**IS WASSERSTEIN ALL YOU NEED?**

Anonymous authors
Paper under double-blind review

**ABSTRACT**

We propose a unified framework for building unsupervised representations of entities and their compositions, by viewing each entity as a histogram over its contexts. This enables us to take advantage of *optimal transport* and construct representations that effectively harness the geometry of the underlying space containing the contexts. Our method captures uncertainty via modelling the entities as distributions and simultaneously provides interpretability with the optimal transport map, hence giving a novel perspective for building rich and powerful feature representations. As a guiding example, we formulate unsupervised representations for text, and demonstrate it on tasks such as sentence similarity and word entailment detection. Empirical results show strong advantages gained through the proposed framework. This approach can potentially be used for any unsupervised or supervised problem (on text or other modalities) with a co-occurrence structure, such as any sequence data. The key tools at the core of this framework are Wasserstein distances and Wasserstein barycenters, hence raising the question from our title.

1 **INTRODUCTION**

One of the main driving factors behind the recent successes in machine learning and natural language processing has been the development of better representation methods for data modalities. Examples include continuous vector representations for language (Mikolov et al., 2013; Pennington et al., 2014), Convolutional Neural Network (CNN) based text representations (Collobert & Weston, 2008; Kim, 2014; Kalchbrenner et al., 2014; Severyn & Moschitti, 2015; Deriu et al., 2017), or via other neural architectures such as RNNs, LSTMs (Hochreiter & Schmidhuber, 1997; Kiros et al., 2015), all sharing one central idea – to map input entities to dense vector embeddings lying in a low-dimensional latent space where the semantics of the inputs are preserved.

While these existing methods directly represent each entity of interest (e.g., a word) as a single point in space (i.e., its embedding vector), we here propose a fundamentally different approach. Starting from co-occurrence information of the entities of interests and their contexts (e.g. context words or entities), we leverage embeddings of the contexts instead of the original entities. So instead of a single point per entity, our representation is given by the histogram of its contexts, each of which itself is represented as a point in a suitable metric space. This allows us to cast the distance between histograms associated with the contexts as an instance of the *optimal transport problem* (Monge, Kantorovich, 1942; Villani, 2008).

Our resulting framework then intuitively seeks to minimize the cost of moving the contexts of a given entity to the contexts of another, which motivates the naming *Context Mover’s Distance* (CMD). Note that the contexts here can be words, phrases, sentences, or any generic entities co-occurring with our entities to be represented, and these entities further could be of various kinds, including e.g., products such as movies or web-advertisements (Grbovic et al., 2015), nodes in a graph (Grover & Leskovec, 2016), sequence data, or any other entities (Wu et al., 2017). Any co-occurrence structure will allow construction of the histogram information, which is the crucial building block of our approach.

The main motivation for our proposed approach here comes from the domain of natural language, where the entities (words, phrases or sentences) generally have different semantics depending on the context under which they are present. Hence, it is important that we consider representations that are able to effectively capture such inherent uncertainty and polysemy, and we will argue that histograms (or probability distributions) over embeddings allows to capture more of this information compared to point-wise embeddings alone. We will call this histogram over contexts embeddings as...
the *distributional estimate* of our object of interest, while we refer to the individual embeddings of contexts as *point estimates*.

The connection to optimal transport at the level of entities and contexts paves the way to make better use of its vast toolkit (like Wasserstein distances, barycenters, barycentric coordinates, etc.) for applications in NLP, which in the past has primarily been restricted to document distances of original words (Kusner et al., 2015; Huang et al., 2016), as opposed to contexts. Thanks to the entropic regularization introduced by Cuturi (2013), optimal transport computations can be carried out efficiently in a parallel and batched manner on GPUs.

**Contributions:**

- Employing the notion of optimal transport of contexts as a distance measure, we illustrate how our framework can be of significant benefit for a wide variety of important tasks, including word sentence representation and similarity, as well as hypernymy (entailment) detection. The method is static and does not require any additional learning, and can be readily used on top of existing embedding methods.

- The resulting representations via the transport map give a clear interpretation of the resulting distance (see also Figure 1), on top of the co-occurrence information.

- Our context mover distance can leverage any kind of distance (even asymmetric) between words, by defining a suitable underlying cost on the movement of contexts, which we show can lead to a state-of-the-art metric for textual entailment.

- Defining the transport over contexts has the significant benefit that the representations are compositional - they directly extend from entities to groups of entities (of any size), such as from word to sentence representations. To this end, we utilize the notion of Wasserstein barycenters, which to the best of our knowledge has never been considered in the past.

- The proposed framework is not specific to words or sentences but allows building unsupervised representations for any entity and composition of entities, where a co-occurrence structure can be devised between entities and their contexts.

2 **Related Work**

Most of the previous work in building representations for natural language has been focused towards vector space models, in particular, popularized through the groundbreaking work in Word2vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014). The key idea in these models has been to map words which are similar in meaning to nearby points in a latent space. Based on which, many works (Levy & Goldberg, 2014a; Melamud et al., 2015; Bojanowski et al., 2016) have suggested specializing the embeddings to capture some particular information required for the task at hand. One of the problems that still persists is the inability to capture, within just a point embedding, the various semantics and uncertainties associated with the occurrence of a particular word (Huang et al., 2012; Guo et al., 2014).

A recent line of work has proposed the view to represent words with Gaussian distributions or mixtures of Gaussian distributions (Vilnis & McCallum, 2014b; Athiwarat & Wilson, 2017), or hyperbolic cones (Ganea et al., 2018) for this purpose. Also, concurrent works by Muzellec & Cuturi (2018) and Sun et al. (2018) have suggested using elliptical and Gaussian distributions endowed with a Wasserstein metric respectively. While these already provide richer information than typical vector embeddings, their form restricts what could be gained by allowing for arbitrary distributions. In addition, hyperbolic embeddings (Nickel & Kiela, 2017; Ganea et al., 2018) are so far restricted to supervised tasks (and even elliptical embeddings (Muzellec & Cuturi, 2018) to the most extent), not allowing unsupervised representation learning as in the focus of the paper here. To this end, we propose to associate with each word a distributional and a point estimate. These two estimates together play an important role and enable us to make use of optimal transport.

Amongst the few explorations of optimal transport in NLP, i.e., document distances (Kusner et al., 2015; Huang et al., 2016), document clustering (Ye et al., 2017), bilingual lexicon induction (Zhang et al., 2017), or learning an orthogonal Procrustes mapping in Wasserstein distance (Grave et al., 2018), the focus has been on transporting words directly. For example, the Word Mover’s Distance (Kusner et al., 2015) casts finding the distance between documents as an optimal transport problem.
between their bag of words representation. Our approach is different as we consider the transport over contexts instead, and use it to propose a representation for words or entities.

3 BACKGROUND ON OPTIMAL TRANSPORT

Optimal Transport (OT) provides a way to compare two probability distributions defined over a space $G$ (commonly known as the ground space), given an underlying distance or more generally a cost of moving one point to another in the ground space. In other terms, it lifts a distance between points to a distance between distributions. Other methods of comparing distributions, such as Kullback-Liebler (KL), squared Hellinger, etc., only focus on the probability mass values, thus ignoring the geometry of the ground space: something which we utilize throughout this work via OT. Also, some divergences like KL are not defined when the supports of distributions under comparison don’t match. Hence, we give a short yet formal background on OT in the discrete case.

**Linear Program Formulation.** Consider an empirical probability measure of the form \( \mu = \sum_{i=1}^{n} a_i \delta(x_i) \) where \( X = (x_1, \ldots, x_n) \in G^n \). \( \delta(x) \) denotes the Dirac (unit mass) distribution at point \( x \in G \), and \((a_1, \ldots, a_n) \) lives in the probability simplex \( \Sigma_n := \{ p \in \mathbb{R}_+^n \mid \sum_{i=1}^{n} p_i = 1 \} \). Now given a second empirical measure, \( \nu = \sum_{j=1}^{m} b_j \delta(y_j) \), with \( Y = (y_1, \ldots, y_m) \in G^m \), and \((b_1, \ldots, b_m) \in \Sigma_m \), and if the ground cost of moving from point \( x_i \) to \( y_j \) is denoted by \( M_{ij} \), then the Optimal Transport distance between \( \mu \) and \( \nu \) is the solution to the following linear program.

\[
\text{OT}(\mu, \nu; M) := \min_{T \in \mathbb{R}_{+}^{n \times m}} \sum_{ij} T_{ij} M_{ij} \quad \text{such that} \quad \forall i, \sum_{j} T_{ij} = a_i, \quad \forall j, \sum_{i} T_{ij} = b_j. \tag{1}
\]

Here, the optimal \( T \in \mathbb{R}_{+}^{n \times m} \) is referred to as the transportation matrix: \( T_{ij} \) denotes the optimal amount of mass to move from point \( x_i \) to point \( y_i \). Intuitively, OT is concerned with the problem of moving goods from factories to shops in such a way that all the demands are satisfied and the overall transportation cost is minimal.

**Distance.** When \( G = \mathbb{R}^d \) and the cost is defined with respect to a metric \( D_G \) over \( G \) (i.e., \( M_{ij} = D_G(x_i, y_j)^p \) for any \( i, j \)), OT defines a distance between empirical probability distributions. This is the \( p \)-Wasserstein distance, defined as \( W_p(\mu, \nu) := \text{OT}(\mu, \nu; D_G^p) \). In most cases, we are only concerned with the case where \( p = 1 \) or \( 2 \).

The cost of exactly solving OT problem scales at least in \( O(n^3 \log(n)) \) \( (n \) being the cardinality of the support of the empirical measure) when using network simplex or interior point methods. Following Cuturi [2013], we consider the entropy regularized Wasserstein distance, \( W_{p, \lambda}(\mu, \nu) \), where the search space for the optimal \( T \) is instead restricted to a smooth solution close to the extreme points of this linear program. The regularized problem can then be solved efficiently using Sinkhorn iterations CITE, albeit at the cost of some approximation error. The regularization strength \( \lambda \geq 0 \) controls the accuracy of approximation and recovers the true OT for \( \lambda = 0 \). The cost of the Sinkhorn algorithm is only quadratic in \( n \) at each iteration.

**Barycenters.** Further on in our discussion, we will make use of the notion of averaging in the Wasserstein space. More precisely, the Wasserstein barycenter, introduced by Agueh & Carlier [2011], is a probability measure that minimizes the sum of \( (p\text{-th power}) \) Wasserstein distances to the given measures. Formally, given \( N \) measures \( \{\nu_1, \ldots, \nu_N\} \) with corresponding weights \( \eta = \{\eta_1, \ldots, \eta_N\} \in \Sigma_N \), the Wasserstein barycenter can be written as \( B_p(\nu_1, \ldots, \nu_N) = \arg \min_{\nu} \sum_{i=1}^{N} \eta_i W_p(\mu, \nu_i)^p \). We similarly consider the regularized barycenter \( B_{p, \lambda} \), using entropy regularized Wasserstein distances \( W_{p, \lambda} \) in the above minimization problem, following Cuturi & Doucet [2014]. Employing the method of iterative Bregman projections (Benamou et al. [2015]), we obtain an approximation of the solution at a reasonable computational cost.

4 METHODOLOGY

In this section, we define the distributional estimate that we use to represent each entity. In view of the guiding example of building text representations, consider each entity to be a word for simplicity.
Distributional Estimate \( (P^w_V) \). For a word \( w \), its distributional estimate is built from a histogram over the set of contexts \( C \), and an embedding of these contexts into a space \( G \). The histogram essentially measures how likely it is for a word \( w \) to occur in a particular context \( c \), i.e., probability \( p(w|c) \). The exact formulation of this distribution is generally intractable and hence it’s common to empirically estimate this by the number of occurrences of the word \( w \) in context \( c \), relative to the total frequency of context \( c \) in the corpus.

Thus one way to build this histogram is to maintain a co-occurrence matrix between words in our vocabulary and all possible contexts, where each entry indicates how often a word and context occur in an interval (or window) of a fixed size \( L \). Then, the bin values \( (H^w)_{c \in C} \) of the histogram \( (H^w) \) for a word \( w \), can be viewed as the row corresponding to \( w \) in this co-occurrence matrix. In Section 5, we discuss possible modifications of the co-occurrence matrix to improve associations and how to reduce the number of bins in the histogram.

The simplest embedding of contexts is into the space of one-hot vectors of all the possible contexts. However, this induces a lot of sparsity/redundancy in the representation and the distance between such embeddings of contexts does not reflect their semantics. A classical solution would be to instead find a dense low-dimensional embedding of contexts that captures the semantics, possibly using techniques such as SVD or deep neural networks. We denote by \( V = (v_c)_{c \in C} \) an embedding of the contexts into this low-dimensional space \( G \subset \mathbb{R}^d \), which we refer to as the ground space. (We will consider example cases of how this metric can be obtained in Sections 6 and 7.)

Combining the histogram \( H^w \) and the embedding \( V \), we represent the word \( w \) by the following empirical distribution:

\[
P^w_V := \sum_{c \in C} (H^w)_c \delta(v_c).
\] (2)

Recall that \( \delta(v_c) \) denotes the Dirac measure at the position \( v_c \) of the context \( c \). We refer to this representation (Eq. 2) as the distributional estimate of the word.

Distance. If we equip the ground space \( G \) with a meaningful metric \( D_G \), then we can subsequently define a distance between the representations of two words \( w_i \) and \( w_j \), as the solution to the following optimal transport problem:

\[
\text{CMD}(w_i, w_j; D^p_G) := \text{OT}(P^w_{V_i}; P^w_{V_j}; D^p_G) \simeq W_{p, \lambda}((P^w_{V_i}, P^w_{V_j}))^p.
\] (3)

Under this formulation, we call the above distance (Eq. 3) the Context Mover’s Distance (CMD), borrowing the name from Rubner et al. (2000)’s famous Earth Mover’s Distance in computer vision.

Intuition. Two words are similar in meaning if the contexts of one word can be easily transported to the contexts of the other, with this cost of transportation being measured by \( D_G \). This idea still remains in line with the distributional hypothesis (Harris 1954; Rubenstein & Goodenough 1965) that words in similar contexts have similar meanings, but provides a precise way to quantify it.

Interpretation. In fact, both elements of the distributional estimate: the histogram and point estimates are closely tied together and required to serve as an effective representation. For instance, let’s take a toy example and discuss a scenario that might arise when we only have the histogram information. Consider three words, ‘Tennis’, ‘Football’, and ‘Law’, admitting as contexts...
We only took into account the histograms, we would reach the inaccurate conclusion that ‘Tennis’ is closer in semantics to ‘Law’ than to ‘Football’, as there is a considerable overlap at the important context of ‘Court’ for ‘Tennis’ and ‘Law’. Whereas, together with the point estimate information, it is apparent that the context ‘Stadium’ (in $H^{Football}$) can be more cheaply moved to ‘Court’ (in $H^{Tennis}$), but moving ‘Firm’ (in $H^{Law}$) to some context in $H^{Tennis}$ is more costly. Lastly, in the reverse scenario of only considering the point estimates, we would lose much of the uncertainty associated about the contexts in which the words occur. We illustrate these scenarios in Figure 2.

Roadmap. First, we discuss a concrete framework of how this can be applied in the next section. In Sections 6, we detail how this framework can be extended to obtain representation for a composition of entities via Wasserstein barycenter. Lastly in section 7, we utilize the fact that the CMD in Eq. 3 is parameterized by ground cost, and illustrate how this flexibility can be used to define an asymmetric cost measuring entailment. Overall, the family of problems where such a representation can potentially be used is not restricted to entities belonging to NLP, but in any domain where a co-occurrence structure exists between entities and their contexts.

5 CONCRETE FRAMEWORK

Making associations better. We consider co-occurrences of a word and a context word if the latter appears in a symmetric window of size $L$ around the target word (the word whose distributional estimate we seek). While each entry of the co-occurrence matrix reflects the co-occurrence count of a target word and its context, the counts alone may not necessarily suggest a strong association between the two. The well-known Positive Pointwise Mutual Information (PPMI) matrix ([Church & Hanks, 1990; Levy et al., 2015]) addresses this shortcoming, and is defined as follows: $PPMI(w, c) := \max(\log\left(\frac{p(w|c)}{p(w)\cdot p(c)}\right), 0)$. The PPMI entries are non-zero when the joint probability of co-occurring target and context words is higher than the probability when they are independent. Typically, these probabilities are estimated from the co-occurrence counts in the corpus. Further improvements to the PPMI matrix have been suggested, like in [Levy & Goldberg, 2014b], and following them we make use of a shifted and smoothed PPMI matrix, denoted by $SPPMI_{\alpha,s}$, where $\alpha$ and $s$ denote the smoothing and k-shift parameters. Overall, these variants of PPMI enable us to extract better semantic associations\footnote{We refer to Appendix A.2 for more details on PPMI and its normalized variants.} from the co-occurrence matrix. Hence, the bin values (at context $c$) for the histogram of word $w$ in Eq. 2 can be formulated as: $\left((H^w)_{c}\right)_c := \frac{SPPMI_{\alpha,s}(w, c)}{\sum_{c \in C} SPPMI_{\alpha,s}(w, c)}$.

Computational considerations. The view of optimal transport between histograms of contexts introduced in Eq. 3 offers a pleasing interpretation (see Figure 1). However, it might be computationally intractable in its current formulation, since the number of possible contexts can be as large as the size of vocabulary (if the contexts are just single words) or even exponential (if contexts are considered to be phrases, sentences and otherwise). This is problematic because the Sinkhorn algorithm for regularized optimal transport ([Cuturi, 2013] see Section 3) scales roughly quadratically in the histogram size, and the ground cost matrix can also become prohibitive to store in memory. One possible fix is to instead consider a set of representative contexts in this ground space, for example via clustering. We believe that with dense low-dimensional embeddings and a meaningful metric between them, we may not require as many contexts as needed before. Apart from the computational

\begin{enumerate}
\item \textbf{Stadium, Court, Match, Firm}
\item \textbf{Tennis, Football, Law}
\item \textbf{Stadium, Court, Match, Firm}
\item \textbf{Tennis, Football, Law}
\item \textbf{Stadium, Court, Match, Firm}
\item \textbf{Tennis, Football, Law}
\end{enumerate}
gain, the clustering will lead to transport between more abstract contexts. This will although come at
the loss of some interpretability.

Now, consider that we have obtained $K$ representative contexts, each covering some part $C_k$ of the
set of contexts $C$. The histogram for word $w$ with respect to these contexts can then be written as
$$
\hat{H}_w = \sum_{k=1}^{K} (H^w)_k \delta(v_k).$$
Here $v_k \in \hat{V}$ is the point estimate of the $k^{th}$ representative context, and $(H^w)_k$ denote the new histogram bin values with respect to the part $C_k$.

$$
(\hat{H}^w)_k \definedas \frac{\text{SPPMI}_{\alpha,s}(w, C_k)}{\sum_{k=1}^{K} \text{SPPMI}_{\alpha,s}(w, C_k)}
,$$
with $\text{SPPMI}_{\alpha,s}(w, C_k) \definedas \sum_{c \in C_k} \text{SPPMI}_{\alpha,s}(w, c)$.

(4)

**Summary.** With the above aspects in account and using batched implementations on (Nvidia TitanX)
GPUs, it is possible to compute around 13,700 Wasserstein-distances/second (for histogram of size
100). Same also holds for barycenters, where we can compute 4,600 barycenters/second for sentences
of length 25 and histogram size of 100. Building this histogram information comes for almost free
during the typical learning of embeddings, as in GloVe (Pennington et al., 2014). One practical
take-home message of this work is, when you use GloVe never throw away the co-occurrence binary,
instead pass it to our method.

6  **Sentence Representation with CoMB**

Traditionally, the goal of this task is to develop a representation for sentences, that captures the
semantics conveyed by it. Most unsupervised representations proposed in the past rely on the
composition of vector embeddings for the words, through either additive, multiplicative, or other
ways (Mitchell & Lapata, 2008; Arora et al., 2017; Pagliardini et al., 2017). We propose to represent
sentences as probability distributions to better capture the inherent uncertainty and polysemy.

Our belief is that a sentence representation is meaningful if it best captures the simultaneous occur-
rence of the words in it. We hypothesize that a sentence, $S = (w_1, w_2, \ldots, w_N)$, can be efficiently
represented via the Wasserstein barycenter (see Section 3) of the distributional estimates of its words,

$$
\tilde{p}_S := B_{p,\lambda}(\tilde{p}_{V_1}^{w_1}, \tilde{p}_{V_2}^{w_2}, \ldots, \tilde{p}_{V_N}^{w_N})
,$$
which is itself again a distribution over the ground space $\mathcal{G}$. We refer to this representation as the
**Context Mover’s Barycenter** (CoMB) henceforth. Interestingly, the classical weighted averaging
of point-estimates, like Smooth Inverse Frequency (SIF) in (Arora et al., 2017) (without principal
component removal), can be seen as a special case of CoMB, when the distribution associated to a
word is just a Dirac at its point estimate. It becomes apparent that having a rich distributional estimate
for a word could turn out to be advantageous.

Since with barycenter representation as in Eq. 5 each sentence is also a distribution over contexts,
we can utilize the Context Mover’s Distance (CMD) defined in Eq. 3 to define the distance between
two sentences $S_1$ and $S_2$, under a given ground metric $D_G$ as follows,

$$
\text{CMD}(S_1,S_2; D_G) := \text{OT}(\tilde{p}_V^{S_1}, \tilde{p}_V^{S_2}; D_G) \simeq W_{p,\lambda}(\tilde{p}_V^{S_1}, \tilde{p}_V^{S_2})^p.
$$

(6)

**Empirical Evaluation.** To evaluate CoMB as an effective sentence representation, we consider 24
datasets from SemEval semantic textual similarity (STS) tasks (Agirre et al., 2012, 2013, 2014, 2015,
2016). The objective here is to give a score of how similar two sentences are in their meanings.

As a ground metric ($D_G$), we consider the Euclidean distance between the point estimates (embed-
bings) of words. We train the word embeddings on the Toronto Book Corpus (Kiros et al., 2015) via
GloVe (Pennington et al., 2014), and in this process also obtain the histogram information needed for
the distributional estimate. Since GloVe embeddings for similar words are constructed to be close in
terms of cosine similarity for similar words, we find the representative points by performing K-means
clustering with respect to this similarity for $K = 300$

We benchmark our performance against SIF (Smooth Inverse Frequency) from Arora et al., 2017
who regard it as a “simple but tough-to-beat baseline”, as well as against the plain Bag of Words

2We use SIF’s publicly available implementation (https://github.com/PrincetonML/SIF) and use SentEval (Conneau & Kiela, 2018) for evaluating BoW and CoMB.
Under review as a conference paper at ICLR 2019

<table>
<thead>
<tr>
<th>Model</th>
<th>STS16</th>
<th>STS12</th>
<th>STS13</th>
<th>STS14</th>
<th>STS15</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>BoW</td>
<td>22.6</td>
<td>23.8</td>
<td>20.2</td>
<td>29.4</td>
<td>31.5</td>
<td>26.2</td>
</tr>
<tr>
<td>SIF (α = 0.001, no PC removed)</td>
<td>22.7</td>
<td>32.9</td>
<td>21.4</td>
<td>51.0</td>
<td>52.0</td>
<td>48.3</td>
</tr>
<tr>
<td>SIF (α = 0.001, PC removed)</td>
<td>55.4</td>
<td>40.5</td>
<td>49.8</td>
<td>51.0</td>
<td>52.0</td>
<td>48.3</td>
</tr>
<tr>
<td>CoMB (α=0.15, β=0.5, s=1)</td>
<td>47.4</td>
<td>44.9</td>
<td>48.1</td>
<td>50.1</td>
<td>52.0</td>
<td>49.0</td>
</tr>
<tr>
<td>CoMB (α=0.55, β=0.5, s=5)</td>
<td>47.6</td>
<td>49.1</td>
<td>40.6</td>
<td>53.4</td>
<td>52.7</td>
<td>48.9</td>
</tr>
<tr>
<td>CoMB (α=0.55, β=1, s=5)</td>
<td>49.1</td>
<td>48.3</td>
<td>41.5</td>
<td>53.6</td>
<td>53.3</td>
<td>49.2</td>
</tr>
</tbody>
</table>

Table 1: Performance of Context Mover’s Barycenter (CoMB) and related baselines on the STS tasks using Toronto Book Corpus. The numbers are average Pearson correlation x 100 (with respect to groundtruth scores). CoMB outperforms the best SIF baseline on 3 out of 4 tasks in the test set and also leads to an overall improvement on average for several hyperparameter settings. It is also 1.5x and 2x better than the SIF with no PC removed and BoW. Here, α, β, s denote the PPMI smoothing, column normalization exponent (Eq. 10) and k-shift.

In the above experiments, our focus was to compare methods which can build up sentence representations by just obtaining the word vector information. Hence, we didn’t include unsupervised methods such as Sent2vec (Pagliardini et al., 2017), that are specifically trained to work well on sentence similarity. The above results are quite encouraging, given the fact that we haven’t even utilized the important property of non-associativity for Wasserstein barycenters (i.e., \( B_p(\mu, B_p(\nu, \xi)) \neq B_p(B_p(\mu, \nu), \xi) \)). This implies that we can take into account the word order with various aggregation strategies, like parse trees, and build the sentence representation by recursively computing barycenters phrase by phrase, which although remains beyond the scope of this paper.

Overall, this highlights towards the advantage of having distributional estimates for words, that can be extended to give a meaningful representation of sentences via CoMB in a principled manner.

7 Hypernymy Detection

In linguistics, hypernymy is a relation between words (or sentences) where the semantics of one word (the hyponym) are contained within that of another word (the hypernym). A simple form of this relation is the is-A relation, e.g., cat is an animal. Hypernymy is a special case of the more general concept of lexical entailment, the detection of which is relevant for tasks such as Question Answering (QA). Given a database of lexical entailment relations containing, e.g., is-A (Roger Federer, tennis player) might help a QA system answer “Who is Switzerland’s most successful tennis player?”.

The early unsupervised approaches for this task exploited different linguistic properties of hypernymy (Weeds & Weir, 2003; Kotlerman et al., 2010; Santus et al., 2014; Rimell, 2014). While most of these are count-based, word embedding based methods (Chang et al., 2017; Nickel & Kiela, 2017; Henderson & Popa, 2016) have become more popular in recent years. Other approaches represent words by Gaussian distributions with KL-divergence as a measure of entailment (Vilnis & McCallum, 2014a; Athiwaratkun & Wilson, 2017). These methods have proven to be powerful, as they not only capture the semantics but also the uncertainty about the contexts in which the word appears.

Therefore, hypernymy detection is a great testbed to verify the effectiveness of our approach (and the particular formulation) to represent each entity by the distribution of its contexts. To be successful on this task, a method has to consider if all contexts of the hyponym can be encompassed within the contexts of the hypernym. It can’t just get away by predicting words that are similar. Hence, it is natural to make use of the Context Mover’s Distance (CMD), Eq. 3, but with an appropriate ground cost that measures entailment relations well.

For this purpose, we utilize a recently proposed method by Henderson & Popa (2016; Henderson, 2017) which explicitly models what information is known about a word, by interpreting each entry of
the embedding as the degree to which a certain feature is present. Based on the logical definition of entailment they derive an operator measuring the entailment similarity between two so-called entailment vectors defined as follows: 

\[ \vec{v}_i = \sigma(-\vec{v}_i) \cdot \log \sigma(-\vec{v}_i), \] 

where the sigmoid \( \sigma \) and log are applied component-wise on the embeddings \( \vec{v}_i, \vec{v}_j \). Thus, we use as ground cost \( D_{ij}^{\text{Hend.}} := -\vec{v}_i \otimes \vec{v}_j \).

This asymmetric ground cost also shows that our framework can be flexibly used with an arbitrary cost function defined on the ground space.

**Evaluation.** In total, we evaluated our method on 9 standard datasets: BLESS (Baroni & Lenci, 2011), EVALution (Santus et al., 2015), Benotto (2015), Weeds et al. (2014), Henderson (2017), Baroni et al. (2012), Kotlerman et al. (2010), Levy et al. (2014) and Turney & Mohammad (2015). As an evaluation metric, we use average precision AP@all Zhu (2004). Following Chang et al. (2017) we pushed any OOV (out-of-vocabulary) words in the test data to the bottom of the list, effectively assuming that the word pairs do not have a hypernym relation.

The foremost thing that we would like to check is the benefit of having a distributional estimate in comparison to just the point embeddings. Here, we observe that by employing CMD along with the entailment embeddings, leads to a significant boost on almost all of the datasets, except on Baroni, where the performance is still competitive with the other state of the art methods like Gaussian embeddings. The more interesting observation is that on some datasets (EVALution, LenciBenotto, Weeds, Turney) we even outperform or match state-of-the-art performance (cf. Table 2), by simply using CMD together with this ground cost \( D_{ij}^{\text{Hend.}} \) based on the entailment embeddings.

Notably, this approach is not specific to the entailment vectors from Henderson (2017). It can possibly be used with any embedding vectors which provide a good measure of the degree of entailment; and a more accurate set of vectors might even further improve the performance. Also, our training dataset, Wikipedia with 1.7B tokens, and our vocabulary with only 80'000 words are rather small compared to the datasets used, e.g., by Vilnis & McCallum (2014a). We expect to get even better results by using a larger vocabulary on a larger corpus.

### 8 Conclusion

We advocate for representing entities by a distributional estimate on top of any given co-occurrence structure. For each entity, we jointly consider the histogram information (with its contexts) as well as the point embeddings of the contexts. We show how this enables the use of optimal transport over distributions of contexts. Our framework results in an efficient, interpretable and compositional metric to represent and compare entities (e.g. words) and groups thereof (e.g. sentences), while leveraging existing point embeddings. We demonstrate its performance on several NLP tasks. Motivated by these empirical results, applying the proposed framework on co-occurrence structures beyond NLP is a promising direction.

---

Table 2: Comparison of the entailment vectors alone (Hend.) and when used together with our CMD\( _\alpha \) in the form of ground cost \( D_{ij}^{\text{Hend.}} \). Besides these, the table also includes other state-of-the-art methods, like Gaussian embeddings with cosine similarity (GE+C) and negative KL-divergence (GE+KL). Scores for GE+C, GE+KL, and DIVE + C\( \cdot \)\( \Delta S \) are taken from Chang et al. (2017) as we use the same evaluation setup. The scores are AP@all (%).

<table>
<thead>
<tr>
<th>Method</th>
<th>EVALution</th>
<th>LenciBenotto</th>
<th>Weeds</th>
<th>Turney</th>
<th>Baroni</th>
</tr>
</thead>
<tbody>
<tr>
<td>GE + C</td>
<td>26.7</td>
<td>43.3</td>
<td>52.0</td>
<td>53.9</td>
<td>69.7</td>
</tr>
<tr>
<td>GE + KL</td>
<td>29.6</td>
<td>45.1</td>
<td>51.3</td>
<td>52.0</td>
<td>64.6</td>
</tr>
<tr>
<td>DIVE + C( \cdot )( \Delta S )</td>
<td>33.0</td>
<td>50.4</td>
<td>65.5</td>
<td>57.2</td>
<td>83.5</td>
</tr>
<tr>
<td>Henderson et al.</td>
<td>31.6</td>
<td>44.8</td>
<td>60.8</td>
<td>56.6</td>
<td>78.3</td>
</tr>
<tr>
<td>CMD( <em>{0.15} ) + ( D</em>{ij}^{\text{Hend.}} )</td>
<td>39.8</td>
<td>48.5</td>
<td>64.7</td>
<td>57.3</td>
<td>65.5</td>
</tr>
<tr>
<td>CMD( <em>{0.5} ) + ( D</em>{ij}^{\text{Hend.}} )</td>
<td><strong>40.5</strong></td>
<td><strong>49.5</strong></td>
<td><strong>66.2</strong></td>
<td>56.1</td>
<td>67.4</td>
</tr>
</tbody>
</table>

Under review as a conference paper at ICLR 2019

---

**Details of training setup & effect of PPMI parameters can be found in section A.1 & Table A.7 of Appendix.**
REFERENCES


A SUPPLEMENTARY MATERIAL

A.1 EXPERIMENTAL DETAILS:

Sentence Representations: While using the Toronto Book Corpus, we remove the errors caused by crawling and pre-process the corpus by filtering out sentences longer than 300 words, thereby removing a very small portion (500 sentences out of the 70 million sentences). We utilize the code\footnote{https://github.com/stanfordnlp/GloVe} from GloVe for building the vocabulary of size 205513 (obtained by setting min_count=10) and the co-occurrence matrix (considering a symmetric window of size 10). Note that as in GloVe, the contribution from a context word is inversely weighted by the distance to the target word, while computing the co-occurrence. The vectors obtained via GloVe have 300 dimensions and were trained for 75 iterations at a learning rate of 0.005, other parameters being the default ones. The performance of these vectors from GloVe was verified on standard word similarity tasks.

Hypernymy Detection: The training of the entailment vector is performed on a Wikipedia dump from 2015 with 1.7B tokens that have been tokenized using the Stanford NLP library\footnote{https://docs.scipy.org/doc/scipy/reference/sparse.html} (Manning et al., 2014). In our experiments, we use a vocabulary with a size of 80’000 and word embeddings with 200 dimensions and 100 cluster centers. We followed the same training procedure as described in\footnote{https://docs.scipy.org/doc/scipy/reference/sparse.html} Henderson (2017) and were able to reproduce their scores on the hypernymy detection task.

A.2 PPMI DETAILS

Formulation and Variants: Typically, the probabilities used in PMI are estimated from the co-occurrence counts $\#(w,c)$ in the corpus and lead to

$$PPMI(w,c) = \max \left( \log \left( \frac{\#(w,c) \times |Z|}{\#(w) \times \#(c)} \right), 0 \right),$$  \hspace{1cm} (7)

where, $\#(w) = \sum_c \#(w,c)$, $\#(c) = \sum_w \#(w,c)$ and $|Z| = \sum_w \sum_c \#(w,c)$. Also, it is known that PMI is biased towards infrequent words and assigns them a higher value. A common solution is to smoothen the context probabilities by raising them to an exponent of $\alpha$ lying between 0 and 1. Levy & Goldberg (2014b) have also suggested the use of the shifted PPMI (SPPMI) matrix where the shift by $\log(s)$ acts like a prior on the probability of co-occurrence of target and context pairs. These variants of PPMI enable us to extract better semantic associations from the co-occurrence matrix. Finally, we have

$$\text{SPPMI}_{\alpha,s}(w,c) := \max \left( \log \left( \frac{\#(w,c) \times \sum_{c'} \#(c')^\alpha}{\#(w) \times \#(c)^\alpha} \right) - \log(s), 0 \right).$$

Computational aspect: We utilize the sparse matrix support of Scipy\footnote{https://docs.scipy.org/doc/scipy/reference/sparse.html} for efficiently carrying out all the PPMI computations.

PPMI Column Normalizations: In certain cases, when the PPMI contributions towards the partitions (or clusters) have a large variance, it can be helpful to consider the fraction of $C_k$’s SPPMI (Eq. 8) \footnote{https://docs.scipy.org/doc/scipy/reference/sparse.html} that has been used towards a word $w$, instead of aggregate values used in\footnote{https://github.com/stanfordnlp/GloVe}. Otherwise the process of making the histogram unit sum might misrepresent the actual underlying contribution. We call this PPMI column normalization ($\beta$). In other words, the intuition is that the normalization will balance the effect of a possible non-uniform spread in total PPMI across the clusters. We observe that setting $\beta$ to 0.5 or 1 help in boosting performance on the STS tasks. The basic form of column normalization is shown in\footnote{https://github.com/stanfordnlp/GloVe} [9]

$$\tilde{H}_k^w := \frac{(H_k^w)}{\sum_{k=1}^{K} (H_k^w)} \quad \text{with} \quad (H_k^w) := \frac{\text{SPPMI}_{\alpha,s}(w,C_k)}{\sum_w \text{SPPMI}_{\alpha,s}(w,C_k)}.$$ \hspace{1cm} (9)
Another possibility while considering the normalization to have an associated parameter $\beta$ that can interpolate between the above normalization and normalization with respect to cluster size.

\[
(\tilde{H}^w_\beta)_k := \frac{(H^w_\beta)_k}{\sum_{k=1}^{K} (\tilde{H}^w_\beta)_k}, \quad \text{where}
\]

\[
(\tilde{H}^w_\beta)_k := \frac{\text{SPPMI}_{\alpha,s}(w, C_k)}{\sum_{w} \text{SPPMI}_{\alpha,s}(w, C_k)'^\beta}
\]

In particular, when $\beta = 1$, we recover the equation for histograms as in [9] and $\beta = 0$ would imply normalization with respect to cluster sizes.

A.3 Optimal Transport

Implementation aspects. We make use of the Python Optimal Transport (POT)\(^8\) for performing the computation of Wasserstein distances and barycenters on CPU. For more efficient GPU implementation, we built custom implementation using PyTorch. We also implement a batched version for barycenter computation, which to the best of our knowledge has not been done in the past. The batched barycenter computation relies on viewing computations in the form of block-diagonal matrices. As an example, this batched mode can compute around 200 barycenters in 0.09 seconds, where each barycenter is of 50 histograms (of size 100) and usually gives a speedup of about 10x.

Scalability. For further scalability, an alternative is to consider stochastic optimal transport techniques (Genevay et al., 2016). Here, the idea would be to randomly sample a subset of contexts from the distributional estimate while considering this transport.

Stability of Sinkhorn Iterations. For all our computations involving optimal transport, we typically use $\lambda$ around 0.1 and make use of log or median normalization as common in POT to stabilize the Sinkhorn iterations. Also, we observe that clipping the ground metric matrix (if it exceeds a particular large threshold) also sometimes results in performance gains.

A.4 Clustering.

For clustering, we make use of kmcuda\(^9\), efficient implementation of K-Means algorithm on GPUs.

A.5 Software Release

We plan to make all our code (for all these parts) and our pre-computed histograms (for the mentioned datasets) publicly available on GitHub soon.

\[^9\]https://github.com/src-d/kmcuda
A.6 Qualitative Analysis

Qualitative Evaluation. Here, we would like to qualitatively probe the kind of results obtained when computing Wasserstein barycenter of the distributional estimates, in particular, when using CoMB to represent sentences. To this end, we consider a few simple sentences and find the closest word in the vocabulary for CoMB (with respect to CMD) and contrast it to SIF with cosine distance.

<table>
<thead>
<tr>
<th>Query</th>
<th>CoMB (with CMD)</th>
<th>SIF (with cosine, no PC removal)</th>
</tr>
</thead>
<tbody>
<tr>
<td>[i', 'love', 'her']</td>
<td>love, hope, always, actually, because, doubt, imagine, but, never, simply</td>
<td>love, loved, bleep-bleep, want, clash-clash-clang, thyself, know, think, hope, life</td>
</tr>
<tr>
<td>[my', 'favorite', 'sport']</td>
<td>sport, costume, circus, costumes, outfits, super, sports, tennis, brand, fabulons</td>
<td>favorite, favourite, sport, wiccan-type, pastime, pastimes, sports, best, hangout, spectator</td>
</tr>
<tr>
<td>[best', 'day', 'of', 'my', 'life']</td>
<td>best, for, also, only, or, anymore, all, is, having, especially</td>
<td>life, day, best, c.5, writer/mummy, days, margin-bottom, time, margin-left, night</td>
</tr>
<tr>
<td>[he', 'lives', 'in', 'europe', 'for']</td>
<td>america, europe, decades, asia, millennium, preserve, masters, majority, elsewhere, commerce</td>
<td>lives, europe, life, america, lived, world, england, france, people, c.5</td>
</tr>
<tr>
<td>[he', 'may', 'not', 'live']</td>
<td>unless, perhaps, must, may, anymore, will, likely, you'll, would, certainly</td>
<td>may, live, should, will, might, must, margin-left, therefore, 0618082132, think</td>
</tr>
<tr>
<td>[can', 'you', 'help', 'me', 'shopping']</td>
<td>anytime, yesterday, skip, overnight, week, after, afterward, choosing, figuring, deciding, shopping</td>
<td>help, can, going, want, go, do, think, need, able, take</td>
</tr>
<tr>
<td>[he', 'likes', 'to', 'sleep', 'a', 'lot']</td>
<td>whenever, forgetting, afterward, pretending, rowan, eden, casper, nash, annabelle, savannah</td>
<td>lot, sleep, much, besides, better, likes, really, think, probably, talk</td>
</tr>
</tbody>
</table>

Table 3: Top 10 closest neighbors for CoMB and SIF (no PC removed) found across the vocabulary, and sorted in ascending order of distance from the query sentence. Words in italics are those which in our opinion would fit well when added to one of the places in the query sentence. Note that, both CoMB (under current formulation) and SIF don’t take the word order into account.

We find that closest neighbors (see Table 3) for CoMB consist of relatively more diverse set of words which fit well in the context of given sentence. For example, take the sentence “i love her”, where CoMB captures a wide range of contexts, for example, “i actually love her”, “i love her because”, “i doubt her love” and more. Also for an ambiguous sentence “he lives in europe for”, the obtained closest neighbors for CoMB include: ‘decades’, ‘masters’, ‘majority’, ‘commerce’ , etc., while with SIF the closest neighbors are mostly words similar to one of the query words. Further, if you look at the last three sentences in the Table 3 the first closest neighbor for CoMB even acts as a good next word for the given query. This suggests that CoMB might perform well on the task of sentence completion, but this additional evaluation is beyond the scope of this paper.
A.7 Detailed Results

Detailed results of the sentence representation and hypernymy detection experiments are listed on the following pages.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
<th>BLESS</th>
<th>EVALution</th>
<th>LenciBenotto</th>
<th>Weeds</th>
<th>Henderson</th>
<th>Baroni</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Henderson et al. ($D^{Hend.}$)</td>
<td>6.4</td>
<td>31.6</td>
<td>44.8</td>
<td>60.8</td>
<td>70.5</td>
<td><strong>78.3</strong></td>
</tr>
<tr>
<td></td>
<td>CMD ($\alpha=0.15$, $s=1$) + $D^{Hend.}$</td>
<td>7.3</td>
<td>37.7</td>
<td>49.0</td>
<td>63.6</td>
<td>74.8</td>
<td>64.4</td>
</tr>
<tr>
<td></td>
<td>CMD ($\alpha=0.15$, $s=5$) + $D^{Hend.}$</td>
<td>6.9</td>
<td>39.1</td>
<td>49.4</td>
<td>64.3</td>
<td>74.0</td>
<td>65.2</td>
</tr>
<tr>
<td></td>
<td>CMD ($\alpha=0.15$, $s=15$) + $D^{Hend.}$</td>
<td>7.0</td>
<td>39.8</td>
<td>48.5</td>
<td>64.7</td>
<td>75.0</td>
<td>65.6</td>
</tr>
<tr>
<td></td>
<td>CMD ($\alpha=0.5$, $s=1$) + $D^{Hend.}$</td>
<td>6.6</td>
<td>39.2</td>
<td>48.6</td>
<td>62.9</td>
<td><strong>76.1</strong></td>
<td>64.6</td>
</tr>
<tr>
<td></td>
<td>CMD ($\alpha=0.5$, $s=5$) + $D^{Hend.}$</td>
<td>5.9</td>
<td>40.4</td>
<td>49.9</td>
<td>65.7</td>
<td>73.9</td>
<td>67.2</td>
</tr>
<tr>
<td></td>
<td>CMD ($\alpha=0.5$, $s=15$) + $D^{Hend.}$</td>
<td>5.5</td>
<td><strong>40.5</strong></td>
<td>49.5</td>
<td><strong>66.2</strong></td>
<td>72.8</td>
<td>67.4</td>
</tr>
</tbody>
</table>

Table 4: Comparison of the entailment vectors alone (Hend.) and when used together with our CMD$_{\alpha,s}$ in the form of ground cost $D^{Hend}$. Avg. gain refers to the average difference relative to the entailment vectors. Avg. gain w/o Baroni refers to the average difference while neglecting the Baroni dataset. The hyperparameter $\alpha$ refers to the smoothing exponent and $s$ to the shift in the PPMI computation. All scores are AP at all (%). Note that, the Henderson dataset is a subset of the Weeds dataset [https://github.com/julieweeds/BLESS](https://github.com/julieweeds/BLESS).
Table 5: Detailed performance of Context Mover’s Barycenter (CoMB) and related baselines on the STS tasks using Toronto Book Corpus. The numbers are average Pearson correlation x 100 (with respect to groundtruth scores). CoMB outperforms the best SIF baseline on 3 out of 4 tasks in the test set and also leads to an overall improvement on average for several hyperparameter settings. It is also 1.5x and 2x better than the SIF with no PC removed and BoW. Here, $\alpha$, $\beta$, $s$ denote the PPMI smoothing, column normalization exponent (Eq. 10) and k-shift.

We observe empirically that the PPMI smoothing parameter $\alpha$, which balances the bias of PPMI towards rare words, plays an important role. While its ideal value would vary on each task, we found the settings mentioned in the Table to work well uniformly across the above spectrum of tasks.
<table>
<thead>
<tr>
<th>Model</th>
<th>Forum</th>
<th>Students</th>
<th>Belief</th>
<th>Headlines</th>
<th>Images</th>
</tr>
</thead>
<tbody>
<tr>
<td>BoW</td>
<td>20.1</td>
<td>45.4</td>
<td>24.4</td>
<td>36.5</td>
<td>31.2</td>
</tr>
<tr>
<td>SIF ($\alpha = 0.001$, no PC removed)</td>
<td>26.4</td>
<td>38.3</td>
<td>31.6</td>
<td>52.3</td>
<td>40.4</td>
</tr>
<tr>
<td>SIF ($\alpha = 0.001$, PC removed)</td>
<td>30.0</td>
<td>62.0</td>
<td>39.0</td>
<td>59.1</td>
<td>50.6</td>
</tr>
<tr>
<td>SIF ($\alpha = 0.0001$, PC removed)</td>
<td>34.0</td>
<td>63.7</td>
<td>48.4</td>
<td>62.4</td>
<td>51.7</td>
</tr>
<tr>
<td>CoMB ($\alpha=0.15$, $\beta=0.5$, $s=1$)</td>
<td><strong>44.7</strong></td>
<td>58.4</td>
<td>43.2</td>
<td>60.0</td>
<td>58.4</td>
</tr>
<tr>
<td>CoMB ($\alpha=0.55$, $\beta=0.5$, $s=5$)</td>
<td>39.0</td>
<td><strong>63.3</strong></td>
<td>37.8</td>
<td>60.3</td>
<td><strong>63.1</strong></td>
</tr>
<tr>
<td>CoMB ($\alpha=0.55$, $\beta=1$, $s=5$)</td>
<td>36.8</td>
<td>63.0</td>
<td>44.5</td>
<td>60.7</td>
<td>61.4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model</th>
<th>Answer</th>
<th>Headlines</th>
<th>Plagiarism</th>
<th>Postediting</th>
<th>Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>BoW</td>
<td>17.1</td>
<td>33.5</td>
<td>25.8</td>
<td>37.1</td>
<td>-0.6</td>
</tr>
<tr>
<td>SIF ($\alpha = 0.001$, no PC removed)</td>
<td>21.3</td>
<td>49.1</td>
<td>14.2</td>
<td>35.5</td>
<td>-6.4</td>
</tr>
<tr>
<td>SIF ($\alpha = 0.001$, PC removed)</td>
<td>26.0</td>
<td>57.0</td>
<td>43.4</td>
<td>61.5</td>
<td>18.2</td>
</tr>
<tr>
<td>SIF ($\alpha = 0.0001$, PC removed)</td>
<td>34.2</td>
<td>60.2</td>
<td>58.0</td>
<td>71.2</td>
<td>53.5</td>
</tr>
<tr>
<td>CoMB ($\alpha=0.15$, $\beta=0.5$, $s=1$)</td>
<td>21.6</td>
<td>51.9</td>
<td>48.8</td>
<td>64.0</td>
<td>50.9</td>
</tr>
<tr>
<td>CoMB ($\alpha=0.55$, $\beta=0.5$, $s=5$)</td>
<td>18.0</td>
<td>53.0</td>
<td>54.6</td>
<td>65.6</td>
<td>46.7</td>
</tr>
<tr>
<td>CoMB ($\alpha=0.55$, $\beta=1$, $s=5$)</td>
<td>26.2</td>
<td>54.8</td>
<td>51.3</td>
<td>66.6</td>
<td>46.6</td>
</tr>
</tbody>
</table>

Table 6: Detailed performance of Context Mover’s Barycenter (CoMB) and related baselines on the STS tasks using Toronto Book Corpus. The numbers are average Pearson correlation x 100 (with respect to groundtruth scores). CoMB outperforms the best SIF baseline on 3 out of 4 tasks in the test set and also leads to an overall improvement on average for several hyperparameter settings. It is also 1.5x and 2x better than the SIF with no PC removed and BoW. Here, $\alpha, \beta, s$ denote the PPMI smoothing, column normalization exponent (Eq. [10]) and k-shift.