Finding Relevant Documents by Sampling over Word2vec Embeddings

Shiyu Ji
shiyu@cs.ucsb.edu

Abstract

All-pairs similarity search (APSS) has extensive applications in many fields such as collaborative filtering, recommendations, search query suggestions, spam detection, etc. This paper proposes an approximation algorithm for APSS over relevant documents. We propose the *averaged cosine similarity* between the word embeddings (e.g., by word2vec) of two documents as a measure of their relevance. We give the upper bounds of the approximation errors for the worst case by using Rademacher average. We also evaluate our algorithms by experiments on real-world data sets to verify that our upper bounds are tight enough for practical use.

1 Introduction

All-pairs similarity search (APSS) has received extensive research interest recently [7, 49, 1, 43]. Collaborative filtering [41], similarity-based recommendations [36], search query suggestions [12], spam and plagiarism detection [14, 30] all find APSS useful in practice. However, it is time consuming to do an APSS since the time complexity grows quadratically with the problem scale [7, 1, 43]. To improve performance, many approximation approaches have been proposed [20, 18, 24, 15, 27]. Hence the computation [7, 17, 49, 1, 43] and approximation [27, 20, 18, 24, 15] of the all-pairs similarities have been popular in research for more than ten years.

This paper considers a sampling-based approximation method for relevant document search. Thanks to word2vec [33, 32], we can evaluate the relevance between two words by computing the cosine similarity between their word embedding vectors. To evaluate the relevance between two documents, we propose to use the averaged cosine similarity among the word embeddings, i.e., we choose one word from document A at random, and choose another word from document B at random, and the expected cosine similarity between the chosen word embeddings should represent the relevance between these two documents. We treat the averaged cosine similarity between two documents as a true average over all the pairs of word embeddings, and then we can use sampled average to approximate the true average. We explain the details as follows.

We first define the relevance between two documents A and B. Suppose A have m words and B has n words. Denote by A_i the word embedding of the i-th word in A and similarily B_i is the i-th word in B. Then the relevance is defined as

$$Rel(A, B) = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} A_i \cdot B_j,$$

where $\cdot$ denotes the dot product between vectors, i.e., $A_i \cdot B_i = A_i^T B_i$. The product mn can be very large. Suppose we want to know the relevance between two long articles, each of which has more than 10K words. Then mn is larger than 100 million. Thus it is time consuming to traverse all the mn pairs of word embeddings and compute the dot products. However, if a precise approximation can also be acceptable, we can approximate the relevance (true average) by using the sampled average

$$\widetilde{Rel}(A, B) = \frac{1}{|S|} \sum_{(s_i, s_j) \in S} A_{s_i} \cdot B_{s_j},$$

where S is the set of sampled pairs of word embeddings, e.g., $(s_i, s_j) \in S$ means the s_i-th word in document A and the s_j-th word in document B are sampled. That is, we can choose k out of mn pairs at random, and compute the average of the k sampled pairs as an approximation of the relevance. Thanks to the recent advance in statistical learning theory (especially the results of Rademacher average), there are tight error bounds for this sampling method.

The approximation error of each our algorithm can be upper bounded by using Rademacher average [6, 34, 5]. We can treat the relevance approximation as a learning problem over large scale data set. In the analysis, we want to upper bound the approximation error for the worst case, i.e., to bound the maximum error $\overline{E}$ among the pairs of our observation:

$$\overline{E} = \max_{(A,B)\in\mathcal{P}} |\widetilde{Rel}(A,B) - Rel(A,B)|,$$

where P is the set of all the document pairs that we want to know their relevance values. Rademacher average can be used to bound such maximum error [39, 40]. The key result we use in this paper is that with probability at least $1-\delta$,

$$\overline{E} \le \min\left\{\sqrt{\frac{2\log 2p + 2\log(1/\delta)}{k}}, R + \left(1 + \sqrt{\frac{2}{k}\log\frac{1}{\delta}} + \sqrt{\frac{2}{k}\log\frac{1}{\delta} + 2R}\right)\sqrt{\frac{2\log\frac{4}{\delta}}{k}}\right\},$$

where $p = |P|$ (the observation size), R denotes the Rademacher average of the data set, k is the number of samples. It is also possible to upper bound the Rademacher average by the state-of-art results [3, 39, 40]. Thus we can find an upper bound of the worst errors for our approximation algorithms.

Our Contributions. We are the first to give the approximation algorithms for APSS on relevant documents based on word embeddings that can achieve upper bounds on maximum errors for the worst case, and to show the upper bounds by using Rademacher average. We also evaluate our algorithms by using real-world data sets.

The rest of this paper is organized as follows. Section 2 discusses the related works. Section 3 introduces the problem settings and reviews some technical background. Section 4 approximates the word embedding based document relevance and also gives the analysis on the upper bound of maximum errors. Section 5 evaluates our algorithms and presents the experimental results. Section 6 concludes this paper and discusses some possible future works.

2 Related Works

All-pairs similarity search (APSS) is recently a popular research topic in the community of information retrieval [7, 49, 1, 43]. Given a big set of objects, the goal of APSS is to efficiently compute (or approximate) the similarities between each pair of objects. Many similarity measures have been proposed [42], e.g., Cosine Similarity [44], SimRank [25], Jaccard Index [21], Metric Distance [42], Pearson Correlation [8], etc. For similarity search, Cosine Similarity and SimRank are very popular ([44, 49, 1, 43] for Cosine, and [28, 19, 26, 50] for SimRank). A major challenge of APSS is the large volume of computation: given n objects, without any assumption on the similarity distribution (e.g., the similarity matrix is sparse), the time complexity is at least $O(n^2)$. To compute more efficiently, the state-of-art research works often use parallelism [13, 22] and dissimilarity detection [16, 4, 27, 1, 43]. One popular method to eliminate dissimilar pairs is to hash vectors into buckets by Locality Sensitive Hashing (LSH) [27], and the candidate pairs are only between the vectors in the same bucket. However, since such methods can introduce non-negligible false positive and false negative [2, 45], further verifications on the candidate pairs (or missing potential pairs) are needed. Partition-based Similarity Search (PSS) uses partitions to cluster the candidate pairs [43]. Very often some partitions are still large, and hence further approximations on the similarities may be needed to avoid redundant computation. If only approximations are needed, how to efficiently sample with negligible error for similarity search is another research focus [20, 18, 24, 15]. This paper focuses on the approximation problem.

We may treat the similarity approximation as a learning problem over large scale data, and we need to upper bound the learning error for the worst case. In statistical learning theory, such upper bounds are usually called risk bounds [46]. There are two classical methods to compute the risk bounds: by VapnikChervonenkis (VC) dimension [47, 48, 46] and by Rademacher average [34, 6, 5]. Many approximation algorithms that use these two techniques have been proposed [37, 38, 39, 40]. Riondato and Kornaropoulos [37, 38] proposed algorithms that use VC dimension to upper bound the sample size that is sufficient to approximate the betweenness centralities [11] of all nodes in a graph with guaranteed error bounds. One limitation of the VC-based algorithms is that the upper bounds of some characteristic quantities (e.g., the maximum length of any shortest path) are needed [37, 38, 40]. But such bounds are not always available. One year later, Riondato and Upfal used Rademacher average to approximate the frequent itemsets [39] and betweenness centralities [40]. They also found that by using Rademacher average, we can avoid the aforementioned limitation of VC-based solutions. It has been proved that Rademacher average can be applied to various approximation problems, which are out of the scope of classical learning framework. This paper basically follows this idea. To the best of our knowledge, there is no research work connecting similarity approximation with Rademacher average. We attempt to fill this void.

3 Problem Formulation and Preliminaries

3.1 Cosine Similarity

We consider the cosine similarity based all-pairs similarity search. Suppose there are n vectors (each vector can represent a user profile or a web page). Each vector contains m non-negative features. Define the cosine similarity between two vectors u and v as

$$Sim(u,v) = \frac{1}{||u|| \cdot ||v||} \sum_{i=1}^{m} u_i \cdot v_i,$$

where u_i, v_i denotes the i-th feature value of u, v. For simplicity, we assume all the vectors are adjusted with the same norm: $||v|| = \sqrt{m}$ for every v in the n vectors. Then the equation above can be simplified as

$$Sim(u,v) = \frac{1}{m} \sum_{i=1}^{m} u_i \cdot v_i.$$

That is, the similarity is defined as the average on the corresponding feature products between the vectors.

To do the all-pairs similarity search (APSS), we need to compute the similarity between each pair of vectors. In the setting of word embedding, to compute the relevance between two documents: A with m words and B with n words, there are mn pairs of word embedding vectors to compute, and for each pair we need to compute the dot product, the total complexity of a naiive algorithm is $O(mnf)$, where f is the dimension of the word embedding, which is typically from 100 to 1000 [33]. Fortunately, there are many methods to detect dissimilar pairs (two vectors without sharing any feature) [1, 43, 29], which can save a lot of computation. One typical detection method is based on Locality Sensitive Hashing (LSH) [27]. We can construct a LSH such that for two vectors u and v, if their cosine similarity is larger than $\cos\theta$, then with probability at least $1 - \theta/\pi$ we can hash u and v into one bucket [27]. The vectors in one bucket are likely to be highly similar. Thus the number of vector pairs to be compared is greatly reduced. For the state-of-art works, to compute all the pairs, the complexity can be lowered to $O(nkm)$, where k is much less than n. However, to the best of our knowledge, there were few discussions on the size of features m. If we can only consider a part of the m features without significantly deteriorating the accuracy, then the total computation time for APSS can be lowered significantly (note that nk is still large for very big dataset). Moreover, the popular LSH based method can introduce false positive. Note that if we assume the cosine similarity ranges from 0 to 1, then the angle θ between any two vectors is between 0 and $\pi/2$. Hence even for two totally dissimilar vectors $u \cdot v = 0$, they still have probability at least $1 - (1 - 0.5^r)^b$ (where r and b are the number of rows and bands that from the LSH signature [27]) to be hashed into one bucket and thus will be falsely believed to be similar. Hence we need further verifications on the candidate pairs. Since there could still be a lot of vectors in one bucket, in practice approximations on the cosine similarities are interesting to research.

3.2 Rademacher Average

This section will review some key results related to Rademacher Average. Suppose we want to compute the cosine similarities between a fixed vector u and other vectors v_1, v_2, $\cdots$, v_n. We take the m features as the sample space S:

$$S = \{s_1, \cdots, s_k\} \subseteq D$$

where k is the number of samples we take, and D is the feature space: $D = \{1, 2, \cdots, m\}$. Let F be a collection of functions from the features D to the interval $[0, m]$. For each function $f \in F$, define the *true average* $A_D(f)$ and *sampled average* $A_S(f)$ as follows:

$$A_D(f) = \frac{1}{m}\sum_{i=1}^{m} f(i), \quad A_S(f) = \frac{1}{k}\sum_{i=1}^{k} f(s_i).$$

Define the *uniform deviation* [35] of F given S as

$$U_S(F) = \sup_{f\in F}[A_S(f) - A_D(f)].$$

Note that if F is a finite set (in this paper we will see this is true), supreme can be replaced by maximum:

$$U_S(F) = \max_{f\in F}[A_S(f) - A_D(f)].$$

For the case when F is finite, we have the upper bound of the approximation error in the worst case as follows.

Theorem 1. If F is finite, then for any $\delta > 0$, with probability at least $1 - \delta$,

$$\sup_{f\in F} |A_S(f) - A_D(f)| \leq \sqrt{\frac{2\log(2|F|) + 2\log(1/\delta)}{k}}.$$

We leave the proof of Theorem 1 to the Appendix.

However if F is not necessarily infinite, the above bound cannot apply. We can use Rademacher average to deal with the general case. Define the *Rademacher average* [34, 6, 35] of F given S as

$$R_S(F) = \mathbb{E}_\sigma\left[\sup_{f\in F}\frac{2}{k}\sum_{i=1}^{k}\sigma_i f(s_i)\right],$$

where each σ_i is a random variable uniformly distributed over $\{-1, 1\}$, and the mean $\mathbb{E}_\sigma$ takes randomness over all the σ_i's conditionally on S. Again, one can replace supreme by maximum if F is finite.

The main motivation of our algorithm is that the difference between true average and sampled average is upper bounded by the Rademacher average as follows. We leave the proofs to the Appendix.

Theorem 2. Let F be a collection of functions f mapping D to $[-1, 1]$. With probability at least $1 - \delta$, we have

$$\sup_{f\in F} |A_S(f) - A_D(f)| \leq R_S(F) + \left(1 + \sqrt{\frac{2}{k}\log\frac{1}{\delta}} + \sqrt{\frac{2}{k}\log\frac{1}{\delta} + 2R_S(F)}\right)\sqrt{\frac{2\log\frac{8}{\delta}}{k}}.$$

The remaining item we need to upper bound is $R_S(F)$. Our result is similar to Massart's lemma [3].

Theorem 3.

$$R_S(F) \leq \frac{\ell}{k}\sqrt{8\log|F|},$$

where $\ell^2 = \sup_{f\in F}\sum_{i=1}^{k} f(s_i)^2$.

By the above two theorems, we can bound our approximation error even for the worst case.

4 Approximating Document Relevance

This section proposes our approximation algorithm and its analysis. The approximation algorithm takes as input a collection of word embedding vectors V, each of which is associated with one word, and a collection of p candidate document pairs P, and two parameters (ϵ, δ) whose values are between 0 and 1. The candidate pairs P can be given by a dissimilarity detection algorithm, e.g., LSH based method [27], Feature Scoring [16], Diamond Sampling [4]. The algorithm outputs a set $C = \{\tilde{S}(u, v) : (u, v) \in P\}$, where $\tilde{S}(u, v)$ is the (ϵ, δ)-approximation of relevance between u and v, i.e., with probability at least $1 - \delta$, the worst approximation error in C is at most ϵ. Each vector in V contains some features. Suppose we have mn of word embedding pairs. We take these mn vector pairs as the total space $D = \{(1, 1), (1, 2), \cdots, (m, n)\}$. For each pair $s = (i, j) \in D$, let $f_{u,v}(s) = u_i \cdot v_j$, where u_i is the embedding vector of the i-th word of the document u. Let $F = \{f_{u,v} : (u, v) \in P\}$. Thus $|F| = p$. It is clear that the true average of each $f_{u,v}$ equals to the document relevance between u and v. Given the sampled features of size k, the upper bound of $R_S(F)$ is

$$R_S(F) \leq \frac{\ell}{k}\sqrt{8 \log p},$$

where $\ell = \sqrt{\max_{f \in F} \sum_{i=1}^{k} f(s_i)^2}$. Since F is finite, we can replace supreme by maximum.

4.1 The Algorithm

The approximation algorithm works in an iterative mode. For each iteration, we sample some new vector pairs s_i's among D and aggregate the feature products given by $f_{u,v}(s_i)$ for each $f_{u,v}$ in F. Then we compute the error upper bound:

$$\Delta = \min\left\{\sqrt{\frac{2\log 2p + 2\log(1/\delta)}{k}}, \frac{\ell\sqrt{8\log p}}{k} + \left(1 + \sqrt{\frac{2}{k}\log\frac{1}{\delta}} + \sqrt{\frac{2}{k}\log\frac{1}{\delta} + \frac{4\ell\sqrt{2\log p}}{k}}\right)\sqrt{\frac{2\log\frac{8}{\delta}}{k}}\right\},$$

where $\ell = \sqrt{\max_{f \in F} \sum_{i=1}^{k} f(s_i)^2}$, and k is the number of aggregated samples. If $\Delta \leq \epsilon$, then we stop and return the average $\frac{1}{k}\sum_{s_i \in S} f_{u,v}(s_i)$ for each pair u, v. Otherwise, we continue to the next round, where we will sample more features. If $\Delta \leq \epsilon$ can never be satisfied, we will sample at most m^2 vector pairs, where m is the maximum number of words in any document in the dataset. Algorithm 1 gives the pseudocode of our solution. We use progressive approximation, i.e., for each round, the next sample size k' is larger than the previous size k. The algorithm stops once the upper bound ϵ can be achieved.

It is clear to verify the correctness of our algorithm by Theorem 1, Theorem 2 and Theorem 3.

5 Evaluation

6 Conclusion

References

[1] Maha Alabduljalil, Xun Tang, and Tao Yang. Cache-conscious performance optimization for similarity search. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 713–722. ACM, 2013.

[2] Maha Ahmed Alabduljalil, Xun Tang, and Tao Yang. Optimizing parallel algorithms for all pairs similarity search. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 203–212. ACM, 2013.

Algorithm 1 Document Relevance Approximation

```
Input: collection of documents $D$; word embedding vectors $V$; candidate document pairs $P$ ($|P| = p$);
    maximum number of words in any document $m$; $\epsilon \in (0,1)$; $\delta \in (0,1)$.
Output: $\widetilde{S}(u,v)$ for each $(u,v) \in P$.
    $k \leftarrow 0$; $S \leftarrow \emptyset$;
    For each $i, j \in D$, let $S(i,j) = 0$;
    while $k < m^2$ do
        $k' \leftarrow$ next-sample-size($k$);
        for $i$ from $k$ to $k' - 1$ do
            for each $(u,v) \in P$ do
                Uniformly sample a vector $u_s$ from the embedding vectors of $u$;
                Uniformly sample a vector $v_s$ from the embedding vectors of $v$;
                $S(u,v) \leftarrow S(u,v) + u_s \cdot v_s$;
            end for
        end for
        $k \leftarrow k'$;
        $\Delta \leftarrow \min\left\{\sqrt{\frac{2\log 2p + 2\log(1/\delta)}{k}}, \frac{\ell\sqrt{8\log p}}{k} + \left(1 + \sqrt{\frac{2}{k}\log\frac{1}{\delta}} + \sqrt{\frac{2}{k}\log\frac{1}{\delta} + \frac{4\ell\sqrt{2\log p}}{k}}\right)\sqrt{\frac{2\log\frac{8}{\delta}}{k}}\right\}$;
        if $\Delta \leq \epsilon$ then
            break the while loop;
        end if
    end while
    $\mathsf{S} \leftarrow \{S(u,v)/k : u, v \in V, (u,v) \in P\}$.
    return $\mathsf{S}$.
```

[3] Davide Anguita, Alessandro Ghio, Luca Oneto, and Sandro Ridella. A deep connection between the vapnik–chervonenkis entropy and the rademacher complexity. *IEEE transactions on neural networks and learning systems*, 25(12):2202–2211, 2014.

[4] Grey Ballard, Tamara G Kolda, Ali Pinar, and C Seshadhri. Diamond sampling for approximate maximum all-pairs dot-product (mad) search. In *Data Mining (ICDM), 2015 IEEE International Conference on*, pages 11–20. IEEE, 2015.

[5] Peter L Bartlett, Olivier Bousquet, and Shahar Mendelson. Local rademacher complexities. *Annals of Statistics*, pages 1497–1537, 2005.

[6] Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

[7] Roberto J Bayardo, Yiming Ma, and Ramakrishnan Srikant. Scaling up all pairs similarity search. In *Proceedings of the 16th international conference on World Wide Web*, pages 131–140. ACM, 2007.

[8] Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. Pearson correlation coefficient. In *Noise reduction in speech processing*, pages 1–4. Springer, 2009.

[9] S Boucheron, G Lugosi, and Pascal Massart. A sharp concentration inequality with applications. *Random Structures and Algorithms*, 1999.

[10] Olivier Bousquet, Stéphane Boucheron, and Gábor Lugosi. Introduction to statistical learning theory. In *Advanced lectures on machine learning*, pages 169–207. Springer, 2004.

[11] Ulrik Brandes. A faster algorithm for betweenness centrality*. *Journal of mathematical sociology*, 25(2):163–177, 2001.

[12] Huanhuan Cao, Daxin Jiang, Jian Pei, Qi He, Zhen Liao, Enhong Chen, and Hang Li. Context-aware query suggestion by mining click-through and session data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 875–883. ACM, 2008.

[13] Liangliang Cao, Brian Cho, Hyun Duk Kim, Zhen Li, Min-Hsuan Tsai, and Indranil Gupta. Delta-simrank computing on mapreduce. In *Proceedings of the 1st International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications*, pages 28–35. ACM, 2012.

[14] Carlos Castillo, Debora Donato, Aristides Gionis, Vanessa Murdock, and Fabrizio Silvestri. Know your neighbors: Web spam detection using the web topology. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 423–430. ACM, 2007.

[15] Moses S Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 380–388. ACM, 2002.

[16] Edith Cohen and David D Lewis. Approximating matrix multiplication for pattern recognition tasks. *Journal of Algorithms*, 30(2):211–252, 1999.

[17] Xin Dong, Alon Halevy, Jayant Madhavan, Ema Nemes, and Jun Zhang. Similarity search for web services. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 372–383. VLDB Endowment, 2004.

[18] Ronald Fagin, Ravi Kumar, and Dandapani Sivakumar. Efficient similarity search and classification via rank aggregation. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 301–312. ACM, 2003.

[19] Yasuhiro Fujiwara, Makoto Nakatsuji, Hiroaki Shiokawa, and Makoto Onizuka. Efficient search algorithm for simrank. In *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*, pages 589–600. IEEE, 2013.

[20] Aristides Gionis, Piotr Indyk, Rajeev Motwani, et al. Similarity search in high dimensions via hashing. *VLDB*, 99(6):518–529, 1999.

[21] Lieve Hamers, Yves Hemeryck, Guido Herweyers, Marc Janssen, Hans Keters, Ronald Rousseau, and André Vanhoutte. Similarity measures in scientometric research: the jaccard index versus salton's cosine formula. *Information Processing & Management*, 25(3):315–318, 1989.

[22] Guoming He, Haijun Feng, Cuiping Li, and Hong Chen. Parallel simrank computation on large graphs with iterative aggregation. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 543–552. ACM, 2010.

[23] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.

[24] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613. ACM, 1998.

[25] Glen Jeh and Jennifer Widom. Simrank: a measure of structural-context similarity. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 538–543. ACM, 2002.

[26] Mitsuru Kusumoto, Takanori Maehara, and Ken-ichi Kawarabayashi. Scalable similarity search for simrank. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 325–336. ACM, 2014.

[27] Jure Leskovec, Anand Rajaraman, and Jeffrey David Ullman. *Mining of massive datasets*. Cambridge University Press, 2014.

[28] Cuiping Li, Jiawei Han, Guoming He, Xin Jin, Yizhou Sun, Yintao Yu, and Tianyi Wu. Fast computation of simrank for static and dynamic information networks. In *Proceedings of the 13th International Conference on Extending Database Technology*, pages 465–476. ACM, 2010.

[29] Jimmy Lin. Brute force and indexed approaches to pairwise document similarity comparisons with mapreduce. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 155–162. ACM, 2009.

[30] Chao Liu, Chen Chen, Jiawei Han, and Philip S Yu. Gplag: detection of software plagiarism by program dependence graph analysis. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 872–881. ACM, 2006.

[31] Colin McDiarmid. On the method of bounded differences. *Surveys in combinatorics*, 141(1):148–188, 1989.

[32] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[33] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[34] Mehryar Mohri and Afshin Rostamizadeh. Rademacher complexity bounds for non-iid processes. In *Advances in Neural Information Processing Systems*, pages 1097–1104, 2009.

[35] Luca Oneto, Alessandro Ghio, Davide Anguita, and Sandro Ridella. An improved analysis of the rademacher data-dependent bound using its self bounding property. *Neural Networks*, 44:107–111, 2013.

[36] Paul Resnick and Hal R Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.

[37] Matteo Riondato and Evgenios M Kornaropoulos. Fast approximation of betweenness centrality through sampling. In *Proceedings of the 7th ACM international conference on Web search and data mining*, pages 413–422. ACM, 2014.

[38] Matteo Riondato and Evgenios M Kornaropoulos. Fast approximation of betweenness centrality through sampling. *Data Mining and Knowledge Discovery*, 30(2):438–475, 2016.

[39] Matteo Riondato and Eli Upfal. Mining frequent itemsets through progressive sampling with rademacher averages. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1005–1014. ACM, 2015.

[40] Matteo Riondato and Eli Upfal. Abra: Approximating betweenness centrality in static and dynamic graphs with rademacher averages. *arXiv preprint arXiv:1602.05866*, 2016.

[41] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001.

[42] Alexander Strehl, Joydeep Ghosh, and Raymond Mooney. Impact of similarity measures on web-page clustering. In *Workshop on Artificial Intelligence for Web Search (AAAI 2000)*, pages 58–64, 2000.

[43] Xun Tang, Maha Alabduljalil, Xin Jin, and Tao Yang. Load balancing for partition-based similarity search. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 193–202. ACM, 2014.

[44] Sandeep Tata and Jignesh M Patel. Estimating the selectivity of tf-idf based cosine similarity predicates. *ACM Sigmod Record*, 36(2):7–12, 2007.

[45] Ferhan Ture, Tamer Elsayed, and Jimmy Lin. No free lunch: brute force vs. locality-sensitive hashing for cross-lingual pairwise similarity. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 943–952. ACM, 2011.

[46] Vladimir Vapnik. *The nature of statistical learning theory.* Springer Science & Business Media, 2013.

[47] Vladimir Vapnik, Esther Levin, and Yann Le Cun. Measuring the vc-dimension of a learning machine. *Neural Computation*, 6(5):851–876, 1994.

[48] Vladimir Naumovich Vapnik and Vlamimir Vapnik. *Statistical learning theory*, volume 1. Wiley New York, 1998.

[49] Zhihua Xia, Xinhui Wang, Xingming Sun, and Qian Wang. A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data. *IEEE Transactions on Parallel and Distributed Systems*, 27(2):340–352, 2016.

[50] Weiren Yu and Julie Ann McCann. High quality graph-based similarity search. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 83–92. ACM, 2015.

7 Appendix

7.1 Proof of Theorem 1

First, for any $\epsilon > 0$,

$$\begin{aligned}\Pr[U_S(F) > \epsilon] &= \Pr[\exists f \in F, A_S(f) - A_D(f) > \epsilon] \\ &\leq \sum_{f\in F} \Pr[A_S(f) - A_D(f) > \epsilon] \\ &= \sum_{f\in F} \Pr[\frac{1}{k}\sum_{i=1}^{k} f(s_i) - A_D(f) > \epsilon].\end{aligned}$$

By Hoeffding's inequality [10] we have

$$\Pr[\frac{1}{k}\sum_{i=1}^{k} f(s_i) - A_D(f) > \epsilon] \leq \exp\left(-\frac{2k\epsilon^2}{2^2}\right).$$

Hence

$$\Pr[U_S(F) > \epsilon] \leq |F| \exp\left(-\frac{k\epsilon^2}{2}\right).$$

Similarly one can show that

$$\Pr[\sup_{f\in F}[A_D(f) - A_S(f)] > \epsilon] \leq |F| \exp\left(-\frac{k\epsilon^2}{2}\right).$$

Thus putting the two cases together,

$$\Pr[\sup_{f\in F} |A_S(f) - A_D(f)| > \epsilon] \leq 2|F| \exp\left(-\frac{k\epsilon^2}{2}\right).$$

Equivalently for any $\delta > 0$, with probability at least $1 - \delta$,

$$\sup_{f \in F} |A_S(f) - A_D(f)| \leq \sqrt{\frac{2\log(2|F|) + 2\log(1/\delta)}{k}}.$$

We have obtained the desired upper bound in Theorem 1.

7.2 Proof of Theorem 2

In this section, we show our main result (Theorem 2). We start from the definition of self bounding function [35].

Definition 1. *Let $s_1, s_2, \cdots, s_k$ be independent random variables taking values from a set D. A function $f : D^k \to [0, +\infty]$ is a self bounding function if there exists a constant c and a function $g : D^{k-1} \to \mathbb{R}$ such that for any $s_1, \cdots, s_{j-1}, s_{j+1}, \cdots, s_k \in D$, the following conditions hold:*

$$0 \leq f(s_1, \cdots, s_k) - g(s_1, \cdots, s_{j-1}, s_{j+1}, \cdots, s_k) \leq c,$$

$$\sum_{j=1}^{k} [f(s_1, \cdots, s_k) - g(s_1, \cdots, s_{j-1}, s_{j+1}, \cdots, s_k)] \leq f(s_1, \cdots, s_k).$$

The following concentration inequality can be achieved for self bounding functions [9].

Lemma 1. *[9] If a function $Z = f(s_1, \cdots, s_k)$ is a self bounding function with constant c, then for $t \leq \mathbb{E}Z$,*

$$\Pr[\mathbb{E}Z - Z \geq t] \leq \exp\left(-\frac{t^2}{2c\mathbb{E}Z}\right).$$

For $t > \mathbb{E}Z$, the left probability is zero trivially. Here we take randomness over $s_1, s_2, \cdots, s_k$.

By using the above lemma, we can show a similar inequality for Rademacher average.

Lemma 2.

$$\Pr[\mathbb{E}R_S(F) \geq R_S(F) + t] \leq \exp\left(-\frac{kt^2}{4\mathbb{E}R_S(F)}\right),$$

where $\mathbb{E}$ takes randomness over the samplings $s_1, s_2, \cdots, s_k$.

Proof. It suffices to show that $R_S(F)$ is a self bounding function with constant $c = 2/k$. Define

$$Z = R_S(F) = \mathbb{E}_\sigma \sup_{f \in F} \left[\frac{2}{k} \sum_{i=1}^{k} \sigma_i f(s_i)\right],$$

$$G_j = \mathbb{E}_\sigma \sup_{f \in F} \left[\frac{2}{k} \sum_{i \neq j} \sigma_i f(s_i)\right].$$

It is clear that Z is non-negative:

$$Z \geq \sup_{f \in F} \left[\mathbb{E}_\sigma \frac{2}{k} \sum_{i=1}^{k} \sigma_i f(s_i)\right] = 0.$$

Also it is clear that $Z \geq G_j$ for each j: suppose $\tilde{f}$ achieves the supreme of G_j. Then

$$\begin{aligned} G_j &= \mathbb{E}_\sigma \left[\frac{2}{k} \sum_{i=1}^{k} \sigma_i \tilde{f}(s_i) - \frac{2}{k} \sigma_j \tilde{f}(s_j) \right] \\ &= \mathbb{E}_\sigma \left[\frac{2}{k} \sum_{i=1}^{k} \sigma_i \tilde{f}(s_i) \right] - \mathbb{E}_\sigma \left[\frac{2}{k} \sigma_j \tilde{f}(s_j) \right] \\ &= \mathbb{E}_\sigma \left[\frac{2}{k} \sum_{i=1}^{k} \sigma_i \tilde{f}(s_i) \right] \leq Z. \end{aligned}$$

Next we show $Z - G_j \leq 2m/k = c$:

$$\begin{aligned} G_j &= \mathbb{E}_\sigma \sup_{f \in F} \left[\frac{2}{k} \sum_{i=1}^{k} \sigma_i f(s_i) - \frac{2}{k} \sigma_j f(s_j) \right] \\ &\geq \mathbb{E}_\sigma \sup_{f \in F} \left[\frac{2}{k} \sum_{i=1}^{k} \sigma_i f(s_i) \right] - \mathbb{E}_\sigma \sup_{f \in F} \left[\frac{2}{k} \sigma_j f(s_j) \right] \\ &\geq \mathbb{E}_\sigma \sup_{f \in F} \left[\frac{2}{k} \sum_{i=1}^{k} \sigma_i f(s_i) \right] - \frac{2}{k}. \end{aligned}$$

Finally we need to verify $\sum_{j=1}^{k} Z - G_j \leq Z$:

$$\begin{aligned} \sum_{j=1}^{k} G_j &= \mathbb{E}_\sigma \sum_{j=1}^{k} \sup_{f \in F} \left[\frac{2}{k} \sum_{i \neq j} \sigma_i f(s_i) \right] \\ &\geq \mathbb{E}_\sigma \sup_{f \in F} \left[\frac{2}{k} \sum_{j=1}^{k} \sum_{i \neq j} \sigma_i f(s_i) \right] \\ &= \frac{2(k-1)}{k} \mathbb{E}_\sigma \sup_{f \in F} \left[\sum_{j=1}^{k} \sigma_i f(s_i) \right] = (k-1)Z. \end{aligned}$$

□

We still need the following lemma on the relation between uniform deviation and Rademacher average.

Lemma 3.

$$\mathbb{E} \sup_{f \in F} [A_S(f) - A_D(f)] \leq \mathbb{E} R_S(F),$$

$$\mathbb{E} \sup_{f \in F} [A_D(f) - A_F(f)] \leq \mathbb{E} R_S(F).$$

Here we take randomness over the k samplings.

Proof. The proof idea is based on ghost samplings, i.e., independently draw another k samples: $s'_1, \cdots, s'_k$, and then we have

$$A_D(f) = \frac{1}{m} \sum_{i=1}^{m} f(i) = \mathbb{E} \frac{1}{k} \sum_{j=1}^{k} f(s'_j),$$

where $\mathbb{E}$ takes randomness over the k ghost samples. Thus

$$\begin{aligned}\mathbb{E}\sup_{f\in F}[A_S(f)-A_D(f)] &= \mathbb{E}\sup_{f\in F}\left[\frac{1}{k}\sum_{i=1}^{k} f(s_i) - \mathbb{E}\frac{1}{k}\sum_{j=1}^{k} f(s'_j)\right]\\ &\le \mathbb{E}\sup_{f\in F}\left[\frac{1}{k}\sum_{i=1}^{k} f(s_i) - \frac{1}{k}\sum_{j=1}^{k} f(s'_j)\right].\end{aligned}$$

Since all the samples s, s' are independently identically distributed, flipping the sign of $f(s_i) - f(s'_i)$ will not change the expected supreme, i.e.,

$$\mathbb{E}\sup_{f\in F}\left[\frac{1}{k}\sum_{i=1}^{k} f(s_i) - \frac{1}{k}\sum_{j=1}^{k} f(s'_j)\right] = \mathbb{E}\sup_{f\in F}\frac{1}{k}\sum_{i=1}^{k}\left[\sigma_i(f(s_i)-f(s'_i))\right],$$

where σ_i is uniformly distributed over $\{-1,1\}$. Since

$$\mathbb{E}\sup_{f\in F}\frac{1}{k}\sum_{i=1}^{k}\left[\sigma_i(f(s_i)-f(s'_i))\right] \le 2\mathbb{E}\sup_{f\in F}\frac{1}{k}\sum_{i=1}^{k}\sigma_i f(s_i) = \mathbb{E}R_S(F),$$

we have shown the first inequality. The second inequality is analogous. $\square$

We also need McDiarmid's inequality [31].

Lemma 4. *[31] Let $s_1, \cdots, s_k$ be independent random variables taking values from a set D. Suppose a function $h : D^k \to \mathbb{R}$ satisfies*

$$\sup_{x_1,\cdots,x_k,x'_i\in D} |h(x_1,\cdots,x_k) - h(x_1,\cdots,x_{i-1},x'_i,x_{i+1},\cdots,x_k)| \le c_i$$

for some constants c_i and every $1 \le i \le k$. Then for any $t > 0$, we have

$$\Pr[h(s_1,\cdots,s_k) - \mathbb{E}h(s_1,\cdots,s_k) \ge t] \le \exp\left(-\frac{2t^2}{\sum_{i=1}^{k} c_i^2}\right).$$

By the above three lemmas, we can bound the difference between true average and sampled average as follows.

Lemma 5.

$$\begin{aligned}&\Pr\left[\sup_{f\in F}|A_D(f)-A_S(f)| \ge R_S(F)+t\right]\\ \le& 4\exp\left(-\frac{kt^2}{2(1+\sqrt{2\mathbb{E}R_S(F)})^2}\right).\end{aligned}$$

Proof. First by Lemma 3,

$$\begin{aligned}&\Pr\left[\sup_{f\in F}[A_D(f)-A_S(f)] \ge R_S(F)+t\right]\\ \le& \Pr\left[\sup_{f\in F}[A_D(f)-A_S(f)] \ge \mathbb{E}\sup_{f\in F}[A_D(f)-A_S(f)] + at\right]\\ &+ \Pr\left[\mathbb{E}R_S(F) \ge R_S(F) + (1-a)t\right]\end{aligned}$$

for any $a \in [0,1]$. Let

$$h(s_1, \cdots, s_k) = A_D(f) - A_S(f) = A_D(f) - \frac{1}{k}\sum_{i=1}^{k} f(s_i).$$

It is clear that

$$\begin{aligned}
&\sup_{x_1,\cdots,x_k,x_i'\in D} |h(x_1,\cdots,x_k) - h(x_1,\cdots,x_{i-1},x_i',x_{i+1},\cdots,x_k)| \\
=&\sup_{x_1,\cdots,x_k,x_i'\in D} \left|\frac{1}{k}\sum_{j=1,j\neq i}^{k} f(x_j) + \frac{1}{k}f(x_i') - \frac{1}{k}\sum_{i=1}^{k} f(x_i)\right| \\
=&\sup_{x_1,\cdots,x_k,x_i'\in D} \left|\frac{1}{k}f(x_i') - \frac{1}{k}f(x_i)\right| \le \frac{2}{k}.
\end{aligned}$$

By McDiarmid's inequality,

$$\begin{aligned}
&\Pr\left[\sup_{f\in F}[A_D(f) - A_S(f)] \ge \mathbb{E}\sup_{f\in F}[A_D(f) - A_S(f)] + at\right] \\
\le&\exp\left(-\frac{2a^2t^2}{\sum_{i=1}^{k} 2^2/k^2}\right) = \exp\left(-\frac{ka^2t^2}{2}\right).
\end{aligned}$$

By Lemma 2,

$$\Pr\left[\mathbb{E}R_S(F) \ge R_S(F) + (1-a)t\right] \le \exp\left(-\frac{k(1-a)^2t^2}{4\mathbb{E}R_S(F)}\right).$$

Let $a = 1/(1+\sqrt{2\mathbb{E}R_S(F)})$. Then putting everything together, we have

$$\begin{aligned}
&\Pr\left[\sup_{f\in F}[A_D(f) - A_S(f)] \ge R_S(F) + t\right] \\
\le&2\exp\left(-\frac{kt^2}{2(1+\sqrt{2\mathbb{E}R_S(F)})^2}\right).
\end{aligned}$$

Similarly one can show

$$\begin{aligned}
&\Pr\left[\sup_{f\in F}[A_S(f) - A_D(f)] \ge R_S(F) + t\right] \\
\le&2\exp\left(-\frac{kt^2}{2(1+\sqrt{2\mathbb{E}R_S(F)})^2}\right).
\end{aligned}$$

Thus we have the inequality as desired. □

By the above lemma we have the following important corollary.

Corollary 1. *With probability at least* $1-\delta$, *we have*

$$\sup_{f\in F} |A_S(f) - A_D(f)| \le R_S(F) + (1+\sqrt{2\mathbb{E}R_S(F)})\sqrt{\frac{2\log\frac{4}{\delta}}{k}}.$$

We still need to upper bound $\mathbb{E}R_S(F)$. By Lemma 2, with probability at least $1-\delta$,

$$\mathbb{E}R_S(F) \le R_S(F) + \sqrt{4\mathbb{E}R_S(F)\frac{\log\frac{1}{\delta}}{k}}.$$

Or equivalently,

$$\sqrt{\mathbb{E}R_S(F)} \leq \sqrt{\frac{1}{k}\log\frac{1}{\delta}} + \sqrt{\frac{1}{k}\log\frac{1}{\delta} + R_S(F)}.$$

Hence with probability at least $1-2\delta$, we have

$$\sup_{f\in F}|A_S(f) - A_D(f)| \leq R_S(F) + \left(1 + \sqrt{\frac{2}{k}\log\frac{1}{\delta}} + \sqrt{\frac{2}{k}\log\frac{1}{\delta} + 2R_S(F)}\right)\sqrt{\frac{2\log\frac{4}{\delta}}{k}}.$$

We have shown Theorem 2.

7.3 Proof of Theorem 3

For any $s > 0$, by Jensen's inequality,

$$\begin{aligned}\exp(skR_S(F)) &= \exp\left(2s\mathbb{E}_\sigma \sup_{f\in F}\sum_{i=1}^k \sigma_i f(s_i)\right)\\ &\leq \mathbb{E}_\sigma \exp\left(2s\sup_{f\in F}\sum_{i=1}^k \sigma_i f(s_i)\right)\\ &\leq \mathbb{E}_\sigma \sum_{f\in F}\exp\left(2s\sum_{i=1}^k \sigma_i f(s_i)\right).\end{aligned}$$

By Hoeffding's Lemma [23],

$$\begin{aligned}&\mathbb{E}_\sigma \sum_{f\in F}\exp\left(2s\sum_{i=1}^k \sigma_i f(s_i)\right)\\ &\leq \sum_{f\in F}\prod_{i=1}^k \exp\left(2s^2 f(s_i)^2\right)\\ &= \sum_{f\in F}\exp\left(2s^2\sum_{i=1}^k f(s_i)^2\right).\end{aligned}$$

Let $\ell^2 = \sup_{f\in F}\sum_{i=1}^k f(s_i)^2$, and then

$$\sum_{f\in F}\exp\left(2s^2\sum_{i=1}^k f(s_i)^2\right) \leq |F|\exp\left(2s^2\ell^2\right).$$

Thus

$$R_S(F) \leq \frac{1}{sk}(\log|F| + 2s^2\ell^2),$$

for any $s > 0$. It turns out that to minimize the right hand side of the above equation, we have

$$s = \sqrt{\frac{\log|F|}{2\ell^2}}.$$

Then

$$R_S(F) \leq \frac{\ell}{k}\sqrt{8\log|F|}.$$