Summary of Paper — From Word Embeddings To Document Distances

Yan Chen
May 4, 2021

1 Introduction

The paper introduced here presented a distance metric called "Word Mover's Distance" to measure the distance between text documents. This metric is an instance of the Earth Mover's Distance, which is a well-studied transportation problem. This paper was inspired by the previous results in word embeddings, namely *word2vec* by Mikolov et al. (2013b)[2]. The original WMD paper [1] on which I'm discussing can be found here.

2 Theory

2.1 *word2vec* Embedding

Given that the work is based on *word2vec* embedding, I'll first briefly discuss this word embedding procedure. Each word vector is trained to maximize the log probability of neighboring words in a corpus. i.e., given a sequence of words $w_1, w_2, \ldots, w_T$,

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{j\in nb(t)}\log p(w_j|w_t)$$

where $nb(t)$ is the set of neighboring words of word w_t and $p(w_j|w_t)$ is the hierarchical softmax of the associated word vectors corresponding to w_j and w_t. The intuition behind is that words can be represented more by the words that tend to appear with it more often. Surprisingly, the model obtains the ability to learn certain complex word relationships like vec(Japan) - vec(sushi) + vec(Germany) $\approx$ vec(bratwwurst) and vec(Einstein) - vec(scientist) + vec(Picasso) $\approx$ vec(painter).

2.2 Simple nBow representation

One naive method would be to consider the normalized bag-of-words (nBow) representation of documents, namely $d \in \mathbb{R}^n$, and then compute their distances. To be precise, if word i appears c_i times in the document, we denote $d_i = \frac{c_i}{\sum_{j=1}^{n} c_j}$. However we can imagine that this d must be very sparse as most words will not appear in any given document. And in fact, the author gave a simple example of comparing two sentences in one document: "Obama speaks to the media in Illinois" and "The President greets the press in Chicago". After stop-word removal, the two corresponding nBow vectors have no common non-zero dimensions while we can tell from the meanings of the two sentences directly that they are actually very close in reality.

2.3 Word Mover's Distance

So here comes to the play of Word Mover's Distance. The authors instead suggested looking at the "minimum travelling distance" from one word of document A to another word in document B, and then summing them up together, by utilizing the earth mover's distance (EMD). Figure 1 in the following illustrates the word mover's distance applied on the two sentences mentioned above. While figure 2 illustrates the WMD metric between a query D_0 and two sentences D_1 and D_2 (with equal BOW distance). We can see from the it that WMD did do a better job than the naive nBOW method when it's applied to the simple examples here.

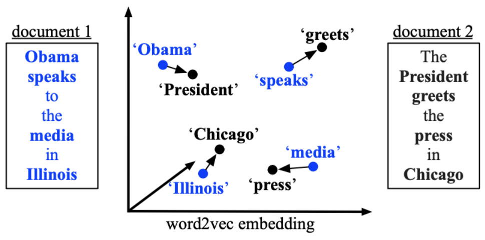
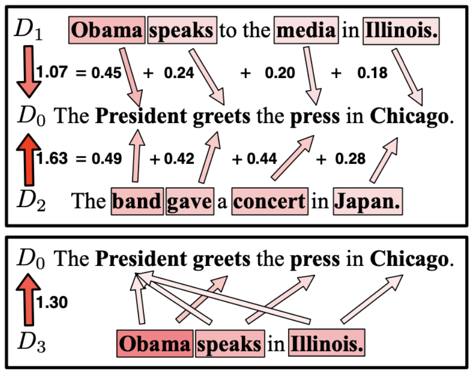


Figure 1. An illustration of the *word mover's distance.* All non-stop words (***bold***) of both documents are embedded into a *word2vec* space. The distance between the two documents is the minimum cumulative distance that all words in document 1 need to travel to exactly match document 2. (Best viewed in color.)



D_1 Obama speaks to the media in Illinois.
1.07 = 0.45 + 0.24 + 0.20 + 0.18
D_0 The President greets the press in Chicago.
1.63 = 0.49 + 0.42 + 0.44 + 0.28
D_2 The band gave a concert in Japan.
D_0 The President greets the press in Chicago.
1.30
D_3 Obama speaks in Illinois.

Figure 2. (Top:) The components of the WMD metric between a query D_0 and two sentences D_1, D_2 (with equal BOW distance). The arrows represent flow between two words and are labeled with their distance contribution. *(Bottom:)* The flow between two sentences D_3 and D_0 with different numbers of words. This mismatch causes the WMD to move words to multiple similar words.

The earth mover's distance, which is also the 1st Wasserstein distance, which is a metric between two probability measures. The transportation problem presented here is the following linear programming problem:

$$\min_{T\geq 0} \sum_{i,j=1}^{n} T_{ij} c(i,j)$$

$$\text{subject to: } \sum_{j=1}^{n} T_{ij} = d_i \quad \forall i \in \{1,\dots,n\}$$

$$\sum_{i=1}^{n} T_{ij} = d'_j \quad \forall j \in \{1,\dots,n\}$$

where $d = (d_1, \dots, d_n)$ and $d' = (d'_1, \dots, d'_n)$ are the nBOW representation of two documents. And $c_{ij} = \|x_i - x_j\|_2$ is the Euclidean distance of words i and j in the *word2vec* embedding space. This formulation is a special case of the earth mover's distance. To highlight this connection the authors call it as the *word mover's distance* (WMD).

2.3.1 Intuition of EMD

This function aims to find a joint distribution T to minimize the expected travelling distance from the words in document d to the words in document j, while satisfying the condition that this joint distribution $T = (T_{ij})_{i,j}$ has marginal distribution d and d'. One can illustrate this procedure as moving piles of earth from location A to location B, where A has piles with volumes $(A_1, A_2, \dots, A_n)$ respectively and location B has available piles to store earth with volume $(B_1, B_2, \dots, B_n)$. The idea is that you need to design a transporting strategy, to determine how to distribute the earth from each pile of location A to each pile of location B. For example, to transport pile A_i to location B, you need to determine how much of A_i will be put in location $B_1, B_2, \dots, B_n$, for $i = 1, 2, \dots, n$. Use T_{ij} to denote the volume of wood transporting from pile A_i to pile B_j, obviously T_i has to satisfy the marginal constraints given by the linear programming formulation above.

Furthermore, since moving the earth from A to B is not for free, for each unit of earth you move from pile A_i to B_j, you will have to pay the cost of $c(i,j)$, thus our goal is to minimize the total cost of moving the earth from location A to B, and it's very obvious that this total cost function should be

$$\sum_{i,j=1}^{n} T_{ij} c(i,j)$$

So that's how we can illustration the optimization problem in the above. And we can see from definition that this probability metric is a natural application to the document distance problem that we care about here.

3 Computation

Since the average time complexity for solving the WMD problem scales $O(p^2 logp)$ where p denotes the number of unique words in the documents. So the authors considered using several cheap lower bounds of the WMD problem to prune away the majority of the documents without computing the exact WMD distance.

3.1 Word Centroid distance

The first one they considered is *Word Centroid distance* $\|Xd - Xd'\|_2$, which is the distance between the weighted average word vectors, i.e.,

$$\begin{aligned}
\sum_{i,j} T_{ij} c(i,j) &= \sum_{i,j}^{n} T_{ij} \left\|x_i - x_j'\right\|_2 = \sum_{i,j}^{n} \left\|T_{ij}(x_i - x_j')\right\|_2 \\
&\geq \left\| \sum_{i,j=1}^{n} T_{ij}(x_i - x_j') \right\|_2 = \left\| \sum_{i=1}^{n} (\sum_{j=1}^{n} T_{ij}) x_i - \sum_{j=1}^{n} (\sum_{i=1}^{n} T_{ij}) x_j' \right\|_2 \\
&= \left\| \sum_{i=1}^{n} d_i x_i - \sum_{j=1}^{n} d_j' x_j' \right\|_2 = \|Xd - Xd'\|_2
\end{aligned}$$

This distance can be computed very fast and scales as $O(dp)$. However this is not a tight bound, which can be seen for the result computed on two datasets below, while RWMD (to be introduced below) well approximates the WMD.

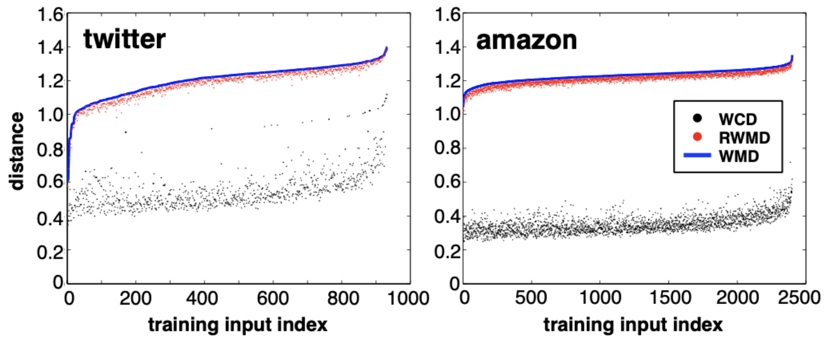


Figure 6. The WCD, RWMD, and WMD distances (sorted by WMD) for a random test query document.

3.2 Relaxed word moving distance

Another method where we can obtain much tighter bounds is achieved by relaxing the WMD optimization problem. For example if just the second constraint is removed then the optimization becomes

$$\min_{T\geq 0} \sum_{i,j=1}^{n} T_{ij} c(i,j)$$

$$\text{subject to: } \sum_{j=1}^{n} T_{ij} = d_i \quad \forall i \in \{1,\ldots,n\}$$

This optimal T^* can be computed as

$$T^*_{ij} = \begin{cases} d_i & \text{if } j = \arg\min_j c(i,j) \\ 0 & \text{otherwise} \end{cases}$$

where T^* would only result in the nearest neighbor search in the Euclidean *word2vec* space. And the result of relaxing the first constraint is similar by the symmetry of the problem. So let the two relaxed solutions be $l_1(d,d')$ and $l_2(d,d')$. We can obtain a even tighter bound by taking the maximum of the two $l_r(d,d') = \max(l_1(d,d'), l_2(d,d'))$, which is referred to as the Relaxed WMD (RWMD). And computation of this distance has a time complexity of $O(p^2)$.

3.3 Prefetch and prune

The authors first sort all documents in increasing order of the WCD distance, because this distance is cheap to compute and compute the exact WMD distance to the first k of these documents. For the remaining documents, they first check if the RWMD lower bound exceeds the distance of the current k-th closest document, if so then they can prove it. Otherwise they'll compute the WMD distance and update the k nearest neighbors if necessary.

The benefit of this method is that the cheap computation of WCD will allow a starting baseline set, and RWMD is a very tight, so can function as a good comparison to determine whether the WMD of a certain document to the query document is worth computing. So in this way the computational cost can be lowered significantly. An additional speedup can be obtained if the traversal across RWMD computation is limited to $m < n$ documents. If $m = k$, then this is equivalent to returning to the k nearest neighbors of the WCD distance.

4 Results

The authors evaluate WMD with a set of classic and state-of-the-art document representations and distances on 8 supervised document datasets. They compare against 7 document representation baselines, including bag-of-words (BOW), TFIDF, BM25 Okapsi, LSI Latent Semantic Indexing, LDA Latent Dirichlet Allocation, mSDA Marginalized Stacked Denoising Autoencoder, CCG Componential Counting Grid. It turns out that WMD outperforms all 7 alternative distances in 6 of 8 real world classification tasks, while for the remaining two, the result is ranked as 2nd, comparable to the best result on that specific task. The figure shown below justifies this by showing the test errors across different tasks.

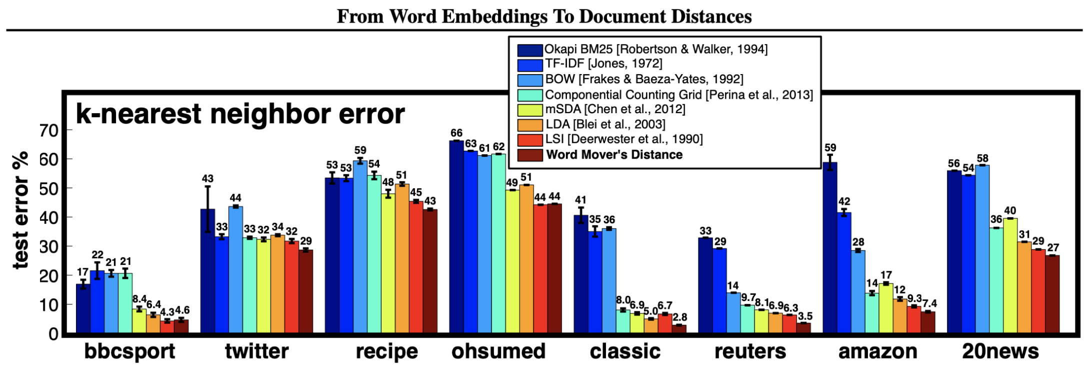


Figure 3. The *k*NN test error results on 8 document classification data sets, compared to canonical and state-of-the-art baselines methods.

The authors also gave the result of *prefetch and prune* algorithm under various values of m. When $m = k$ they use the WCD metric. For all other results they prefetch m instances via WCD, use RWMD to check if a document can be pruned and only if not, then compute the exact WMD distance. The authors noted that in all cases the increase in error through prefetching is relatively minor whereas the speedup can be substantial.

5 Discussion and directions of extensions

The WMD method utilizes the high quality of *word2vec* embedding. And as the authors point out, it's hyper-parameter free, highly interpretable as the distance between two documents can be broken down and explained as the sparse distances between few individual words. And it can lead to high retrieval accuracy as well.

However, since the metric is computational-expensive, the authors need to trigger the *prefetch and prune* approximation by utilizing WCD and RWMD. Although there's empirical results of the good approximation of WMD results, it would be great if there can be some theoretical analysis of lower bound of error of this kind of approximation.

And as the authors pointed out, it will also be interesting if interpretability of WMD can be further explored in future, like finding out a away to visualize and explain the sparse distances to humans, for example just like the nice vector meaning relationship obtained in *word2vec*. This might be feasible by incorporating document structure into the distances, etc.

Furthermore, it could be interesting to apply the idea of this algorithm by trying other word embedding methods, or maybe trying other probability metrics, like p-th Wasserstein distance $(p > 1)$, KL-divergence, Cramer distance, etc. to check whether we can get better results.

References

[1] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. From word embeddings to document distances. In *International conference on machine learning*, pages 957–966. PMLR, 2015.

[2] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *arXiv preprint arXiv:1310.4546*, 2013.