Stochastic Neighbor Embedding

Geoffrey Hinton and Sam Roweis
Department of Computer Science, University of Toronto
10 King's College Road, Toronto, M5S 3G5 Canada
{*hinton,roweis*}@*cs.toronto.edu*

Abstract

We describe a probabilistic approach to the task of embedding high-dimensional objects into a low-dimensional space in a way that preserves neighbor identities. A Gaussian is centered on each object in the high-dimensional space and the densities under this Gaussian are used to define a probability distribution over all the potential neighbors of the object. The aim of the embedding is to approximate this distribution as well as possible when the same operation is performed in the low-dimensional space. The natural cost function is a sum of Kullback-Leibler divergences, one per object. This leads to a simple gradient for adjusting the positions of objects in the low-dimensional space. Unlike other dimensionality reduction methods, this probabilistic framework makes it easy to represent each high-dimensional object by a mixture of widely separated low-dimensional points. This allows the embeddings of ambiguous objects, like the document count vector for the word "bank", to be close to the embeddings of both "river" and "finance" without forcing outdoor concepts to be located close to corporate ones.

1 Introduction

Automatic dimensionality reduction is an important "toolkit" operation in machine learning, both as a preprocessing step for other algorithms (e.g. to reduce classifier input size) and as a goal in itself for visualization, interpolation, compression, etc. There are many ways to "embed" high-dimensional objects into a lower-dimensional space. Multidimensional scaling methods[1] preserve dissimilarities between items, as measured either by Euclidean distance, some nonlinear squashing of distances, or shortest graph paths as with Isomap[2, 3]. Other methods attempt to preserve local geometry (e.g. LLE[4]) or associate high-dimensional points with a fixed grid of points in the low-dimensional space (e.g. self-organizing maps[5] or their probabilistic extension GTM[6]). All of these methods, however, require each high-dimensional point to be associated with only a single location in the low-dimensional embedding space. This makes it difficult to unfold "many-to-one" mappings in which a single ambiguous object really belongs in several disparate locations in the low-dimensional space. In this paper we define a new notion of embedding based on probable neighbors. Our algorithm, stochastic neighbor embedding (SNE) tries to embed the original data so as to preserve neighborhood identity as well as possible and can be naturally extended to allow multiple embeddings of each object.

2 The SNE algorithm

For each object, i, and each potential neighbor, j, we start by computing the asymmetric probability, p_{ij}, that i would pick j as its neighbor:

$$p_{ij} = \frac{\exp(-d_{ij})}{\sum_{k \neq i} \exp(-d_{ik})} \tag{1}$$

The dissimilarity, d_{ij}, may be given as part of the problem definition (and it need not be symmetric), or it may be computed using the scaled squared Euclidean distance ("affinity") between two high-dimensional points, $\mathbf{x}_i, \mathbf{x}_j$

$$d_{ij} = \frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma_i^2} \tag{2}$$

where σ_i is either set by hand or (as in some of our experiments) found by a binary search for the value of σ_i that makes the entropy of the distribution over neighbors equal to $\log k$. Here, k is the effective number of local neighbors or "perplexity" and is chosen by hand.

In the low-dimensional space we also use Gaussian neighborhoods but with a fixed variance (which we set without loss of generality to be $\frac{1}{2}$) so the probability q_{ij} that point i picks point j as its neighbor is given by

$$q_{ij} = \frac{\exp(-||\mathbf{y}_i - \mathbf{y}_j||^2)}{\sum_{k \neq i} \exp(-||\mathbf{y}_i - \mathbf{y}_k||^2)} \tag{3}$$

The aim of the embedding is to preserve distances to nearby points as well as possible. This is achieved by minimizing a cost function which is a sum of Kullback-Leibler divergences between the high-dimensional and low-dimensional distributions over neighbors for each object:

$$C = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} = \sum_i KL(P_i||Q_i) \tag{4}$$

Notice that making q_{ij} large when p_{ij} is small wastes some of the probability mass in the q distribution so there is a cost for modeling a big distance in the high-dimensional space with a small distance in the low-dimensional space, though it is much less than the cost of modeling a small distance with a big one. In this respect, SNE is an improvement over methods like LLE [4] or SOM [5] in which widely separated objects are often "collapsed" as near neighbors in the embedding. The intuition is that while SNE emphasizes local distances, its cost function cleanly enforces both keeping the embeddings of nearby objects nearby in the low-dimensional space *and* keeping faraway objects relatively far away.

Differentiating C is tedious because $\mathbf{y}_k$ affects q_{ij} via the normalization term in Eq. 3, but the result is simple

$$\frac{\partial C}{\partial \mathbf{y}_i} = 2 \sum_j (\mathbf{y}_i - \mathbf{y}_j)(p_{ij} - q_{ij} + p_{ji} - q_{ji}) \tag{5}$$

which has the nice interpretation of a sum of forces pulling $\mathbf{y}_i$ toward $\mathbf{y}_j$ or pushing it away depending on whether j is observed to be a neighbor more or less often than desired.

Given the gradient, there are many possible ways to minimize C and we have only just begun the search for the best method. Steepest descent in which all of the points are adjusted in parallel is inefficient and can get stuck in poor local optima. Adding random jitter that decreases with time finds much better local optima and is the method we used for the examples in this paper, even though it is still quite slow. We initialize the embedding by putting all the low-dimensional points in random locations very close to the origin. Several other minimization methods, including annealing the perplexity, are discussed in section 5.

3 Application of SNE to image and document collections

As a graphic illustration of the ability of SNE to model high-dimensional, near-neighbor relationships using only two dimensions, we ran the algorithm on a collection of images of handwritten digits and on a set of word-author counts taken from the scanned proceedings of the NIPS conference papers. Both of these datasets are likely to have intrinsic structure in many fewer dimensions than their raw dimensionalities: 256 for the handwritten digits and 13679 for the word counts.

To begin, we used a set of 3000 digit images from the UPS database[7] with equal numbers from each of the five classes 0,1,2,3,4. The variance of the Gaussian around each point in the 256-dimensional image space was set to achieve a perplexity of 15 in the distribution over high-dimensional neighbors. SNE was initialized by putting all the digits in random locations very close to the origin and then was trained using gradient descent with annealed noise.[1] Although SNE was given no information about class labels, it quite cleanly separates the digit groups as shown in figure 1. Furthermore, within each region of the low-dimensional space, SNE has arranged the data so that properties like orientation, skew and stroke-thickness tend to vary smoothly. With a uniform distribution across low-dimensional neighbors, the SNE cost function in Eq. 4 has a value of $3000 \log_e(2999/15) = 15894$ nats. The embedding shown here has a cost of 6719 nats. As a comparison, we also applied principal component analysis (PCA)[8] to the same data. The projection onto the first two principal components, show in figure 2 does not separate classes as cleanly since PCA is much more interested in getting the large separations right which causes it to jumble up some of the boundaries. In this experiment, we chose the digit classes to avoid very similar pairs like 3 and 5 or 7 and 9. When there are more classes and only two available dimensions, SNE does not as cleanly separate very similar pairs.

We have also applied SNE to word-document and word-author matrices calculated from the OCRed text of NIPS volumes 0-12[9]. Figure 3 shows a map locating NIPS authors into two dimensions. Each of the 676 authors who published more than one paper in NIPS vols. 0-12 is shown by a dot; larger red dots and corresponding last names are authors who published six or more papers in that period. The `bow` toolkit[10] was used for part of the pre-processing of the data.[2] Distances d_{ij} were computed as the norm of the difference between log aggregate author word counts, summed across all NIPS papers. Co-authored papers gave fractional counts evenly to all authors. All 13649 words occurring in six or more documents were included, except for stopwords. The σ_i were set to achieve a local perplexity of $k = 25$ neighbors. SNE seems to have grouped authors by broad NIPS field: generative models, support vector machines, neuroscience, reinforcement learning and VLSI all have distinguishable localized regions.

4 A mixture version of SNE

It is easy to modify the definition of the term q_{ij} used in the cost function to allow a distribution over neighbors in the high-dimensional space to be modeled by the weighted average of several different distributions in the low-dimensional space:

$$q_{ij} = \sum_b \pi_{i_b} \sum_c \frac{\pi_{j_c} \exp(-||\mathbf{y}_{i_b} - \mathbf{y}_{j_c}||^2)}{\sum_k \sum_e \pi_{k_e} \exp(-||\mathbf{y}_{i_b} - \mathbf{y}_{k_e}||^2)} \tag{6}$$

[1]For the first 3500 iterations, each 2-D point was jittered by adding Gaussian noise with a standard deviation of 0.3 after each position update. The jitter was then reduced to 0 for a further 500 iterations. On each parallel update, every point is moved by -0.2 times the gradient in Eq. 5. Annealing the jitter from 2 down to 0.018 by multiplying by 0.99 on each iteration achieves very similar results in only 400 iterations.

[2]All the data is available at `http://www.cs.toronto.edu/~roweis/data.html`

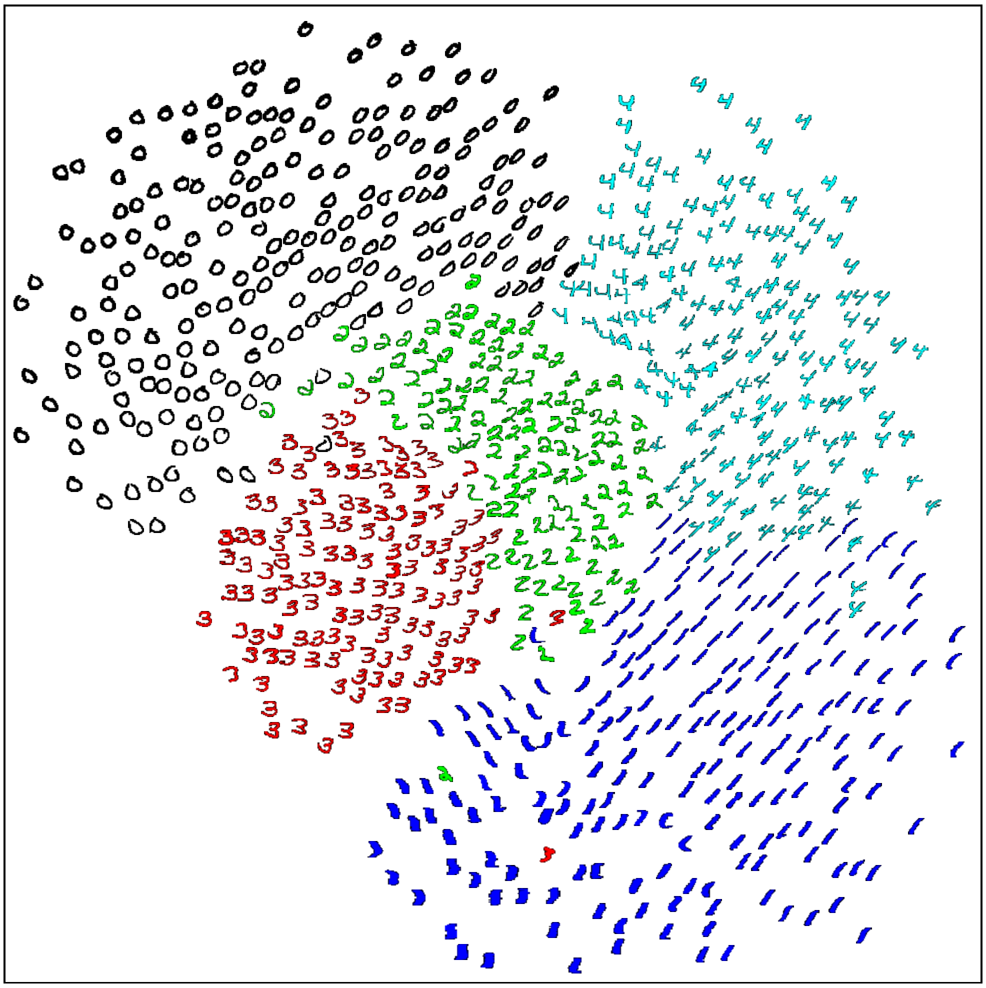

Figure 1: The result of running SNE on 3000 256-dimensional digit images. Images of data vectors $\mathbf{x}_i$ (digits) are shown at the location corresponding to their low-dimensional embedding $\mathbf{y}_i$. The classes are quite well separated even though SNE has no information about class labels. Furthermore, within each class, properties like orientation, skew and stroke-thickness tend to vary smoothly across the space. Not all points are shown: to produce this display, digits are chosen in random order and are only displayed if a 16 x 16 region of the display centered on the 2-D location of the digit in the embedding does not overlap any of the 16 x16 regions for digits that have already been displayed.

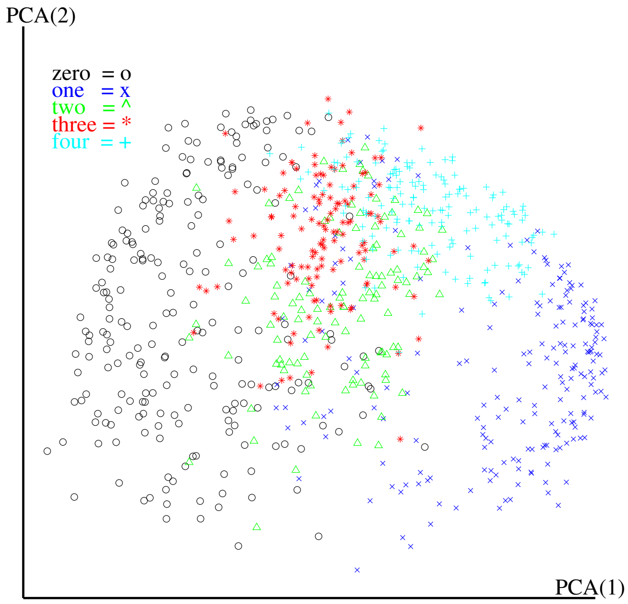

Figure 2: The result of running PCA on the same data as used for the previous figure. Exactly the same digits are displayed as before, using symbols at their embedding locations $\mathbf{y}_i$. PCA is much more interested in getting the large separations right which causes it to jumble up some of the classes.

where π_{i_b} is the mixing proportion of the b^{th} component in the mixture for i and $\mathbf{y}_{i_b}$ is the location of that component in the low-dimensional space.

As a "proof-of-concept", we recently implemented a simplified mixture version in which every object is represented in the low-dimensional space by exactly two components that are constrained to have mixing proportions of 0.5. The two components are pulled together by a force which increases linearly up to a threshold separation. Beyond this threshold the force remains constant. [3]

We took a dataset containing 300 images of each of the digits 2,3,4 and added 100 hybrid images that were each constructed by picking new examples of two of the classes and taking each pixel at random from one of these two "parents". After minimization, 66% of the hybrids and only 19% of the non-hybrids had significantly different locations for their two mixture components. Moreover, the mixture components of each hybrid always lay in the regions of the space devoted to the classes of its two parents and never in the region devoted to the third class. For this example we used a perplexity of 10 in defining the local neighborhoods, multiplied the gradient by -0.7 for each position update, and used a constant jitter of 0.05.

[3] We used a threshold of 0.05. At threshold the force was 0.025 nats per unit length. The low-d space has a natural scale because the variance of the Gaussian used to determine q_{ij} is fixed at 0.5.

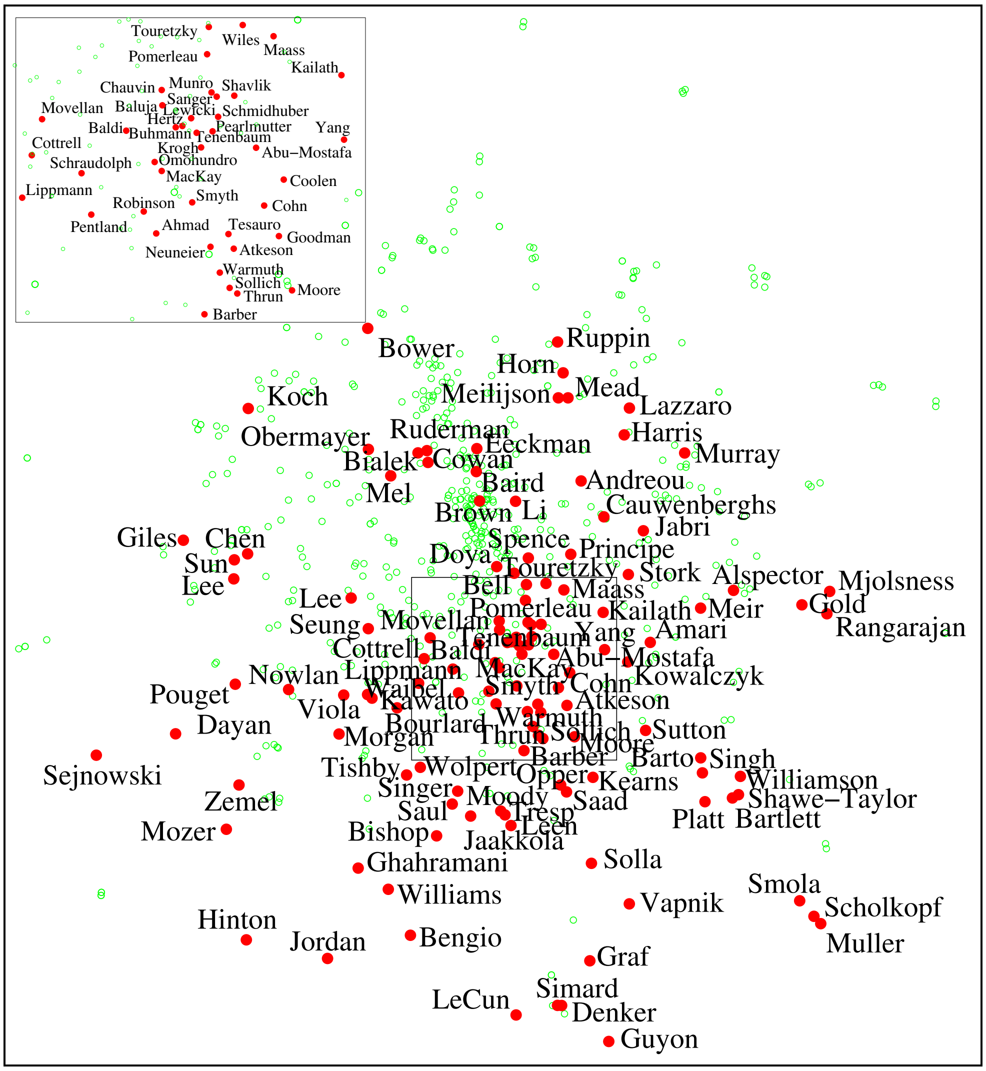

Figure 3: Embedding of NIPS authors into two dimensions. Each of the 676 authors who published more than one paper in NIPS vols. 0-12 is show by a dot; larger red dots and corresponding last names are authors who published six or more papers in that period. The inset in upper left shows a blowup of the crowded boxed central portion of the space. Distances were computed based on log aggregate author word counts (co-authored papers gave fractional counts evenly to all authors). All 13649 words occurring in six or more documents were included, except for stopwords All data is available at `http://www.cs.toronto.edu/~roweis/data.html`.

The example with hybrid digits demonstrates that even the most primitive mixture version of SNE can deal with ambiguous high-dimensional objects that need to be mapped to two widely separated regions of the low-dimensional space. More work needs to be done before SNE is efficient enough to cope with large matrices of document x word counts, but it is the only dimensionality reduction method we know of that promises to treat homonyms sensibly without going back to the original documents to disambiguate each occurrence of the homonym.

Our very simple mixture version of SNE also makes it possible to map a circle onto a line without losing any near neighbor relationships or introducing any new ones. Points near 360^o (on the circle) are mapped to a mixture of two points, one near 360^o (on the line) and one near 0^o (on the line). Obviously, the location of 360^o on the two-dimensional circle gets decided by which pairs of mixture components split first during the optimization. To get this to work without fragmenting the circle into many small line-segments requires hand-tuning of the parameters that control the ease with which two mixture components can be pulled apart.

5 Practical optimization and relationship to PCA/MDS

Our current method is to use steepest descent with added jitter that is slowly reduced. It produces quite good embeddings which demonstrates that the SNE cost function is worth minimizing, but it takes several hours to find good 2-D embeddings for 3000 datapoints. The time per iteration could be reduced considerably by ignoring points for which both p_{ij} and q_{ij} are low, but we would prefer a better search algorithm. Preliminary experiments show that we can get solutions of similar quality with much less computational effort by first annealing the perplexity (with high jitter) and only reducing the jitter after the final perplexity has been reached. This raises the question of what SNE is doing when the variance of the Gaussian centered on each high-dimensional point is very big so that the distribution across neighbors is almost uniform. It is clear that in the high variance limit, the contribution of $\log(p_{ij}/q_{ij})$ to the SNE cost function is just as important for distant neighbors as for close ones. We are currently investigating the relationship between high-variance SNE, PCA and multidimensional scaling.

In the mixture version of SNE there appears to be an interesting way of avoiding local optima that does not involve annealing the jitter or the perplexity. Consider two components in the mixture for an object that are far apart in the low-dimensional space. By raising the mixing proportion of one and lowering the mixing proportion of the other, we can move probability mass from one part of the space to another without it ever appearing at intermediate locations. This type of "probability wormhole" seems like a good way to avoid local optima that arise because a cluster of low-dimensional points must move through a bad region of the space in order to reach a better one.

Yet another search method, which we have used with some success on toy problems, is to provide extra dimensions in the low-dimensional space but to penalize non-zero values on these dimensions. During the search, SNE will use the extra dimensions to go around lower-dimensional barriers but as the penalty on using these dimensions is increased, they will cease to be used.

In order to improve the efficiency of any of these methods, we have also experimented with representing p_{ij} and q_{ij} sparsely, by storing values only for a few of the most probable neighbors and assuming others to be a tiny constant. For p_{ij}, this sparsification needs to be done only once, at the beginning of the algorithm but for q_{ij} this representation needs to be updated as the embedding points move around. With sparse representations, we can efficiently compute a very accurate approximation to the gradient by only keeping terms if $\partial C/\partial \mathbf{y}_i$ for which at least one of $p_{ij}, q_{ij}, p_{ji}, q_{ji}$ has a non-tiny value.

6 Discussion

We have presented a new criterion for mapping high-dimensional points into a low-dimensional space based on stochastic selection of similar neighbors. Unlike self-organizing maps, in which the low-dimensional coordinates are fixed to a grid and the high-dimensional ends are free to move, in SNE the high-dimensional coordinates are fixed to the data and the low-dimensional points move. The gradient of the cost function has an appealing "push-pull" property in which the forces act on $\mathbf{y}_i$ to bring it closer to points it is under-selecting and further from points it is over-selecting as its neighbor. We have shown results of applying this algorithm to image and document collections in which it sensibly placed similar objects nearby in a low dimensional space while keeping dissimilar objects far apart.

SNE is an interesting special case of Linear Relational Embedding (LRE) [11]. In LRE the data consists of triples (*e.g.* Colin has-mother Victoria) and the task is to predict the third term from the other two. LRE learns an N-dimensional vector for each object and an NxN-dimensional matrix for each relation. To predict the third term in a triple, LRE multiplies the vector representing the first term by the matrix representing the relationship and uses the resulting vector as the mean of a Gaussian. Its predictive distribution for the third term is then determined by the relative densities of all known objects under this Gaussian. SNE is just the degenerate version of LRE in which the only relationship is "near" and the matrix representing this relationship is the identity.

Most importantly, because of its probabilistic formulation, SNE has the ability to be extended to mixtures of embedding coordinates in which ambiguous high-dimensional objects (such as the word "bank") can be simultaneously located at several different points in the low-dimensional map.

Acknowledgements This research was funded by NSERC.

References

[1] T. Cox and M. Cox. *Multidimensional Scaling*. Chapman & Hall, London, 1994.

[2] J. Tenenbaum. Mapping a manifold of perceptual observations. In *Advances in Neural Information Processing Systems*, volume 10, pages 682–688. MIT Press, 1998.

[3] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.

[4] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

[5] T. Kohonen. *Self-organization and Associative Memory*. Springer-Verlag, Berlin, 1988.

[6] C. Bishop, M. Svensen, and C. Williams. GTM: The generative topographic mapping. *Neural Computation*, 10:215, 1998.

[7] J. J. Hull. A database for handwritten text recognition research. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 16(5):550–554, May 1994.

[8] I. T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, New York, 1986.

[9] Yann LeCun. Nips online web site. http://nips.djvuzone.org, 2001.

[10] Andrew Kachites McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/ mccallum/bow, 1996.

[11] A. Paccanaro and G.E. Hinton. Learning distributed representations of concepts from relational data using linear relational embedding. *IEEE Transactions on Knowledge and Data Engineering*, 13:232–245, 2000.