu - \nu \right)
\]
More efficient to compute.
- Gradients are available from automatic differentiation (GCJ 18).

What can we embed in theory?

An embedding of metric space, \( A \to B \), is a map \( \phi : A \to B \) that approximately preserves distances in the sense that the distortion is small:
\[
\delta_d(A, \phi(A), \phi(B)) = \max_{x, y \in A} \frac{d(x, y)}{d(\phi(x), \phi(y))} \leq C \delta_d(A, B)
\]
for some universal constants \( C > 0 \).

Euclidean spaces have limited capacity: Many common spaces do not embed into Euclidean space, including \( L^1 \) and hyperbolic spaces.
- This limits the semantic structures that can be represented, such as hierarchies.

Wasserstein spaces are very large: Many spaces embed into Wasserstein spaces, even when the converse is not true.
- E.g., \( W_p(\mu) \) embeds \( \mathbb{R}^n \) for any metric space \( A \).

Some Wasserstein spaces are universal: They can embed arbitary metrics over finite spaces.
- E.g., \( W_1(\ell_1^k) \).
- Open question: Is \( W_p(\mathbb{R}^k) \) universal for some \( k < +\infty \)?

ANN ’15 \( W_p(\mathbb{R}^k) \) embeds the \( \ell_p^k \) power of any metric on a finite space, \( p > 1 \).

Representational capacity: complex networks

Wasserstein embeddings can be directly visualized, as each embedded element is a probability distribution over a potentially low-dimensional space.
- This is unlike other high-dimensional embeddings, which require a dimensionality-reduction method such as t-SNE for visualization.

Figure 3 (above) shows learned word embeddings.
- We apply kernel density estimation to the discrete embedded distribution.
- We threshold the density and show the upper level set.
- The opacity within the level set shows the density.

Direct visualization can be useful for debugging a learned embedding.
- Figure 3c shows synonyms of the word “kind.”
- Figure 3d shows that the network actually learned that “Nasa” is a city in France.

References


Learning an embedding

Goal: learn a map \( \phi : X \to W_p(\mathbb{R}^k) \) that encodes a target relationship \( r : X \times X \to \mathbb{R} \) in the Wasserstein distance \( W_p(\phi(x), \phi(y)) \).
- \( X \) is a collection of objects (words, images, symbols, etc.).
- E.g. metric embedding: \( r \) is a distance metric and we want \( W_p(\phi(x), \phi(y)) \approx r(x, y) \).
- E.g. graph embedding: \( r \) is the adjacency relation between vertices; neighborhoods in \( W_p(X) \) should correspond to graph adjacency.
- E.g. word embedding: \( r \) is semantic similarity between words; small \( W_p(\phi(x), \phi(y)) \) should correspond to high semantic similarity.

Problem: Given samples \( \{(x_i, y_i, r_i)\}_{i=1}^n \subset X \times X \times \mathbb{R} \), learn \( \phi \) via
\[
\phi = \arg \min_{\phi} \frac{1}{n} \sum \left[ W_p^2(\phi(x_i), \phi(y_i)) - r_i \right]^2
\]
with \( L \) a problem-specific loss function. Use gradient descent.

Wasserstein space definitions

CTdKrevl 2015

Wasserstein spaces are defined as a metric space with a probability measure on the space.

- Wasserstein space: A metric space \( (X, d) \) with a probability measure \( \mu \) on \( X \).
- Wasserstein distance: \( W_p(\mu, \nu) \) measures the cost of transporting mass from \( \mu \) to \( \nu \).

Wasserstein spaces are useful because they allow the embedding of complex structures.
- Discrete structures can be embedded with low distortion.
- Wasserstein spaces are universal, meaning they can embed any metric space.

Wasserstein embeddings are visualized using techniques such as t-SNE or neural networks.
- t-SNE is a dimensionality-reduction technique for visualizing high-dimensional data.

Figure 3 shows the learned word embeddings.
- We apply kernel density estimation to the discrete embedded distribution.

Direct visualization can be useful for debugging a learned embedding.
- Figure 3c shows synonyms of the word “kind.”
- Figure 3d shows the network actually learned that “Nasa” is a city in France.

Wasserstein spaces have applications in machine learning and statistical inference.
- They allow for the modeling of uncertainty in data.

Table 2: Performance on a number of similarity benchmarks when dimensionality of point clouds increase given a fixed total number of parameters. The middle block shows the performance of the proposed method. The right block shows the performance of baselines. The training corpus size when known appears below each model name.

<table>
<thead>
<tr>
<th>Task Name</th>
<th>( k )</th>
<th>Points</th>
<th>( W_1(\mathbb{R}^k) )</th>
<th>( W_2(\mathbb{R}^k) )</th>
<th>( W_3(\mathbb{R}^k) )</th>
<th>( R )</th>
<th>M</th>
<th>S</th>
<th>G</th>
<th>H</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>Big-65</td>
<td>65</td>
<td>0.32</td>
<td>0.07</td>
<td>0.83</td>
<td>0.27</td>
<td>0.03</td>
<td>0.48</td>
<td>0.54</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Weis-115</td>
<td>115</td>
<td>0.33</td>
<td>0.12</td>
<td>0.44</td>
<td>0.18</td>
<td>0.09</td>
<td>0.50</td>
<td>0.69</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Krevl</td>
<td>k</td>
<td>0.33</td>
<td>0.12</td>
<td>0.44</td>
<td>0.18</td>
<td>0.09</td>
<td>0.50</td>
<td>0.69</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MO-30</td>
<td>30</td>
<td>0.29</td>
<td>0.12</td>
<td>0.44</td>
<td>0.18</td>
<td>0.09</td>
<td>0.50</td>
<td>0.69</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ratn-Word</td>
<td>1024</td>
<td>0.06</td>
<td>0.22</td>
<td>0.10</td>
<td>0.29</td>
<td>0.15</td>
<td>0.59</td>
<td>0.89</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MR99</td>
<td>99</td>
<td>0.30</td>
<td>0.12</td>
<td>0.44</td>
<td>0.18</td>
<td>0.09</td>
<td>0.50</td>
<td>0.69</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTask-267</td>
<td>267</td>
<td>0.40</td>
<td>0.12</td>
<td>0.44</td>
<td>0.18</td>
<td>0.09</td>
<td>0.50</td>
<td>0.69</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTask-771</td>
<td>771</td>
<td>0.13</td>
<td>0.23</td>
<td>0.45</td>
<td>0.18</td>
<td>0.09</td>
<td>0.50</td>
<td>0.69</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SNAP-999</td>
<td>999</td>
<td>0.09</td>
<td>0.05</td>
<td>0.35</td>
<td>0.23</td>
<td>0.01</td>
<td>0.30</td>
<td>0.51</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SNAP-144</td>
<td>144</td>
<td>0.03</td>
<td>0.05</td>
<td>0.35</td>
<td>0.23</td>
<td>0.01</td>
<td>0.30</td>
<td>0.51</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Nearest neighbors in the embedding space.