BIOINFORMATICS ORIGINAL PAPER

Vol. 23 no. 5 2007, pages 612–618
doi:10.1093/bioinformatics/btl645

Data and text mining

Speeding up tandem mass spectrometry database search: metric embeddings and fast near neighbor search

Debojyoti Dutta*, Ting Chen

Department of Computational Biology, University of Southern California, 1050 Childs Way, Los Angeles, CA 90089, USA

Received on March 28, 2006; revised on December 16, 2006; accepted on December 18, 2006

Advance Access publication January 19, 2007

Associate Editor: Alfonso Valencia

ABSTRACT

Motivation: Due to the recent advances in technology of mass spectrometry, there has been an exponential increase in the amount of data being generated in the past few years. Database searches have not been able to keep with this data explosion. Thus, speeding up the data searches becomes increasingly important in mass-spectrometry-based applications. Traditional database search methods use one-against-all comparisons of a query spectrum against a very large number of peptides generated from *in silico* digestion of protein sequences in a database, to filter potential candidates from this database followed by a detailed scoring and ranking of those filtered candidates.

Results: In this article, we show that we can avoid the one-against-all comparisons. The basic idea is to design a set of hash functions to pre-process peptides in the database such that for each query spectrum we can use the hash functions to find only a small subset of peptide sequences that are most likely to match the spectrum. The construction of each hash function is based on a random spectrum and the hash value of a peptide is the normalized shared peak counts score (cosine) between the random spectrum and the hypothetical spectrum of the peptide. To implement this idea, we first embed each peptide into a unit vector in a high-dimensional metric space. The random spectrum is represented by a random vector, and we use random vectors to construct a set of hash functions called locality sensitive hashing (LSH) for preprocessing. We demonstrate that our mapping is accurate. We show that our method can filter out >95.65% of the spectra without missing any correct sequences, or gain 111 times speedup by filtering out 99.64% of spectra while missing at most 0.19% (2 out of 1014) of the correct sequences. In addition, we show that our method can be effectively used for other mass spectra mining applications such as finding clusters of spectra efficiently and accurately.

Contact: tingchen@usc.edu

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 INTRODUCTION

Tandem mass spectrometry is now the most widely used high throughput techniques to analyze proteins and peptides (Yates *et al.*, 1995; Mann and Jensen, 2003; Pandey and Mann, 2003; Aebersold and Mann, 2003). Due to the recent technological advances, the current generation of mass spectrometers can generate thousands to hundreds of thousands of spectra in a single run within an hour. The sheer amount of spectra that can be generated from small amounts of biological samples over a day is staggering and is increasing at an exponential rate with mass spectrometers being often operated in parallel, around the clock. Therefore, faster and accurate data analysis algorithms become increasingly important.

The most reliable and widely used method for mass spectrometry data analysis is the database search. Popular database search programs include SEQUEST (Eng *et al.*, 1994) and MASCOT (Perkins *et al.*, 1999). One generic way to perform database search is to pre-process a protein database by digesting each protein into smaller peptides *in silico* and indexing these peptides by mass. Then, for a query spectrum with the precursor ion mass m, the indexing allows these programs to extract, or filter, a set of peptides S_m that contains all the peptides whose masses are within a certain range Δ of m: $[m - \Delta, m + \Delta]$. All these extracted peptides are then compared against the query spectrum using some scoring functions such as the cross-correlations used in SEQUEST and the probabilistic model used in MASCOT, and the peptides with the best scores are reported and ranked. We call this type of searching methods as the *one-against-all*. The primary bottleneck is the first step of extracting a candidate set to be scored.

Such one-against-all searching methods take *linear time* in terms of the size of S_m, noted by $|S_m|$, which is proportional to the number of peptide sequences having a small precursor mass difference (usually 1–2 Da). For a standard search in MSDB (described in Section 3), a large protein database, $|S_m|$ is roughly between 100 and 200K. In other types of searches such as PTM searches, $|S_m|$ is much larger, and the search is much slower. Standard searches for one run of 100K spectra requires at least $100\,\mathrm{K} \times 100\,\mathrm{K} = 10$ billion comparisons which may take hours to days on a single computer. To meet this kind of challenges, we propose a new method to avoid one-against-all comparisons by reducing the size of S_m through filtering out most of the unrelated peptides.

Our basic idea is to design a set of hash functions to pre-process peptides in the database so that for each

*To whom correspondence should be addressed.

 © The Author 2007. Published by Oxford University Press. All rights reserved. For Permissions, please email: journals.permissions@oxfordjournals.org

Downloaded from https://academic.oup.com/bioinformatics/article/23/5/612/238280 by guest on 20 August 2022

query spectrum we can use these hash functions to search for peptides that are most likely to match the spectrum. Consider a random spectrum and the hash function is the normalized dot product (cosine) between the random spectrum and the hypothetical spectrum of the peptide. Now any query spectrum which is similar to a hypothetical spectrum will have similar scores when compared against the random spectrum. Thus, a pre-filtering strategy would only consider hypothetical spectra that have similar scores.

To implement the above idea, we need to first embed each spectrum as well as the virtual spectra from peptides as unit vectors in a high dimensional metric space, as shown in Figure 1. The random spectrum is represented by a random vector, and we use random vectors to construct a set of hash functions called locality sensitive hashing (LSH), defined later, for preprocessing. The normalized cross-correlation score between two spectra can be approximated by the distance between two vectors or embedded points. Thus database search is equivalent to performing approximate near neighbor (ANN) searches in the embedded high-dimensional space. Using LSH, one can perform ANN searches in *sub-linear* time (Figure 2).

The key reason why we need a measure, such as the Euclidean distance, that obeys the triangle inequality to speedup similarity search is as follows. With triangle inequality, if we embed three spectra S1, S2, S3, in a metric space and we know the distance from S1 to S2 and from S2 to S3, we can bound the distance from S1 to S3. Such guarantees are not possible if the triangle inequality is not valid, such as in the case of cosine similarities (Tabb *et al*., 2003) or cross correlations (Tabb *et al*., 1998). Thus, we can avoid doing pair-wise computations to find similarities.

Our idea of mapping or embedding spectra to a high-dimensional point is both simple, general and novel. We can use our scheme either to search for similar hypothetical spectra generated from sequences, as in database search, or to find similar spectra within experimental databases. This is also the first known application of LSH-based computation of ANN search algorithms in the area of tandem mass spectrometry data analysis. Very recently, Ramakrishnan *et al*. (2006) have constructed database filters using fuzzy and tandem cosine distances and multiple vantage point trees. We believe that our embedding into Euclidean spaces without considering the precursor ion mass of the spectra has potential, and will lead to various applications including comparing spectra and their PTM variants; very preliminary results to support the PTM case are included in the supplementary material. Besides, we have shown that our mapping may be used for other mining tasks such as clustering. Also we use much less dimensions.

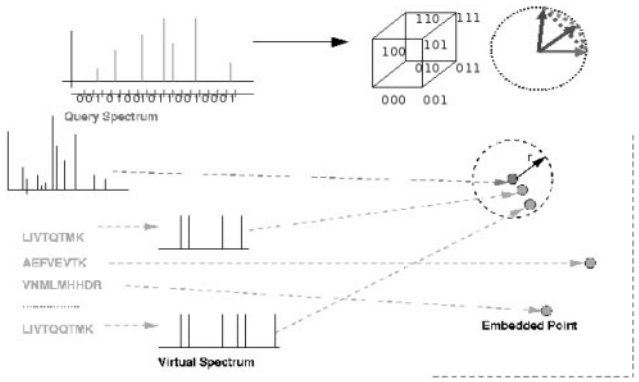

Fig. 1. Overall idea of our method. Spectra are mapped into high-dimensional points and so are hypothetical spectra generated from peptide sequences. To speedup database search, the one-against-all comparisons are avoided by filtering out the most correlated candidates by finding all the near neighbors from the mapped query spectra within a specified radius.

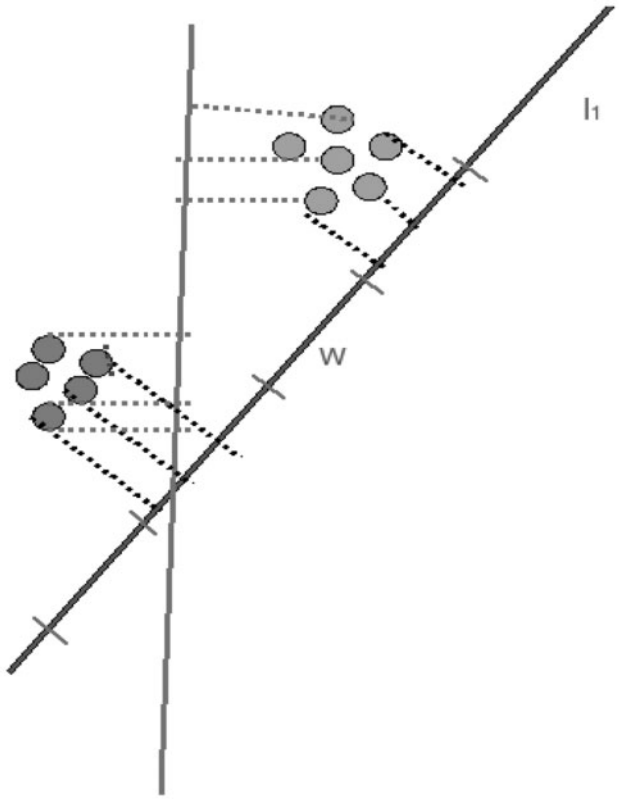

Fig. 2. Basic idea of LSH and near neighbor searches using LSH.

2 METHODS

In this article, we present a general framework that could be used for speeding up database searching of tandem mass spectra as well as other spectral mining tasks. Our main contributions are the following: (i) We accurately map, or *embed* spectra into a high-dimensional Euclidean space, called the metric space. (ii) We demonstrate that the distance between two high-dimensional points corresponding to two mapped spectra is similar to $1 - cc$ where cc is the well-known correlation coefficient between two spectra (Tabb *et al*., 1998, 2003). (iii) We use LSH (Datar *et al*., 2004) to filter out or directly extract near neighbors or peptides whose hypothetical spectra are highly correlated with the query spectrum to avoid the one-against-all comparisons. (iv) We also show how this framework could be used for other mining tasks such as clustering of spectra.

Downloaded from https://academic.oup.com/bioinformatics/article/23/5/612/238280 by guest on 20 August 2022

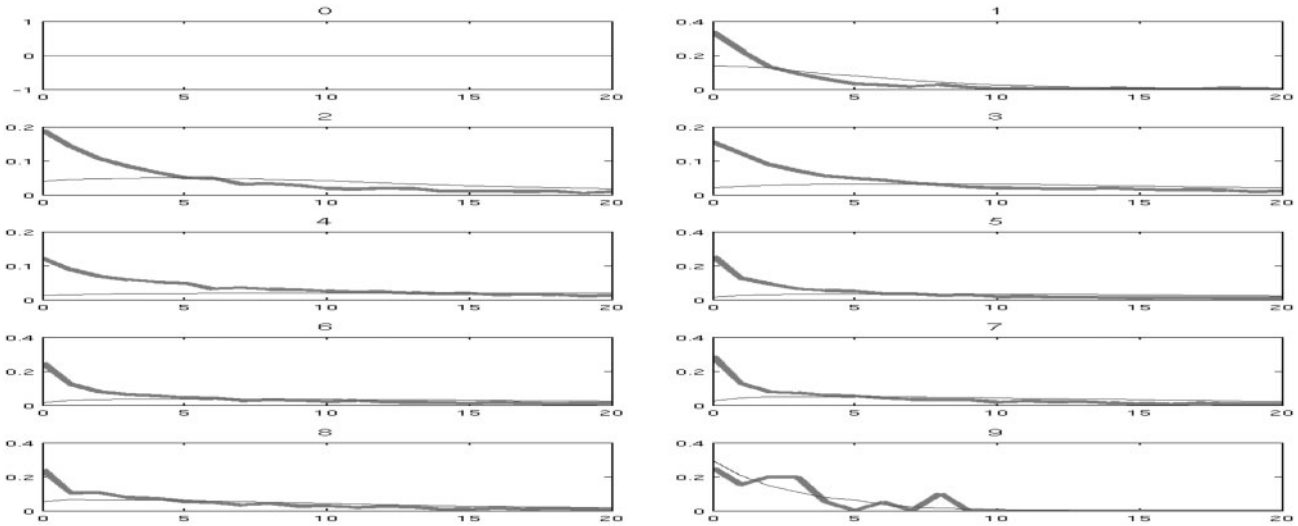

Fig. 3. Signal and noise distributions of peak intensities in different regions of spectra (from the training set). The bolder red lines indicate the distribution for signal while the thinner blue line is for that of the noise. The x-axis is the rank intensity and each of the subgraphs is a pdf for a region.

To perform a database search for a query spectrum, we first embed the spectra into a metric space and then obtain other points corresponding to peptide sequences that lie within a threshold radius from the mapped query point. This is known as a *coarse grain filter* construction. This filtering can be achieved by using ANN search algorithms. Now, fast algorithms with performance guarantees for near neighbor searches are only possible for metric spaces such as Euclidean, Hamming and Manhattan. This is why we map spectra into unit vectors or points in a high-dimensional space.

2.1 Embedding spectra

2.1.1 Preprocessing spectra The Achilles heel of tandem mass spectra analysis is the amount of noise in the mass spectra. In fact, most peaks (around 80%) cannot be explained and are called '*noise*' peaks. '*Signal*' peaks (such as b and y ions) are much more important for similarity measures. As a first step, we remove noise peaks enriching the signal-to-noise ratio.

We first find the intensity distributions of signal and noise peaks using a set of annotated spectra $\{(s, p)\}$ from Keller *et al*. (2002), where s is the spectrum and p is the corresponding peptide. The data set is described in more detail in Section 3. We divide the mass range, for example [1, 2000] dalton in most experiments, into 10 sections. For each section i and for each peak t, we consider its normalized relative intensity rank $rank(t)$: the highest peak has rank 0 and the lowest 1. We divide the peaks into two sets S_i and N_i, where S_i contains only signal peaks and N_i contains only noise peaks. We do this by comparing t with the following ions: b, $b - H_2O$, $b - NH_3$, y, $y - H_2O$, $y - NH_3$ in the hypothetical spectra. We calculate the probability that a peak t in section i is a signal peak by the following formula,

$$\Pr(t = signal) = \frac{\Pr[\text{rank}(t) \mid S_i]}{\Pr[\text{rank}(t) \mid N_i]}.$$

From Figure 3 we can conclude that the SNR is very poor at the ends of the spectra, i.e. at low-mass and high-mass regions. This statistical observation reinforces the mass spectrometry folklore that the *middle region* is the most suitable for finding signal peaks. We perform an approximate method based on the above figure. Instead of scoring each peak, we note that choosing 5 peaks per 100 Da window ensures us the maximum number of signal peaks and the minimum number of noise peaks. Thus, this is our approach.

2.1.2 Mapping and distances In this section, we describe our mapping or embedding of spectra to points in a high-dimensional space. We first *clean* spectra as mentioned in the previous subsection. We assume a common mass range for all spectra. Then we divide the entire mass range of spectra (from 0 to some maximum range) into discrete intervals of 2 Da. For each interval of 2 Da, a bit is set to 1 if the cleaned spectrum contains a peak in that interval, else we set that bit to 0. Thus, for each spectrum, we obtain a bit string where each bit represents the presence of peaks in the corresponding mass range as shown in Figure 1. This method embeds each spectrum to a vertex of an n-dimensional cube. Our feature vectors are then defined to be *unit* vectors in the direction of the corresponding vertices of the n-dimensional cube. This implies that our mapping transforms a spectrum to a point on the surface of an n-dimensional sphere.

We now define the spectral distance or distance between spectra x, y, as the Euclidean distance of the mapped points. Without loss of generality, we represent the mapped points and their corresponding spectra by the same notation. Thus, the distance between two spectra x, y is $||x - y||$ or $D(x, y)$. Now, if the angle between the embedding of the two similar spectra x, y is θ, the spectral similarity is given by $\cos(x, y)$ as in cosine similarity while the cosine distance is given by $1 - \cos(x, y)$.

In mass spectrometry, literature variants of the cosine similarity has been used extensively (Tabb *et al*., 2003). For two very similar spectra, ideally $\cos\theta$ should be close to 1, and the distance $1 - \cos\theta$ should ideally be very small. Since x, y are nearby unit vectors in the mapped space, their Euclidean distance will also be small. Thus, for small angles, $1 - \cos\theta \approx D(x, y)$, where D is the Euclidean distance. However, in reality, this may not be true due to noise in the spectra. But a clear separation of the angles between similar spectra and those between dissimilar ones should exist. Instead of calculating the $1 - \cos\theta$, we calculate $D(x, y)$. This is because for any two vectors x, y, making an angle θ, $||x - y||^2 = 2(1 - \cos\theta)$. Thus, using Euclidean distances upon embedding has the same effect, in terms of similarity calculations, as calculating the cosine distances.

Downloaded from https://academic.oup.com/bioinformatics/article/23/5/612/238280 by guest on 20 August 2022

2.2 Fast near neighbor search

The basic query primitive we use is the following:

Primitive 1: Given a spectrum x and a set of spectra S, we want to find all the spectra S_r that are similar to x, i.e. spectrum $y \in S_r$, iff $D(x, y) < r_q$, where D is the Euclidean distance and the r_q is a query radius.

Note that the set S can be a set of hypothetical spectra from the *in silico* digestion of protein sequence, as in database search. Or it can be a set of experimental spectra too. A very simple approach would be to do a linear scan on the database and output every spectrum y such that $D(x, y) < r_q$. This takes $O(n)$ time. However, if S becomes very large and so do the number of queries, say $O(n)$, then we have a $O(n^2)$ algorithm. Thus, we are willing to trade off accuracy for search speedup. Several sub-linear near neighbor methods exist but we leverage LSH (Datar *et al.*, 2004) since, unlike others, it promises bounded performance guarantees and has been used in different scenarios and has shown to much faster than metric trees based approaches. We briefly present the idea below.

2.2.1 Locality sensitive hashing The basic idea behind random projections is a class of hash functions that are locality sensitive, i.e. if two points (p, q) are close they will have small $|p - q|$ and they will hash to the same value with high probability. If they are far they should collide with small probability.

DEFINITION 1: *A family* $\{H = h : S \rightarrow U\}$ *is called locality-sensitive, if for any point* q*, the function*

$$p(t) = Pr_H[h(q) = h(v) : |q - v| = t]$$

is strictly decreasing in t*. That is, the probability of collision of points* q *and* v *is decreasing with the distance between them.*

Hash function: Consider a random vector a of n dimensions. For any two n-dimensional vectors p, q that we obtain upon embedding two spectra, the distance between their projections $(a \cdot p - a \cdot q)$ is distributed as $|p - q|_s X$ where X is an s-stable distribution. The above can be shown to be locality preserving. For example, in Figure 1, we see two groups of similar points. For each group and a random line l, we project the points onto this random line. Then we *chop* the real line into equal width segments of appropriate size W and assign the hash values to vectors based on which segment they project onto.

Searching: Now, if we are given a query spectrum, to find similar spectra (both hypothetical as well as real), we need to embed this query spectrum into a query point. Then we project the embedded query point onto each of the lines. That is, we use tuples of LSH functions to generate the exact bins where the point hashes. Then, we search for near neighbors only among all the points that fall or collide into the same bin. Thus this avoids searching the entire database. We use several lines to increase the collision probability of similar spectra.

More details of the hash functions and the parameters used are presented in the supplementary document. The key to the successful application of LSH in our case is to use the correct query radius r. We show in the next section how this can be chosen. If we give too high a radius, it might yield a large data set and if the radius is too low, it might not yield any neighbor. If an appropriate query radius is chosen, finding replicates or tight clusters is a simple application of our method.

2.3 Speedup database search

Given a query spectrum x, and a mass spectra database MSDB (described in Section 3), the problem is to find out which peptide $p \in$ MSDB corresponds to x.

Database search is a well-explored topic, see Wan and Chen (2005) for example. Most tools index MSDB by the peptide mass. Then, for a spectrum x, the precursor mass m_x is found. Then all the spectra $S_{m_x} = y \mid y \in \text{MSDB}$ such that $|m_y - m_x| < \Delta$ are compared with x, where Δ is some pre-defined mass tolerance. Each comparison operation between the query spectrum and the candidate spectrum takes a while depending on the scoring function used. We reduce the size of S_{m_x} by filtering the unrelated spectra, speeding up the search. We ensure that we do not filter out the true peptide for a given spectrum while we discard most of the unrelated peptide.

We generate the hypothetical spectrum from each peptide sequence in the database, and then embed those hypothetical spectra in the Euclidean space, as mentioned. Then for filtering, we choose an appropriate threshold radius r and query the LSH algorithm to yield all the candidates within a ball of radius r. The ratio of the total number of peptides within a mass tolerance divided by the number of candidates returned is our speedup.

3 EXPERIMENTAL RESULTS

In this section, we describe the empirical evaluation of our embedding and its application to similarity searching using LSH. Unless otherwise stated, we use the following data set from Keller *et al.* (2002) in which two mixtures, A and B, were obtained by mixing 18 purified proteins together, each having different properties, relative molar amounts and modifications. They performed 22 runs of LC/MSMS on the samples, with 14, 8 runs performed on mixture A, B, respectively. The resultant spectra obtained were analyzed by SEQUEST, and the peptide assignments were then manually scrutinized to determine whether they were correct. The final data set, on curation, contains 125, 1649, 1010 charge +1, +2, +3 spectra, respectively. In this study, we only consider charge +2 spectra. In the supplementary we provide further validations based on two other data sets [HUPO plasma proteome (Adkins *et al.*, 2002) and an ecoli data set from OPD (Marcotte opd (open proteomics database)) annotated by Bern and Goldberg (2005)].

For database search filters, we use a non-redundant protein sequence database called MSDB, which is maintained by the Imperial College, London. The release (20042301) has 1 454 651 protein sequences (around 550M amino acids) from multiple organisms. Peptide sequences were generated by *in silico* digestion by trypsin, with all possible missed cleavages as long as the mass of the resultant peptide was <3500 Da. The final list of peptides with a precursor ion mass till 3500 Da were grouped into different files by their precursor ion mass, a different file for 10 Da. Our database contains 87.4 million peptide sequences.

Note that our techniques are unsupervised except for the selection of query radii. For database search filtering, we started with the 1649 interpreted charge +2 spectra from the above data set. Then we chose all the spectra whose sequences were inside our MSDB database and ended with a K or R. Overall, we used 1014 spectra. We ran those spectra again through MASCOT with a 2 Da tolerance and 1 missed cleavage to verify the peptide annotations only.

Downloaded from https://academic.oup.com/bioinformatics/article/23/5/612/238280 by guest on 20 August 2022

3.1 Empirical evaluation of the embedding

In this section, we carefully analyze our spectral mapping and different distance metrics (between spectra). We cleaned the spectra by picking the most likely to be the signal peaks. Then we constructed the binary bit vector as discussed earlier. In our implementation, we chose the maximum mass for binning to be 1500, 1800 or 2000 Da. This gave us 750, 900 and 1000, dimensions, respectively, for our mapping. However, since we choose on an average of 5 peaks per 100 Da, the space is very sparse. Generating the feature vector took linear time. In particular, for 1649 spectra, it took a total of 6.932 s including file IO.

For the above set of spectra, we know that there are 115 odd clusters with 16 spectra per cluster on an average. Upon embedding the spectra as described above, we calculate the pairwise distances between spectra within the same cluster after embedding the spectra as described above and we term this the similar set, SS. We then choose a representative from each cluster at random and calculate the distances and we call this set the dissimilar set, DS. Then we plot the frequency distribution of DS and SS as they both have similar number of pairwise distances in Figure 4 for three metrics: Hamming, 1-cosine and Euclidean. Its very clear that Hamming is unsuitable as a metric as it has low discriminability. As expected, 1-cosine and Euclidean looks almost similar with low overlaps between the sets DS and SS. More evidence is presented in the supplementary section.

Now, we consider the database of tryptic peptides, MSDB. For each peptide, we generate its hypothetical spectrum and then construct the feature vector as above. For each real spectrum, we calculate the distance with the correct hypothetical spectra and we call this set of scores to be the similar set, SS. Then we choose, from the database, 100 random peptides within 2 Da from precursor ion mass of the given spectrum. We then add the set of scores to the dissimilar set, DS. We then plot the probability distribution of SS and DS in Figure 5. Again, we can see the clear separation between the two sets of distances (with <1% overlap). This indicates that the efficacy of Euclidean distance in our embedded space is a good metric to design filters for database search; note the sharp impulse at 1.414 corresponding to distances between real spectra and completely dissimilar peptides within a mass tolerance of 2 Da.

3.2 Speeding up database search

To test the efficacy of our framework on speeding up database search, we define speedup by the ratio of total number of candidate peptides with a mass tolerance of 2 Da within a database and the total number of peptides whose hypothetical spectra have a distance of at most Δ with the query spectrum and have the same mass tolerance of 2 Da. The parameter Δ allows to trade-off accuracy for speedup.

In Figure 6a we plot the speedup on a logarithmic scale against the miss percentage. This gives us the speedup (or quality of filtering) versus accuracy trade off of using our framework for two different mappings. For a 2 Da range,

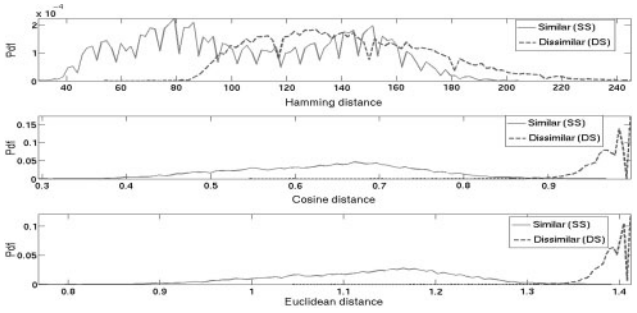

Fig. 4. Distribution of scores with real spectra using different metrics (Hamming, 1-cosine, Euclidean). The dotted curves plot the inter-cluster distances while the solid lines represent the intra-cluster distribution.

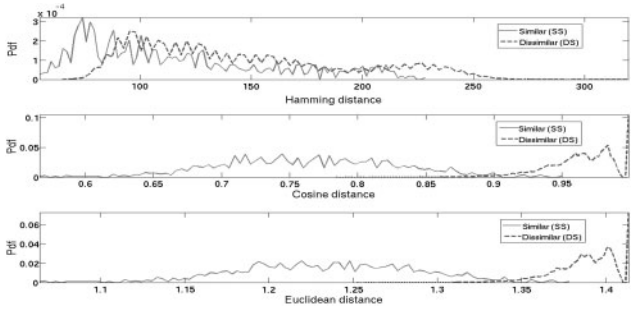

Fig. 5. Distribution of distance between real and hypothetical spectra using different metric. The dotted curves represent the distance between real spectra and distances to hypothetical spectra from 100 different peptides of similar precursor masses. The sequences are from MSDB. The other curve shows the distribution of distances between spectra and the hypothetical spectra from the true peptides.

the number of peptides are around 100–200K. For around a 100K peptide set and 750 dimension mapping, LSH takes 0.21 s on an average to answer queries. As we see from Figure 6a, we can get an average speedup of 111 if we allow 0.29% misses.

When the number of dimensions of our mapping were increased to 1000 (i.e. when we considered 2000Da as the maximum mass), the results were significantly better. For example, for just one spectrum missed, or 0.09% misses, we get a speedup of 111 and for 0.19% misses, the speedup was 281.06. The total number of peptides considered within a 2 Da range was 114 380 on an average. For the above 111 speedup, the frequency distribution of nearest neighbors is shown in Figure 6b. Note that for the same speedup, the error rates are much smaller when 900 or 1000 dimensions were used. In the 900-dimension case, we have no error even at a speedup of 23.66. Using higher dimensions, we will need more space but will get much lower error rates. This is a tradeoff.

The quality of the results obtained may be reasonable for many applications. In fact, we found that some of our errors were

Downloaded from https://academic.oup.com/bioinformatics/article/23/5/612/238280 by guest on 20 August 2022

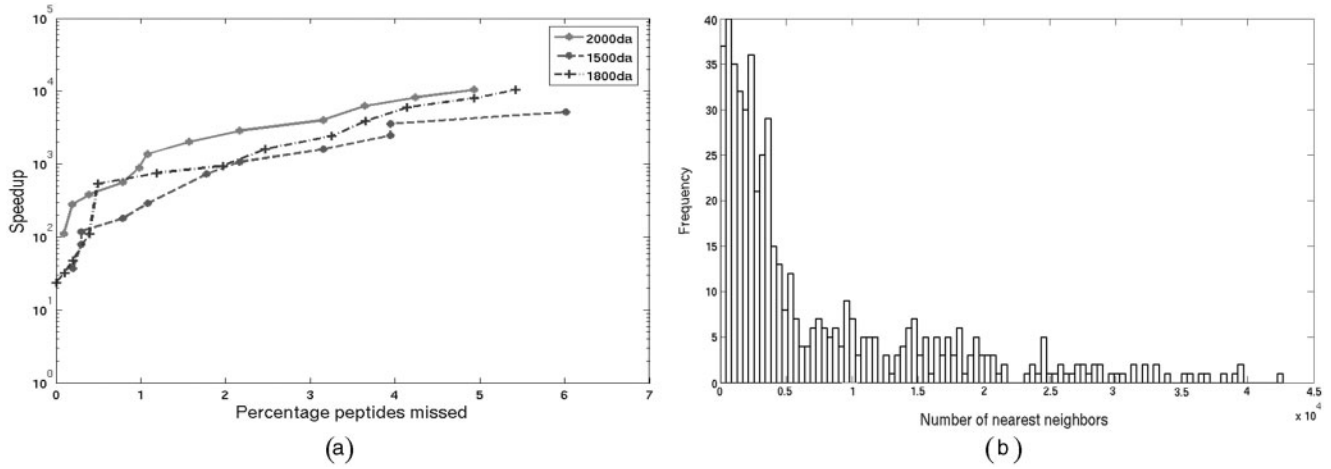

Fig. 6. (**a**) Filtering of spectra for DBASE search. The upper curve represents the speedups when 1000 dimensions were used for LSH while the lower curve represents the 750 dimension case. The middle curve represents the 900 dimension case. (**b**) Filter efficacy: the number of nearest neighbors or spectra filtered by approach in the case when the speedup was 111 and the misses were 0.19% with 1000 dimensions.

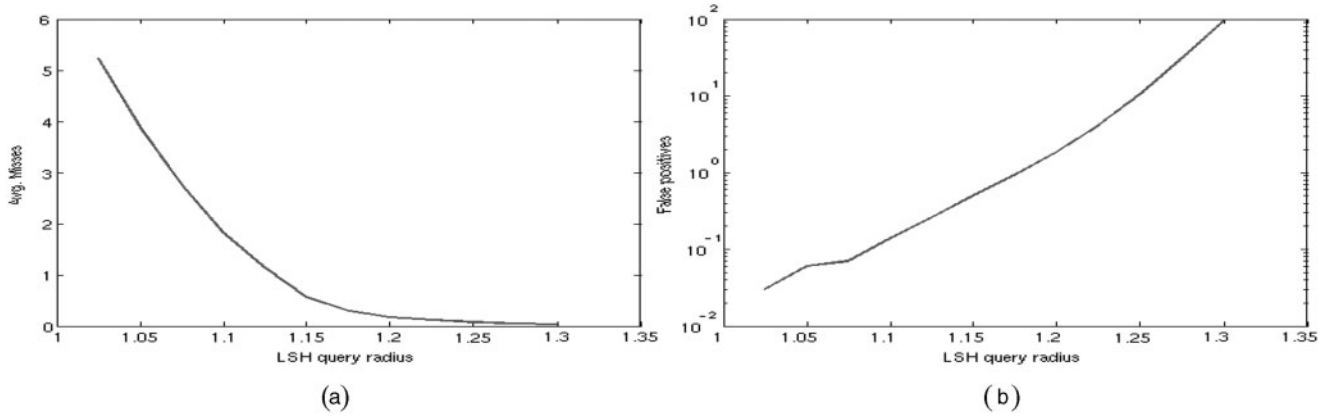

Fig. 7. Performance of similarity searching. (**a**) The average number of spectra that are present in the cluster containing the query spectrum but are missed by LSH. (**b**) The average number of spectra that are not present in the cluster containing the query spectrum but are reported by LSH.

due to low-quality spectra in our test data set. For three spectra, even MASCOT failed to match the annotations of the spectra.

3.3 Similarity searching using LSH

In this section, we additionally quantify the accuracy of our framework for similarity searching and clustering using our embeddings. Note that our filtering idea is essentially similarity searching amidst a database of *in silico*-generated hypothetical spectra. Any metric that is a good database search filter should naturally be suitable as for similarity searching of real spectra.

We first indexed the 1014 spectra using our embedding followed by LSH. For each of the 1014 spectra, we queried LSH with a radius r. We varied r. We plot the number of missed spectra that were actually present in the cluster of the query spectrum in Figure 7a and the number of false positives in Figure 7b. As we increased the radius, the number of misses decreased. This is expected as the radius of the *query ball* increases the number of possible data points that can be considered. As expected, the number of false positives also increased as r increased. We miss an average of one spectrum within each cluster while admitting only one false spectrum.

At $r = 1.0-1.1$ the false positives are not high. In many situations, it might be fine to miss out some bad-quality spectra (distances to bad-quality spectra are usually higher). Also, consider situations where we would like to coarsely partition the data set (e.g. for clustering). Then, we can afford to have a few false positives but we cannot miss any true positives. In such cases we increase the radius to at most 1.25 as the likelihood of an intra-cluster distance being >1.25 is low; from Figure 4.

4 DISCUSSION

The results in the previous section look promising. The clear separation between the dissimilar set (DS) and similar set (SS)

Downloaded from https://academic.oup.com/bioinformatics/article/23/5/612/238280 by guest on 20 August 2022

curves while comparing metrics was interesting. In the supplementary section, we show similar results for two other real world data sets. Also, since we first transform the spectra into binary bit strings, we avoided the huge variations of intensity among different experimental spectra. The signal-to-noise ratio pilot study also underscored the fact that we need to segment the mass range. Note that another reason why we obtained clear separations between the DS and SS with our mapping is that we avoided using precursor ion mass as a feature, since such masses are prone to instrument errors.

For LSH, the performance is quite satisfying. However, there are two implementation issues. Our current indexing takes place in the main memory. This means we need lots of memory to index millions of mass spectra. Even though this is possible with the current 64 bit machines, we need to design disk-based LSH schemes. We are working on a large scale implementation of our framework based on such techniques. Another issue is the choice of the number of bins and the mass coverage. Increasing the number of bins leads us to the curse of dimensionality which would slow down LSH and reduce the filtering speedup.

5 CONCLUSIONS AND FUTURE WORK

In this article, we showed that our method of mapping spectra into high-dimensional points along with the use of fast ANN-searches-based filtering provides a good framework for speeding up database search for mass spectra. In particular, we have demonstrated that Euclidean distances between embedded spectra are highly correlated with the well-known cosine similarity while enabling us to use approximate fast near neighbor search techniques for constructing coarse grain, fast filters. We demonstrated that we can get two to three orders of magnitude speedups for database using these filters. Using this framework, we also showed how we can do similarity searches and find tight clusters or replicates.

This work is the first step in the direction of an integrated framework for large-scale filtering and mining of tandem mass spectra using simple techniques from embeddings, vector spaces and computational geometry. Several directions are being investigated at this point. The main areas of investigation are (i) better embeddings that offer better resolution and faster blind filtering for PTM spectra, (ii) faster external database searching algorithms that use embeddings, (iii) large-scale clustering of mass spectrometry data and (iv) integrating data from different sources using our embeddings. This first involves analyzing spectra across different data sets from the same instrument, followed by the much more challenging problem of studying spectra from different instruments.

Conflict of Interest: none declared.

REFERENCES

Adkins,J.N. *et al*. (2002) Toward a human blood serum proteome i: analysis by multidimensional separation coupled with mass spectrometry. *Mol. Cell. Proteomics*, **1**, 947–955.

Aebersold,R. and Mann,M. (2003) Mass spectrometry-based proteomics. *Nature*, **422**, 198–207.

Bern,M.W. and Goldberg,D. (2005) Eigenms: de novo analysis of peptide tandem mass spectra by spectral graph partitioning. In *RECOMB '05: Proceedings of the Ninth Annual International Conference on Computational Molecular Biology*.

Datar,M. *et al*. (2004) Locality-sensitive hashing scheme based on *p*-stable distributions. In *SCG '04: Proceedings of the Twentieth Annual Symposium on Computational Geometry*, ACM Press, New York, NY, USA, pp. 253–262.

Eng,J. *et al*. (1994) An approach to correlate tandem mass spectral data of peptides with amino acid sequences in a protein database. *J. Am. Soc. Mass. Spec.,* **5**, 976–989.

Keller,A. *et al*. (2002) Experimental protein mixture for validating tandem mass spectral analysis. *OMICS*, **6**, 207–212.

Mann,M. and Jensen,O. (2003) Proteomic analysis of post-translational modifications. *Nat. Biotechnol*., **21**, 255–261.

Marcotte,E.M. Opd (open proteomics database). http://apropos.icmb.utexas.edu/opd/.

Pandey,A. and Mann,M. (2003) Proteomics to study genes and genomes. *Nature*, **405**, 837–846.

Perkins,D.N. *et al*. (1999) Probability-based protein identification by searching sequence databases using mass spectrometry data. *Electrophoresis*, **20**, 3551–3567.

Ramakrishnan,S. *et al*. (2006) A fast coarse filtering method for peptide identification by mass spectrometry. *Bioinformatics* (in press).

Tabb,D.L. *et al*. (2003) Similarity among tandem mass spectra from proteomic experiments: detection, significance, and utility. *Anal. Chem*., **75**, 2470–2477.

Tabb,D.L. *et al*. (1998) Method to compare collision-induced dissociation spectra of peptides: potential for library searching and subtractive analysis. *Anal. Chem*., **70**, 3557–3565.

Wan,Y. and Chen,T. A hidden markov model based scoring function for tandem mass spectrometry. In *RECOMB 2005*.

Yates,J.R.3rd, *et al*. (1995) Method to correlate tandem mass spectra of modified peptides to amino acid sequences in the protein database. *Anal. Chem*., **67**, 1426–1436.

Downloaded from https://academic.oup.com/bioinformatics/article/23/5/612/238280 by guest on 20 August 2022