EFFICIENT MANIFOLD LEARNING FOR SPEECH RECOGNITION USING LOCALITY SENSITIVE HASHING

Vikrant Singh Tomar, Richard C. Rose

Department of Electrical and Computer Engineering, McGill University, Montreal, QC, Canada

vikrant.tomar@mail.mcgill.ca, rose@ece.mcgill.ca

ABSTRACT

This paper considers the application of a random projections based hashing scheme, known as locality sensitive hashing (LSH), for fast computation of neighborhood graphs in manifold learning based feature space transformations in automatic speech recognition (ASR). Discriminative manifold learning based feature transformations have already been found to provide significant improvements in ASR performance. The motivation of this work is the fact that the high computational complexity of these techniques has prevented their application to very large speech corpora. The performance of this integrated system is evaluated both in terms of computational complexity and ASR word recognition accuracy. Further comparisons of ASR performance with the well-known linear discriminant analysis are provided. These results demonstrate that LSH provides the much needed speed boost to manifold learning techniques with minimal impact on their ASR performance, thus enabling the application of these algorithms to large speech databases.

Index Terms: Locality sensitive hashing, locality preserving discriminant analysis, manifold learning, dimensionality reduction, speech recognition

1. INTRODUCTION

Manifold learning algorithms have found extensive usage in feature space transformation and dimensionality reduction techniques for speech and image analysis [1, 2]. It has been suggested that the acoustic feature space is confined to lie on one or more low dimensional manifolds [3, 4]. Therefore, a feature space transformation technique that explicitly models and preserves the local relationships of data along the underlying manifold should be more effective for speech processing. Accordingly, multiple studies have demonstrated gains in automatic speech recognition (ASR) performance when using features derived from a manifold learning based approach. Tang et. al. reported gains in ASR performance using features derived from locality preserving projections (LPP) [5]. In previous work [6, 7], the authors presented discriminative manifold learning techniques that led to significant improvements in ASR word error rates (WER) for a speech in noise task as compared to well-known techniques such as linear discriminant analysis (LDA) [8, 9] and LPP.

Despite the advantages, a major criticism against the application of manifold learning techniques to speech processing has been the sheer computational complexity of these methods [2, 5, 10, 11]. This complexity originates from the need to calculate a pair-wise similarity measure between feature vectors to construct nearest neighborhood graphs, which are essential to all manifold learning techniques. If a given training set consists of N feature vectors of dimensionality d, it would take computational complexity $O(dN^2)$ in order to construct the similarity graphs. For large amounts of speech data, where each corpus can have up to hundreds of millions of feature vectors each having hundreds of dimensions, $O(dN^2)$ is a formidable computational requirements for an algorithm. A number of algorithms exist for faster but approximate nearest neighbors search such as kd-trees; however, many of these algorithms reach the complexity of linear search as the dimensionality of feature vectors increases [12].

This work investigates a randomized algorithm, known as locality sensitive hashing (LSH) [13–15], for fast construction of the neighborhood graphs as applied to manifold based feature transformations in ASR. LSH is particularly well suited for finding nearest neighbors in high-dimensional speech feature spaces. LSH creates hashed signatures of vectors in order to distribute them into a number of discrete buckets such that vectors close to each other are more likely to fall into the same bucket. For a given query point, the nearest neighbors search is restricted to data points belonging to the bucket that the query point is hashed to. LSH can drastically reduce the computational time, at the cost of a small probability of failing to find the absolute closest match. In this work, use of LSH provided a factor of 10 speed-up without sacrificing much ASR performance for manifold based algorithms. These reductions in computational complexity should enable application of manifold based approaches to large speech datasets.

The LSH algorithm is evaluated in the context of the locality preserving discriminant analysis (LPDA) technique [6]. LPDA attempts to preserve the underlying local sub-manifold based relationships of feature vectors while at the same time tries to maximize a criterion related to the separability between classes of feature vectors. The LSH scheme is incorporated within the LPDA approach for fast computation of neighborhood graphs with the expectation of achieving high computational efficiency with minimum impact on ASR performance. LPDA is chosen as an example of manifold learning techniques primarily because of its good ASR performance, as reported in previous work [6].

The rest of this paper is structured as follows. A review of LPDA is presented in Section 2, followed by a general introduction to LSH and LPDA specific implementation details in Section 3. Section 4 provides the experimental study and discussions on the evaluation of the performance of LPDA with LSH in terms of ASR WER and training time. This section also discusses the trade-offs between LSH efficiency and the choice of parameters, and their impact on ASR performance. Finally, Section 5 concludes the paper.

2. LOCALITY PRESERVING DISCRIMINANT ANALYSIS

This section summarizes the LPDA algorithm. A more details discussion can be found in [6]. LPDA is a discriminative manifold

This work is supported by Google Inc., Natural Sciences and Engineering Research Council of Canada, and McGill University.

learning technique that attempts to maximize class separability while preserving local sub-manifold based relationships of the data vectors. The goal of LPDA is to estimate the parameters of a projection matrix $\boldsymbol{P} \in \mathbb{R}^{d \times m}$, with $m \leq d$, in order to perform a constrained transformation of the features from a d-dimensional space onto an m-dimensional space.

Following the graph-embedding framework [2], LPDA characterizes the underlying manifold by embedding the training feature vectors, $\boldsymbol{X} = \{\boldsymbol{x}_1, \cdots \boldsymbol{x}_N\} \in \mathbb{R}^d$, into two undirected weighted graphs, namely the intrinsic graph $\mathcal{G}_{int} = \{\boldsymbol{X}, \boldsymbol{W}_{int}\}$ and the penalty graph $\mathcal{G}_{pen} = \{\boldsymbol{X}, \boldsymbol{W}_{pen}\}$. The nodes of the graphs, $\boldsymbol{X}$, represent the feature vectors. Therefore, $\boldsymbol{X}$ is same for both the intrinsic and penalty graphs. $\boldsymbol{W}_{int}$ and $\boldsymbol{W}_{pen} \in \mathbb{R}^{N \times N}$ are the intrinsic and penalty affinity matrices that represent the weights on the edges connecting the graph nodes. The affinity matrices characterize the statistical and geometrical similarities of the feature vectors. The elements of the affinity matrices are defined in terms of a Gaussian kernel as,

$$w_{ij}^{int} = \begin{cases} exp\left(\frac{-||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}{\rho}\right) & ; C(\boldsymbol{x}_i) = C(\boldsymbol{x}_j), e(\boldsymbol{x}_i, \boldsymbol{x}_j) = 1 \\ 0 & ; \text{Otherwise} \end{cases} \tag{1}$$

and

$$w_{ij}^{pen} = \begin{cases} exp\left(\frac{-||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}{\rho}\right) & ; C(\boldsymbol{x}_i) \neq C(\boldsymbol{x}_j), e(\boldsymbol{x}_i, \boldsymbol{x}_j) = 1 \\ 0 & ; \text{Otherwise} \end{cases} \tag{2}$$

where ρ is the kernel scale parameter. $C(\boldsymbol{x}_i)$ refers to the class or label of vector $\boldsymbol{x}_i$. The function $e(\boldsymbol{x}_i, \boldsymbol{x}_j)$ indicates whether $\boldsymbol{x}_i$ lies in the near neighborhood of $\boldsymbol{x}_j$. Closeness to a vector $\boldsymbol{x}_i$ can be measured either by K-nearest neighbors or neighbors within radius R. In this work, a node $\boldsymbol{x}_i$ is connected to the 200 nearest neighbors belonging to the same class $C(\boldsymbol{x}_i)$ in the intrinsic graph, $\mathcal{G}_{int}$. Similarly, in the penalty graph, $\mathcal{G}_{pen}$, a node $\boldsymbol{x}_i$ is connected to the 200 closest neighbors *not* belonging to the class $C(\boldsymbol{x}_i)$.

For a given graph $\mathcal{G}$, a scatter measure is defined in terms of the target space vectors $\boldsymbol{y}_i$, where $\boldsymbol{y}_i$ is obtained according to the projection $\boldsymbol{y}_i = \boldsymbol{P}^T \boldsymbol{x}_i$,

$$F(\boldsymbol{P}) = \sum_{i \neq j} ||\boldsymbol{y}_i - \boldsymbol{y}_j||^2 w_{ij} \tag{3a}$$

$$= 2\boldsymbol{P}^T \boldsymbol{X}(\boldsymbol{D} - \boldsymbol{W})\boldsymbol{X}^T \boldsymbol{P} \tag{3b}$$

where $\boldsymbol{D}$ is a diagonal matrix whose elements correspond to the column sum of the affinity matrix $\boldsymbol{W}$, *i.e.,* $\boldsymbol{D}_{ii} = \sum_j w_{ij}$. An optimal projection matrix $\boldsymbol{P}$ can be obtained by minimizing or maximizing the scatter in Eq. (3b), depending on whether the goal is to preserve or discard the concerned graph structure.

In LPDA, the properties corresponding to inter-class compactness are penalized, *i.e.*, the scatter of the penalty graph is maximized, while the properties inherent to within-class compactness are preserved, meaning, the scatter of the intrinsic graph is minimized. To this end, the ratio of the penalty graph scatter measure to that of the intrinsic graph is treated as a measure of class separability and graph-preservation. An optimal projection matrix is obtained by maximizing this measure,

$$\underset{\boldsymbol{P}}{\arg\max}\, \text{tr}\left((\boldsymbol{X}(\boldsymbol{D}_i - \boldsymbol{W}_i)\boldsymbol{X}^T \boldsymbol{P})^{-1}(\boldsymbol{P}^T \boldsymbol{X}(\boldsymbol{D}_p - \boldsymbol{W}_p)\boldsymbol{X}^T \boldsymbol{P})\right) \tag{4}$$

where the subscripts i and p signify ‘intrinsic’ and ‘penalty’ graphs, respectively [2,6]. Eq. (4) can be solved as a generalized eigenvalue problem,

$$(\boldsymbol{X}(\boldsymbol{D}_p - \boldsymbol{W}_p)\boldsymbol{X}^T)\boldsymbol{p}_{lpda}^j = \lambda_j(\boldsymbol{X}(\boldsymbol{D}_i - \boldsymbol{W}_i)\boldsymbol{X}^T)\boldsymbol{p}_{lpda}^j \tag{5}$$

where $\boldsymbol{p}_{lpda}^j$ is the j^{th} column of the linear transformation matrix $\boldsymbol{P}_{lpda} \in \mathbb{R}^{d \times m}$, and is the eigenvector associated with the j^{th} largest eigenvalue.

One of the biggest issues in applying manifold based techniques to larger datasets is the very high computational complexity required for populating the affinity matrices, for instance, $\boldsymbol{W}_{int}$ and $\boldsymbol{W}_{pen}$ in LPDA. The problem originates from the need to find the nearest neighbors to all the vectors in the training set. Computational complexity for this search grows in proportion to the square of the number of training vectors. Therefore, it is important to find a solution for faster computation of these neighborhoods. One such algorithm, namely LSH, is discussed in the next section.

3. LOCALITY SENSITIVE HASHING

LSH is a class of randomized algorithms that promise fast nearest neighborhood search with a high degree of accuracy. A number of different implementations exist for these schemes [16]. This work utilizes a particular algorithm that tries to find the nearest neighbors in the Euclidean space using random projections derived from a p-stable distribution [14]. Techniques based on LSH have previously been applied to speech [17]. However, the work in [17] utilized an implementation of LSH that finds the approximate nearest neighbors in the cosine correlation space [18]. A general description of random projections based LSH algorithm is presented in Section 3.1, followed by the details related to its implementation and incorporation with LPDA in Section 3.2. Further discussion on LSH can be found in [13–15].

3.1. Random Projections based LSH

In this LSH scheme, each vector $\boldsymbol{x}_i$ is hashed to an integer value (bucket) by a family of hash functions $\mathcal{H} = \{h : \mathbb{R}^d \to \mathbb{N}\}$. This is achieved by performing inner-product of $\boldsymbol{x}_i$ with a random vector, $\bar{\boldsymbol{a}}$, and assigning a hash value based on which bucket (segment on real line) it projects into. The hash function used in this work is given by,

$$h(\boldsymbol{x}) = \left\lfloor \frac{< \bar{\boldsymbol{a}}, \boldsymbol{x} > + \bar{b}}{w} \right\rfloor, \tag{6}$$

where $\bar{\boldsymbol{a}}$ is a d-dimensional random vector whose entries are chosen from a p-stable distribution, w is the width of each segment or bucket on the real-line and acts as a quantization factor, and the bias, $\bar{b}$, is a uniform random number taken from $[0, w]$. The projection of all the vectors in this manner results in a chain or table of hash buckets, each having pointers to one or more vectors. Note that only non-empty buckets are retained.

It can be inferred from p-stable distributions that for two arbitrary vectors, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, the distance between their projections, $< \bar{\boldsymbol{a}}, \boldsymbol{x_i} >$ and $< \bar{\boldsymbol{a}}, \boldsymbol{x_j} >$, is distributed as $||\boldsymbol{x}_i - \boldsymbol{x}_j||_p \bar{\boldsymbol{Z}}$ where $\bar{\boldsymbol{Z}}$ is a random variable that follows p-stable distribution [14, 19]. This property guarantees that the aforementioned hash family is locality sensitive, indicating that, if two points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are close together then they should have high probability of collision or hashing to the same bucket $Pr[h(\boldsymbol{x}_i) = h(\boldsymbol{x}_j)] \geq P_1$ and if the two points are far apart then the collision probability should be small $Pr[h(\boldsymbol{x}_i) = h(\boldsymbol{x}_j)] \leq P_2$, where $P_1/P_2 > 1$.

For optimal performance, the difference between P_1 and P_2 should be large. To this end, a number of random projections are used to create a family of composite hash functions $\mathbb{G} = \{g : \mathbb{R}^d \to \mathbb{N}^k\}$ such that $g(\boldsymbol{x}) = [h_1(\boldsymbol{x}), \ldots, h_k(\boldsymbol{x})]$, where $h_i(\boldsymbol{x}) \in \mathcal{H}$. Increasing the dimensionality k of the hash functions improves the hashing discriminative power as $(P_1/P_2)^k > P_1/P_2$. Effectively, a large k might result in a higher number of buckets each having fewer points and in turn, a smaller probability that the query and the nearest neighbors fall in the same bucket in all k projections. To reduce the impact of such unlucky hashing, L independent hash tables are created for which hash functions, $g_1, \ldots, g_L$, are uniformly chosen from $\mathbb{G}$. This is motivated by the fact that a true nearest neighbor will be unlikely to be unlucky in all the projections. By increasing L one can find the true nearest neighbors with arbitrarily high probability.

After hashing, each data point is represented by a k-dimensional hash signature. However, comparing these k-dimensional signatures to detect collisions may still be computationally expensive. To this end, a second level conventional hashing is implemented to store the k-dimensional signatures. The table size in the bucket hashing is chosen to be large enough to ensure, with a high probability, that different signatures lead to different buckets. Such a secondary hashing further reduces the number of comparisons during collision detection and bucket lookup from $O(k)$ to $O(1)$.

For a given query $\boldsymbol{q}$, the search proceeds as follows. First, the query $\boldsymbol{q}$ is hashed to one of the buckets in each of the L tables. Then, candidate nearest neighbors to $\boldsymbol{q}$ are gathered by performing the union of these buckets from all the tables. Finally, the required nearest neighbors are searched – either selecting the K closest points for K-nearest neighbors or selecting the points $\boldsymbol{x}_i$ such that $||\boldsymbol{x}_i - \boldsymbol{q}|| \leq R$ for R-nearest neighbors – from these candidates.

3.2. Implementation Details

The conventional LSH algorithm is targeted at applications with a small query set that is separate from the training set on which LSH tables are generated. Each query point is hashed to one of the existing buckets in each of the tables, and then the approximate nearest neighbors are identified from the union of all such buckets. However, it is often necessary to search for nearest neighbors for all the points given in the training set. For example, for the LPDA algorithm described in Section 2, the nearest neighbors were calculated for all 1.4 million feature vectors from the training set in order to populate the affinity matrices $\boldsymbol{W}_{int}$ and $\boldsymbol{W}_{pen}$. This is true for all manifold learning algorithms. In such cases, it will not be feasible by both time and resources to re-iterate over the entire training set in order to find the nearest neighbors for each query. To avoid this issue, this work has implemented a modified version of the conventional LSH algorithm. For each hash bucket in the LSH data structure, pairwise distances are calculated between all hashed vectors in order to create candidate neighborhood structures for all the points in that bucket. Then, with reference to given class labels, these candidate neighborhood structures are concatenated to create two separate graphs for within-class and inter-class distances. Final nearest neighborhood structures are selected from these candidates.

4. EXPERIMENTAL STUDY AND DISCUSSION

This section describes the experiments performed to evaluate the effectiveness of LSH for building neighborhood graphs when incorporated with discriminative manifold learning based LPDA as applied to ASR feature space transformations. The effectiveness is measured in terms of reduction in time required to train the LPDA projection matrix, $\boldsymbol{P}$, and ASR word error rate (WER) obtained using the transformed features. The results also present ASR performance comparison of LPDA with LSH (LPDA-LSH) to that of linear discriminant analysis (LDA). Furthermore, the reduction in computational complexity and issues related to the choice of LSH parameters are also discussed in this section.

4.1. Task Domain and Setup

The experiments in this work are conducted on the European Telecommunications Standards Institute's Aurora-2 speech in noise corpus. Aurora 2 training set contains a total of 8440 noisy utterances collected from 55 male and 55 female speakers. The corpus was created by adding noise to connected digit utterances spoken in a quiet environment. As a result, the corpus represents a simulation of a speech in noise task, and one must be careful when generalizing these results to the wide range of actual speech in noise tasks.

The ASR system is configured using whole word continuous density hidden Markov models (CDHMMs) with 16 states per word-model, plus 3 states for the silence model, and 1 state for the short pause model. There were 11 CDHMM models and a total of 180 states. Each state is modeled by a mixture of 3 Gaussians. Semitied covariance (STC) transformations are applied prior to recognition to account for the correlation introduced to the transformed features by the LPDA and LDA projections, as described in [6, 20].

Mel-frequency cepstrum coefficients (MFCCs) features – consisting of 12 static coefficients, normalized $\log$ energy, Δ-cepstrum and $\Delta\Delta$-acceleration – are used for baseline comparison. The transformations, LPDA and LDA, are estimated from 117 dimensional super-vectors obtained by concatenating 9 vectors of MFCC augmented with $\log$ energy. The classes are defined as states of the CDHMMs. A neighborhood size of $K_{int} = K_{pen} = 200$ is used for the calculations of affinity matrices, $\boldsymbol{W}_{int}$ and $\boldsymbol{W}_{pen}$ in LPDA. Systems labeled as LPDA-LSH use LSH for building the nearest neighborhood graphs. The resultant projection matrix $\boldsymbol{P}$ is then used to project the 117-dimensional training and test vectors to a 39 dimensional space.

4.2. Results

The ASR WER obtained for the aforementioned speech in noise task are displayed in Table 1. All feature space transformations and HMMs are trained using training utterances corrupted by a mixture of noise conditions. The test results are averaged over a mix of utterances corrupted by three noise types, namely Sub.=subway, Exh.=exhibition hall and car. The table contains ASR WER for four different feature types for clean testing and four different noise levels ranging from 20dB to 5dB SNR. The first row in the table displays the baseline ASR WER obtained when no feature space transformation is performed. The second row, labeled "LDA", corresponds to application of the 117 by 39 dimensional projection matrix obtained by LDA to the concatenated super vectors. The third row presents ASR WER results for features obtained by LPDA transformation. Finally, the last row, labeled "LPDA-LSH" refers to ASR results when LPDA transformation is obtained while using the fast LSH scheme for nearest neighbors calculations. Note that for all but baseline features STC transforms are estimated to minimize the impact of the data distributions resulting from the feature space transformations.

When comparing ASR performance of LPDA-LSH with that of LPDA in Table 1, it can be seen that LPDA-LSH shows almost no impact on ASR performance as compared to LPDA in high SNR cases. However, ASR performance of these randomized algorithms seems to be affected by the presence of noise. By comparing the ASR performance of LPDA-LSH with LDA it can be seen that LPDA-LSH produces improved ASR performance over LDA in most noise conditions. Many additional approaches for discrim-

Table 1. WER for mixed noise training and noisy testing on Aurora-2 speech corpus for Baseline, LDA, LPDA and LPDA-LSH.

Features	SNR (dB)				
	Clean	20	15	10	5
Baseline	1.88	3.03	3.73	6.10	12.31
LDA	1.98	2.57	3.35	5.98	14.18
LPDA	1.44	2.23	3.23	5.71	12.77
LPDA-LSH	1.45	2.20	3.28	5.67	14.28

inative feature space transformations have been proposed including heteroscedastic linear discriminant analysis (HLDA) [21], which allows for unequal class specific covariances. No direct comparisons between LPDA and HLDA are presented here, primarily because of performance similarities which have been observed between HLDA and LDA when STC transformations are applied in HMM [22].

The results in Table 1 are obtained using a training set which contained a total of 1.4 million vectors of dimensionality 117 extracted from 8440 utterances. The execution times reported for all techniques are obtained using the same multi-core computing system. The time taken in training the projection matrices for LDA, LPDA, and LPDA-LSH are 90 seconds, 26 hours, and 2.5 hours, respectively. LSH helps speed up the LPDA transformation by reducing the total time from 26 to 2.5 hours, thus providing a factor of 10 speed up. This remarkable increase in computational boost should enable the application of manifold learning based feature space transformation techniques to generally large speech databases.

4.3. Computational Analysis

Application of LSH reduces the computational complexity for calculating nearest neighbors from $O(dN^2)$ to $O(dkNL) + O(dN_B^2)$, where d = dimensionality of feature vectors, N = total number of feature vectors, N_B = average number of points in each bucket, k= dimension of hash functions $g(\boldsymbol{x})$, and L= number of hash tables. Note that this is a significant improvement as N_B can be several orders of magnitude smaller than N. In this work, compared to $N = 1.4$ million, N_B had a value in the range of 60 for $w = 5, L = 6$ and $k = 3$.

4.4. LSH Parameterization vs Performance

There are three main parameters that affect the performance of LSH, the quantization factor, w, the number of projections or dimensions of the hash function, k, and number of tables, L. The parameter w controls the width of the buckets and hence the probability of collision for any two points. A large w results in large buckets, thus an increase in the false collisions and computational complexity. It has been observed in other domains that a small positive value of w suffices to achieve optimal LSH performance and larger values do not have a huge impact on accuracy [15]. In this work, $w = 5$ is found to provide good LSH performance. Increasing the dimensionality of the hash functions, k, improves the hashing discriminative power, hence effectively decreasing the probability of collision of two points. It represents a trade-off between the time spent in computing hash values and time spent in pruning candidate neighbors to find the nearest neighbors from a bucket. In this work, values of k are searched in the range of 1 to 10. Some of these results are presented in Figure 1 that shows a trade-off between the time required for training the LPDA projection matrix and ASR performance. Best performance is observed for $k = 3$ with significant reduction in training time. Using value of $k < 3$ did not provide worthwhile gains in computational complexity. Increasing L should increase the probability of finding accurate nearest neighbors, however, computational complexity also increases. This is because more tables mean more projections to perform and more buckets to scan. In this work, a suitable value of L is searched in the range of 1 to 6. Figure 2 presents a graph of average ASR WER versus the number of tables for the dataset described in Section 4.1 with reference to the WER from LPDA without LSH. For these experiments, the training time of LPDA-LSH increases from 0.6 to 2.5 hours with the increase in L. Since $L = 6$ provides near-optimal ASR performance, a higher value for the number of tables are not investigated.

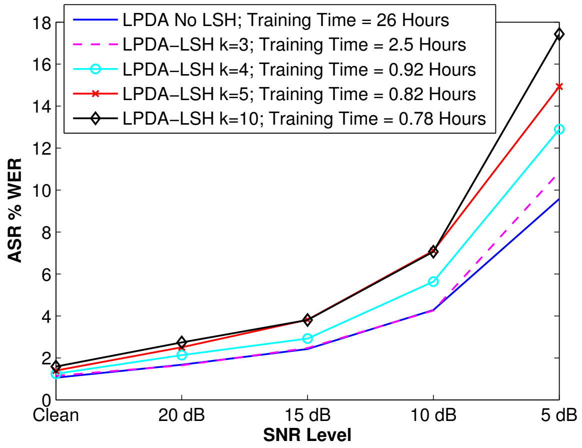

Fig. 1. Impact of varying the dimension, k, of hash functions on ASR performance and LPDA training time. For these experiments $w = 5$ and $L = 6$ were fixed.

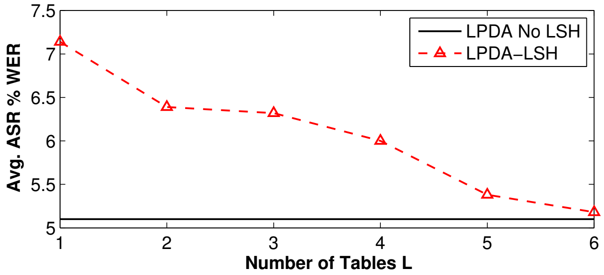

Fig. 2. Impact of varying the number of tables, L, of LSH on ASR performance. For these experiments $w = 5$ and $k = 3$ were fixed.

5. CONCLUSION

This paper has investigated the application of a fast approximate nearest neighbor search algorithm, known as locality sensitive hashing (LSH), in conjunction to a recently proposed discriminative manifold learning technique, locality preserving discriminant analysis (LPDA). ASR WER and execution times were reported for LPDA with and without LSH. Performance comparisons were also made between these approaches and the more widely used LDA. It was demonstrated that LSH provides the much needed speed boost to manifold learning techniques with minimal impact on their ASR WER performance. These results should enable the application of manifold learning algorithms to large speech databases.

6. REFERENCES

[1] Mikhail Belkin and Partha Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural computation*, 2003.

[2] Shuicheng Yan, Dong Xu, Benyu Zhang, Hong-Jiang Zhang, Qiang Yang, and Stephen Lin, "Graph embedding and extensions: a general framework for dimensionality reduction.," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 1, pp. 40–51, Jan. 2007.

[3] K. N. Stevens, *Acoustic Phoenetics*, MIT Press, Cambridge, MA, USA, 1998.

[4] Aren Jansen and Partha Niyogi, "Intrinsic Fourier analysis on the manifold of speech sounds," in *ICASSP: IEEE International Conference on Acoustics Speech and Signal Processing*, 2006.

[5] Yun Tang and Richard Rose, "A study of using locality preserving projections for feature extraction in speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Las Vegas, NV, Mar. 2008, pp. 1569–1572, IEEE.

[6] Vikrant Singh Tomar and Richard C. Rose, "Application of a locality preserving discriminant analysis approach to ASR," in *2012 11th International Conference on Information Science, Signal Processing and their Applications (ISSPA)*, Montreal, QC, Canada, July 2012, pp. 103–107, IEEE.

[7] Vikrant Singh Tomar and Richard C Rose, "A correlational discriminant approach to feature extraction for robust speech recognition," in *Interspeech*, Portland, OR, USA, 2012.

[8] K. Beulen, L. Welling, and H. Ney, "Experiments with linear feature extraction in speech recognition," in *European Conference on Speech Communication and Technology*, 1995.

[9] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, Wiley Interscience, 2nd edition, 2000.

[10] Xiaofei He and Partha Niyogi, "Locality preserving projections," in *Neural Information Processing Systems (NIPS)*, 2002.

[11] Yong Ma, Shihong Lao, Erina Takikawa, and Masato Kawade, "Discriminant analysis in correlation similarity measure space," *Proceedings of the 24th international conference on Machine learning - ICML '07*, , no. 1, pp. 577–584, 2007.

[12] Christopher M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 1st edition, 2006.

[13] Piotr Indyk and R Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality," in *Proceedings of the thirtieth Annual ACM Symposium on Theory of Computing*, 1998, pp. 604–613.

[14] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," *Proceedings of the twentieth annual symposium on Computational geometry - SCG '04*, p. 253, 2004.

[15] Alexandr Andoni, Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab Mirrokni, *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice*, MIT Press, 2006.

[16] Loïc Paulevé, Hervé Jégou, and Laurent Amsaleg, "Locality sensitive hashing: A comparison of hash function types and querying mechanisms," *Pattern Recognition Letters*, vol. 31, no. 11, pp. 1348–1358, Aug. 2010.

[17] Aren Jansen and Benjamin Van Durme, "Efficient spoken term discovery using randomized algorithms," in *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*, Dec. 2011, pp. 401–406.

[18] Moses S. Charikar, "Similarity estimation techniques from rounding algorithms," *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing - STOC '02*, p. 380, 2002.

[19] V. M. Zolotarev, "One-Dimensional Stable Distributions," in *Vol. 65 of Translations of Mathematical Monographs*. American Mathematical Society, 1986.

[20] M. J. F. Gales, "Adapting Semitied Full Covariance Matrix HMMs," Tech. Rep., Cambridge University, UK, 1997.

[21] Nagendra Kumar, *Investigation of silicon-auditory models and generalization of linear discriminant analysis for improved speech recognition*, Ph.D. thesis, Johns Hopkins University, Baltimore, MD, 1997.

[22] M. J. F. Gales, "Maximum likelihood multiple subspace projections for hidden Markov models," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 2, pp. 37–47, 2002.