
Multidimensional Scaling of Combinatorial Libraries without Explicit Enumeration

DIMITRIS K. AGRAFIOTIS, VICTOR S. LOBANOV

3-Dimensional Pharmaceuticals, Inc., 665 Stockton Drive, Exton, Pennsylvania 19341

Received 2 January 2001; accepted 15 May 2001

ABSTRACT: A novel approach for the multidimensional scaling of large combinatorial libraries is presented. The method employs a multilayer perceptron, which is trained to predict the coordinates of the products on the nonlinear map from pertinent features of their respective building blocks. This method limits the expensive enumeration and descriptor generation to only a small fraction of products and, in addition, relieves the enormous computational effort required for the low-dimensional embedding by conventional iterative multidimensional scaling algorithms. In effect, the method provides an explicit mapping function from reagents to products, and allows the vast majority of compounds to be projected without constructing their connection tables. The advantages of this approach are demonstrated using two combinatorial libraries based on the reductive amination and Ugi reactions, and three descriptor sets that are commonly used in similarity searching, diversity profiling and structure–activity correlation. © 2001 John Wiley & Sons, Inc. *J Comput Chem* 22: 1712–1722, 2001

Keywords: data mining; pattern recognition; dimensionality reduction; feature extraction; multidimensional scaling; nonlinear mapping; Sammon mapping; neural network; multilayer perceptron; combinatorial network; combinatorial library; combinatorial chemistry; high-throughput screening; virtual screening; molecular similarity; molecular diversity; QSAR; QSPR; molecular descriptor

Introduction

This work describes the use of machine learning techniques for the effective representation and visualization of data related to combinatorial li-

braries. More specifically, we address the problem of extracting low-dimensional representations of virtual libraries that preserve some predetermined notion of molecular similarity defined over the abstract entities.¹ This problem is generally known as multidimensional scaling (MDS)^{2,3} or nonlinear mapping (NLM),⁴ and has two primary applications: (1) reducing the dimensionality of high-

Correspondence to: V. S. Lobanov; e-mail: victor@3dp.com

dimensional data in a way that preserves the original relationships of the data objects, and (2) producing Cartesian coordinate vectors from data supplied directly in the form of proximities, so that they can be analyzed with conventional statistical and data mining techniques. Despite nearly 50 years of intensive research and countless applications in many disciplines of science,⁵ MDS remains a formidable problem that is considered intractable for large data sets.

Given a set of k objects, a symmetric matrix, d_{ij} , of dissimilarities between these objects, and a set of images on a m -dimensional display plane $\{\mathbf{y}_i, i = 1, 2, \dots, k; \mathbf{y}_i \in \mathfrak{R}^m\}$, the problem is to place \mathbf{y}_i onto the plane in such a way that their Euclidean distances $\delta_{ij} = \|\mathbf{y}_i - \mathbf{y}_j\|$ approximate as closely as possible the corresponding values d_{ij} . Although an exact projection is only possible when the metric matrix is positive semidefinite, meaningful projections can be obtained even when this criterion is not satisfied. The quality of the projection is determined using a loss function such as Kruskal's stress:

$$S = \sqrt{\frac{\sum_{i < j} (\delta_{ij} - d_{ij})^2}{\sum_{i < j} \delta_{ij}^2}} \quad (1)$$

which is numerically minimized to find the optimal configuration. The actual embedding is carried out in an iterative fashion by: (1) generating an initial set of coordinates \mathbf{y}_i , (2) computing the distances δ_{ij} , (3) finding a new set of coordinates \mathbf{y}_i using a steepest descent algorithm such as Kruskal's linear regression or Guttman's rank-image permutation, and (4) repeating steps 2 and 3 until the change in the stress function falls below some predefined threshold. A particularly popular implementation is Sammon's nonlinear mapping algorithm.⁴ This method uses a modified stress function:

$$E = \sqrt{\frac{\sum_{i < j} [d_{ij} - \delta_{ij}]^2 / d_{ij}}{\sum_{i < j} d_{ij}}} \quad (2)$$

which is minimized using steepest descent. The initial coordinates, \mathbf{y}_i , are determined at random or by some other projection technique such as linear MDS or PCA (when applicable), and are updated using eq. (3):

$$\mathbf{y}_{pq}(m+1) = \mathbf{y}_{pq}(m) - \lambda \Delta_{pq}(m) \quad (3)$$

where m is the iteration number and λ is the learning rate parameter, and

$$\Delta_{pq}(m) = \frac{\partial E(m) / \partial \mathbf{y}_{pq}(m)}{|\partial^2 E(m) / \partial \mathbf{y}_{pq}(m)^2|} \quad (4)$$

There is a wide variety of MDS algorithms involving different loss functions and optimization heuristics, ranging from iterative majorization to Newton–Raphson minimization, Tabu search, simulated annealing, and genetic algorithms. A thorough review can be found in ref. 5.

Unfortunately, the quadratic nature of the stress function [eqs. (1) and (2), and their variants] make these algorithms impractical for large data sets containing more than a few thousand items. Several attempts have been devised to reduce the complexity of the task. Chang and Lee⁶ proposed a heuristic relaxation approach in which a subset of the original objects (the frame) are scaled using a Sammon-like methodology, and the remaining objects are then added to the map by adjusting their distances to the objects in the frame. An alternative approach proposed by Pykett⁷ is to partition the data into a set of disjoint clusters, and map only the cluster prototypes, i.e., the centroids of the pattern vectors in each class. In the resulting two-dimensional plots, the cluster prototypes are represented as circles whose radii are proportional to the spread in their respective classes. Lee⁸ proposed a triangulation method that restricts attention to only a subset of the distances between the data samples. This method positions each point on the plane in a way that preserves its distances from the two nearest neighbors already mapped. An arbitrarily selected reference point may also be used to ensure that the resulting map is globally ordered. Biswas, Jain, and Dubes⁹ later proposed a hybrid approach that combined the ability of Sammon's algorithm to preserve global information with the efficiency of Lee's triangulation method. Although the triangulation can be computed quickly compared to conventional MDS methods, it tries to preserve only a small fraction of distances, and the projection may be difficult to interpret for large data sets.

Recently, we presented an alternative algorithm that combines conventional nonlinear mapping techniques with feed-forward neural networks, and allows the processing of very large data sets that are intractable with conventional methodologies.¹⁰ The method employs a conventional nonlinear mapping algorithm to project a small random sample, and then "learns" the underlying nonlinear transform using a multilayer perceptron trained with the back-propagation algorithm. Once trained, the neural network can be used in a feed-forward manner to project the remaining members of the population as well as new, unseen samples with minimal distortion. The distinct advantage of this approach is that it captures the nonlinear mapping relationship

in an explicit function, and allows the scaling of additional patterns as they become available, without the need to reconstruct the entire map. The basic algorithm was subsequently elaborated,¹¹ and generalized to handle complex distance functions and input data supplied in nonvectorial form.¹² Our method differs substantially from that of Mao and Jain,¹³ who proposed a similar neural network architecture trained with a special back-propagation rule that relies on errors that are functions of the interpattern distances. However, because only a single distance is examined during each iteration, these networks require a very large number of iterations and converge extremely slowly.

The development of our neural network approach was triggered by our need to effectively analyze and visualize large data sets derived from combinatorial chemistry¹⁴ and high-throughput screening.¹⁵ Combinatorial chemistry is becoming an integral part of modern drug discovery and involves data sets of staggering size.^{16–18} Although MDS and NLM have found numerous applications in the chemical sciences,^{19–22} their use in combinatorial chemistry has been limited to the analysis and selection of building blocks.^{23–26} Besides our own work,^{27–29} the only attempt to scale the actual products of a combinatorial library was recently reported by Clark,³⁰ who essentially used Pykett's approach of cluster sampling⁷ coupled with a modified distance function that focused attention on short-range interactions, and ignored distances beyond some predetermined horizon. In our own experience, as with Lee's triangulation approach,⁸ any gains in the reproduction of local relationships occur at the expense of global structure. More importantly, the method requires a potentially expensive preprocessing step (clustering or k-dissimilarity selection), does not provide individual detail, and requires recursive application of MDS if the analyst is interested in any particular region of the feature space.

The present work describes a variant of our original algorithm that is specifically tailored to combinatorial libraries. As it was originally implemented, our method required each compound to be presented to the neural network using a set of descriptors or latent variables that were closely related to the similarity measure used to construct the nonlinear map. Thus, even though the networks were able to project more than 10,000 objects/CPU/s on a modern Intel processor, computing the input parameters required enumeration of the products (i.e., construction of their connection table) and calculation of their pertinent molecular features. For all but the simplest cases, the latter required a much

more substantial computational effort, effectively limiting the throughput of the entire process to a few hundred compounds/CPU/s. Recently, however, we presented evidence that this approach may be wasteful and unnecessary.³¹ Indeed, we found that most of the descriptors that are commonly used in library design can be predicted accurately from the properties of their respective building blocks, including nondecomposable descriptors that cannot be computed by simple addition of fragment contributions. This observation led us to believe that it might be possible to circumvent the generation of product descriptors altogether, and predict the coordinates of the compounds on the nonlinear map directly from reagent data. In the remaining sections we present the algorithmic details of this general scheme and demonstrate its wide applicability across several molecular architectures, descriptor sets, and similarity functions.

Methods

COMBINATORIAL NETWORKS

Combinatorial neural networks (CNNs) are multilayer perceptrons (MLPs) trained to reproduce properties of combinatorial products from pertinent features (descriptors) of their respective building blocks (Fig. 1). In general, CNNs are comprised of an input layer containing $r \times n$ neurons, where r is the number of variation sites in the combinatorial library and n is the number of reagent descriptors, one or more hidden layers containing from 2 to 15 units, depending on the complexity of the transformation, and an output layer having a single neuron for each product feature predicted by the neural network (Fig. 1).³¹ In the case at hand, the output features represent the coordinates of the compounds on the nonlinear map.

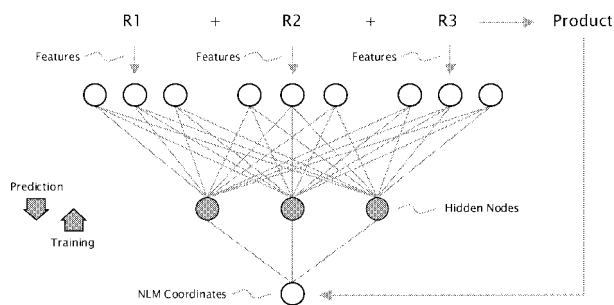


FIGURE 1. Multiple-input single-output (MISO) combinatorial neural network architecture.

Thus, each training sample consists of two sets of parameters: one or more descriptors for each of the building blocks, and one or more coordinates for the respective product. The reagent descriptors are concatenated into a single array, and are presented to the network in the same order ($f_{11}, f_{12}, \dots, f_{1n}, f_{21}, f_{22}, \dots, f_{2n}, \dots, f_{r1}, f_{r2}, \dots, f_{rn}$, where f_{ij} is the j th descriptor of the reagent at the i th variation site).

Our analysis was based on three-layer, fully connected MLPs, trained with the standard error back-propagation algorithm.³² The logistic transfer function $f(x) = 1/(1 + e^{-x})$ was used for both hidden and output layers. Each network was trained for a fixed number of epochs or until a predefined error threshold was met, using a linearly decreasing learning rate from 1.0 to 0.01 and a momentum of 0.8. During each epoch, the training patterns were presented to the network in a randomized order. Initially, the training was constantly monitored with a separate validation set, but no signs of overfitting were observed for any of the training sets and network parameters used in our study. Thus, this validation methodology was abandoned, and the performance of each neural network was evaluated after the system was trained, using a separate test set. A similar resistance to overfitting was also observed in the original nonlinear mapping networks described in refs. 10–12.

The training set is determined using probability sampling. The method is based on the recognition that nonlinear maps derived from small stochastic samples of a large population are virtually indistinguishable from those derived from the entire population, and that the nonlinear transform can be easily encoded in the synaptic parameters of a neural network.¹⁰ Thus, our method consists of the following steps: first, a set of descriptors are computed for each of the reagents that make up the virtual library. These descriptors may be physical, chemical, and/or biological properties computed from the molecular structure, or latent parameters obtained indirectly through statistical methods such as PCA, MDS, NLM, etc. A random sample of products from the virtual library is then identified, and multidimensionally scaled with a conventional MDS or NLM methodology using the descriptors and/or similarity measure of choice. In the present work, the embedding was carried using a fast variant of Sammon's⁴ nonlinear mapping algorithm developed in our group. The resulting coordinates are used as input to a combinatorial network, which is trained to predict the position of compounds on the nonlinear map from the parameters of their re-

spective reagents. Once trained, the neural network can be used in a feed-forward manner to project the remaining members of the virtual library or any subset thereof. In the original, product-based implementation of this idea, we demonstrated that simple three-layered networks with modest complexity can lead to nonlinear maps that have very similar characteristics to those derived by scaling the entire data set with a conventional approach. More importantly, we found that for moderately large data sets ($\sim 10^5$ items), a training set of the order of 1–3% was sufficient to capture the mapping, and that the composition of the training set had very little influence on the quality of the projection as long as it was randomly chosen. For all three descriptor sets used in this study (see below), the reagent parameters that were used as input to the CNNs were the principal components that were required to capture 99% of the total variance in the reagent data.

In summary, the training part of the algorithm involves the following steps:

1. Compute the descriptors of choice for each reagent in the combinatorial library $\{f_{ijk}, i = 1, 2, \dots, r; j = 1, 2, \dots, r; k = 1, 2, \dots, n\}$, where r is the number of variation sites in the library, r_i is the number of building sites blocks at the i th variation site, and n is the number of descriptors used to characterize each reagent.
2. Extract a random subset of products $\{p_i, i = 1, 2, \dots, k; p_i \in P\}$ from the combinatorial library P .
3. Map the patterns p_i onto \mathcal{N}^m using a conventional nonlinear mapping algorithm ($p_i \rightarrow y_i, i = 1, 2, \dots, k; y_i \in \mathcal{N}^m$) using the descriptors and/or similarity function of choice. These descriptors need not be the same as those used to characterize the reagents in step 1.
4. For each training product p_i identified in step 2, identify the corresponding reagents $\{t_{ij}, j = 1, 2, \dots, r\}$, and concatenate their descriptors $f_{1t_{i1}}, f_{2t_{i2}}, \dots, f_{rt_{ir}}$ into a single vector, x_i . Denote $T = \{(x_i, y_i), i = 1, 2, \dots, k\}$ as the training set.
5. Train a neural network, *net*, to recognize the mapping $x_i \rightarrow y_i$ using the input/output pairs in the training set T . Export the network *net* and its associated parameters.

Once the network is trained, the remaining products are mapped using the following procedure:

1. For each product p , identify the corresponding reagents $\{t_j, j = 1, 2, \dots, r\}$, and concatenate their descriptors computed in training step 1, $f_{1t_1}, f_{2t_2}, \dots, f_{rt_r}$, into a single vector, x .
2. Map $x \rightarrow y, x \in \mathfrak{R}^n, y \in \mathfrak{R}^m$ using the neural network *net* derived during training step 5. Store the coordinates y .

SOFTWARE

All computations, including virtual library generation, descriptor calculation, and network training, were carried out using proprietary software written in the C++ programming language and based on 3-Dimensional Pharmaceuticals' Mt++ class library.³³ These programs are part of the DirectedDiversity[®]³⁴ software suite, and were designed to run on all POSIX-compliant Unix and Windows platforms. Parallel execution on systems with multiple CPUs is supported through the multithreading classes of Mt++. The calculations were performed on a Dell Inspiron 7500 laptop computer equipped with a 733 MHz Intel Pentium III processor running Windows 2000 Professional.

Data Sets

The algorithm was tested on two combinatorial libraries that we use routinely for evaluating new library design methodologies. The first is a two-component library based on the reductive amination reaction (Fig. 2). A set of 300 primary amines and 300 aldehydes were selected from the Available Chemicals Directory,³⁵ and were used to generate a virtual library of 90,000 products using the library enumeration classes of DirectedDiversity[®]. These classes take as input lists of reagents supplied in SD or SMILES format, and a reaction scheme written in a proprietary language³⁶ that is based on SMARTS and an extension of the scripting language Tcl.³⁷ All chemically feasible transformations are supported, including multiple reactive functionalities, different stoichiometries, cleavage of protecting groups, stereo-specificity, and many others. The computational and storage requirements of the algorithm are minimal (even a billion-membered library can be

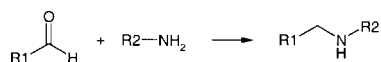


FIGURE 2. Reaction scheme for the reductive amination library.

generated in a few seconds on a personal computer) and scale linearly with the number of reagents.

Each of the 600 reagents and 90,000 products was described by three sets of descriptors: (1) Kier–Hall topological indices (KH), (2) Ghose–Crippen atom types (GC), and (3) ISIS keys (IK). The first is a collection of 117 molecular connectivity indices, kappa shape indices, subgraph counts, information-theoretic indices, Bonchev–Trinajstić indices, and topological state indices.^{38,39} These descriptors have a long, successful history in structure–activity correlation, can be computed directly from the connection table, and are consistent with the medicinal chemists' perception of molecular similarity. Moreover, they have been shown to exhibit proper “neighborhood behavior,”⁴⁰ and are thus well suited for diversity analysis and similarity searching.^{41,42} The Ghose–Crippen descriptors represent an extended set of atom types used by the ALOGP method for the prediction of hydrophobic properties of small organic compounds.⁴³ This set is comprised of 142 descriptors, which represent the counts of the respective atom types in the target molecule. Finally, the ISIS keys are 166-dimensional binary vectors, where each bit encodes the presence or absence of a particular structural feature in the molecule. The bit assignment is based on the fragment dictionary used in the ISIS chemical database management system, and is often referred to as the ISIS *public keys*.

To eliminate redundancy in the data, the Kier–Hall (KH) and Ghose–Crippen (GC) descriptors were independently normalized and decorrelated using principal component analysis (PCA). This process resulted in an orthogonal set of 23 and 47 latent variables, which accounted for 99% of the total variance in the respective data, and formed the basis of the similarity calculations used to construct the nonlinear maps. The same procedure was applied separately to the reagents, resulting in 24 and 45 PCs for the KH and GC descriptors, respectively. To simplify the input to the neural networks, decorrelation was also applied to the ISIS keys, resulting in 66 and 70 PCs for the reagents and products, respectively.

For the KH and GC descriptors, molecular dissimilarity was defined as the Euclidean distance in the decorrelated descriptor space. For the ISIS keys, dissimilarity was assessed using the Tanimoto distance:

$$S = 1 - T \quad (5)$$

where T is the Tanimoto coefficient:

$$T = \frac{|AND(x, y)|}{|IOR(x, y)|} \quad (6)$$

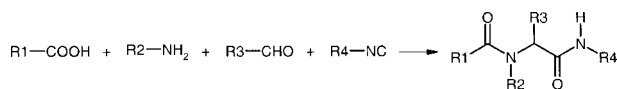


FIGURE 3. Reaction scheme for the Ugi library.

where x and y represent two binary encoded molecules, *AND* is the bitwise “and” operation (a bit in the result is set if both of the corresponding bits in the two operands are set), and *IOR* is the bitwise “inclusive or” operation (a bit in the result is set if the either of corresponding bits in the two operands are set).

The same type of approach was also used to characterize a second, four-component combinatorial library based on the Ugi reaction (Fig. 3). This library was constructed by combining 30 acids, 30 amines, 10 aldehydes, and 10 isonitriles, to yield 90,000 products. As with the previous example, reagents and products were described using the KH, GC, and IK descriptors, which were subsequently decorrelated to a 99% variance cutoff using PCA. This process resulted in 23 KH, 41 GC, and 50 IK components for the reagents, and 21 KH, 39 GC, and 71 IK components for the products, respectively. The results of PCA for the two combinatorial libraries are summarized in Table I. The size of the two collections was intentionally restricted to enable a direct comparison with the steepest descent nonlinear mapping algorithm.

Results and Discussion

Six factors affecting the quality of the nonlinear maps produced by CNNs were examined in this study: (1) network topology and training parameters, (2) sample size, (3) sample composition, (4) structure representation, (5) input and output dimensionality, and (6) combinatorial complexity. To

TABLE I. Number of Principal Components Required to Capture 99% of the Total Variance in the Kier–Hall (KH), Ghose–Crippen (GC), and ISIS Key (IK) Descriptors for the Reductive Amination and Ugi Libraries and Their Respective Building Blocks.

	Amination			Ugi		
	KH	GC	IK	KH	GC	IK
Reagents	24	45	66	23	41	50
Products	23	47	70	21	39	71

get a better appreciation of the relative error introduced by CNN, the projections were compared to those derived by direct NLM and PCA, as well as those derived from our two original, product-based neural net architectures described in refs. 10 and 11. Unlike CNNs, these networks take as input product rather than reagent descriptors, and require enumeration as a preprocessing step. The two architectures differ in the number of networks used to perform the projection: the first (referred to as NN) employs a single perceptron trained over the entire input–output domain,¹⁰ whereas the second (referred to as TNN) uses a tandem architecture involving an initial projection by a global positioning network, followed by a more accurate projection by a specialized network trained over a local domain of the feature space.¹¹ In effect, this method partitions the data space into a set of Voronoi polyhedra, and uses a separate local network to project the patterns in each partition. The global positioning network performs a preliminary mapping to determine which polyhedron a particular pattern falls into, and forwards the sample to the respective local network, which performs the final projection.

The impact of molecular representation was assessed using three different chemical metrics: (1) Euclidean distances based on Kier–Hall PCs, (2) Euclidean distances based on Ghose–Crippen PCs, and (3) Tanimoto distances based on ISIS keys. In the latter case, the input to the neural networks consisted of the reagent (CNN) or product (NN and TNN) PCs that accounted for 99% of the total variance in the respective binary matrices. To increase the separability of the input patterns, the reagent and product descriptors were normalized and decorrelated independently. These descriptor sets encode different aspects of chemical structure and, as a result, lead to qualitatively different estimates of molecular similarity, as manifested by the 2D nonlinear maps of the reductive amination library shown in Figures 9a, 10a, and 11a. In our experience, the Kier–Hall descriptors lead to representations that are more consistent with the chemists’ perception of chemical distance, because they encode global molecular properties such as size, molecular weight, degree of branching, ring character, heteroatom content, etc. GC descriptors, on the other hand, are local in nature, and the general structure of the nonlinear map is often determined by relatively minor structural differences. A good example is the map of the amination library shown in Figure 10a, whose dominant characteristic (the presence of two well-separated clusters) is de-

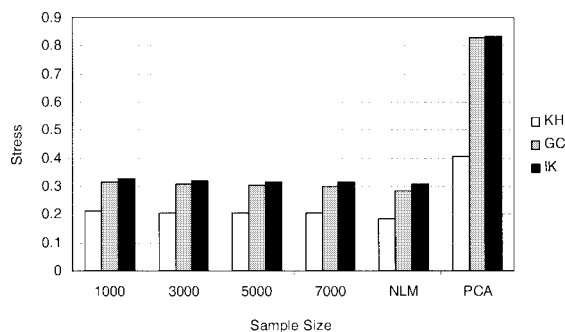


FIGURE 4. Stress of the 2D nonlinear maps of the reductive amination library produced by CNNs using the Kier–Hall (KH), Ghose–Crippen (GC), and ISIS keys (IK) descriptors as a function of the number of compounds used as a training set. All CNNs had 10 hidden neurons and were trained for 500 epochs.

terminated by whether the central amine is aromatic or aliphatic.

Our initial studies aimed at establishing an appropriate sample size, topology, and training parameters for the systems under investigation. Figure 4 shows the dependency of the stress on the number of compounds comprising the training set for a 2D projection of the amination library, using a CNN with 10 hidden units trained for 500 epochs. These results are consistent with our previous findings, which suggested that, for a library of this size, a sample of 1–3% is sufficient to capture the essential details of the nonlinear map. Indeed, while there is some modest improvement in stress going from 1000 to 3000 samples (0.006, 0.008, and 0.006 for the KH, GC, and IK descriptors, respectively), the gain is diminished as additional patterns are inserted into the training set, regardless of the chemical space

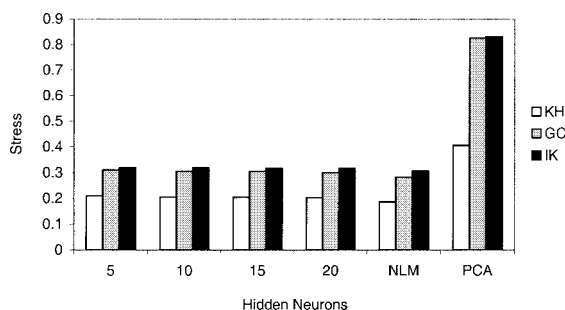


FIGURE 5. Stress of the 2D nonlinear maps of the reductive amination library produced by CNNs using the Kier–Hall (KH), Ghose–Crippen (GC), and ISIS keys (IK) descriptors as a function of the number of hidden neurons. All CNNs were trained for 500 epochs using 3000 randomly chosen compounds as a training set.

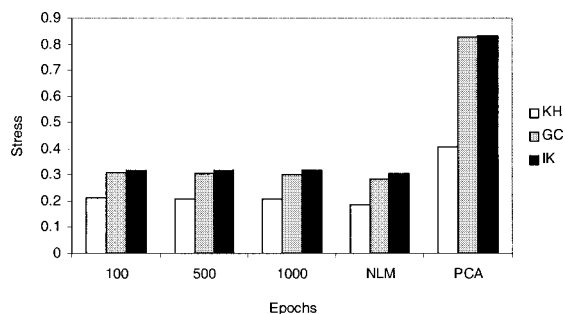


FIGURE 6. Stress of the 2D nonlinear maps of the reductive amination library produced by CNNs using the Kier–Hall (KH), Ghose–Crippen (GC), and ISIS keys (IK) descriptors as a function of the number of training epochs. All CNNs had 10 hidden neurons, and were trained for 500 epochs using 3000 randomly chosen compounds as a training set.

being modeled. We thus concluded that a sample of 3000 compounds represented the best compromise between accuracy and performance, and we used that number for all subsequent calculations.

Also consistent with our past experience is the observation that CNNs do not require an excessive number of hidden units to achieve the desired results. As illustrated in Figure 5, 10 hidden neurons were sufficient to encode the nonlinear mapping, and any further increase resulted in negligible gains at the expense of performance (e.g., increasing the number of hidden units from 10 to 20 decreased the stress by only 0.001, 0.005, and 0.003 for KH, GC, and IK descriptors, respectively, but increased the processing time by a factor of 2 for both training and prediction). The system proved extremely resistant to overfitting, and backpropagation converged reliably within a few hundred epochs (Fig. 6), thus obviating the need for a more robust and potentially expensive procedure for determining the stopping criteria. Interestingly enough, these results were consistent across the descriptor sets, even though the networks differed substantially in the number of free parameters (the KH, GC, and IK CNNs had 48, 90, and 132 input neurons, respectively).

To establish the true generalization capabilities of combinatorial networks and to ensure that the results do not reflect the idiosyncrasies of the individual training sets, 10 different random subsets of 3000 compounds were extracted from the virtual library, and were independently mapped in two dimensions by NLM using each of the three chemical metrics and the first two principal components as a starting configuration. Each of these sets was used to train a series of network models based on the

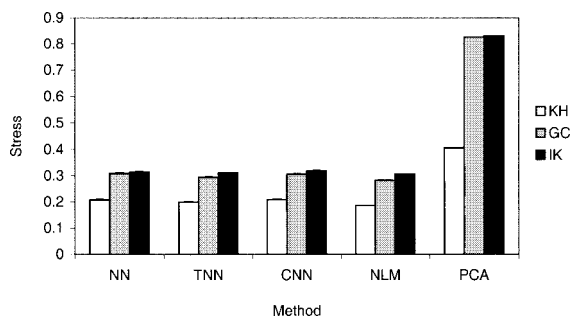


FIGURE 7. Stress of the 2D nonlinear maps of the reductive amination library produced by conventional neural networks (NN), tandem neural networks (TNN), and combinatorial neural networks (CNN) using the Kier–Hall (KH), Ghose–Crippen, (GC) and ISIS keys (IK) descriptors. The columns and their respective error bars represent the average and standard deviation of the stress obtained by each architecture using 10 different random samples of 3000 compounds. All networks had 10 hidden neurons, and were trained for 500 epochs.

NN, TNN, and CNN architectures. Once trained, the models were used to project the entire collection of 90,000 products, and the stress of the resulting maps was computed. The results are summarized in Figure 7. Each column refers to a particular model, and represents the mean and standard deviation of the stress of the entire virtual library, computed over all 10 trials (training sets). The CNN maps had an average stress of 0.208 ± 0.003 , 0.306 ± 0.002 , and 0.318 ± 0.001 for the KH, GC, and IK descriptors, re-

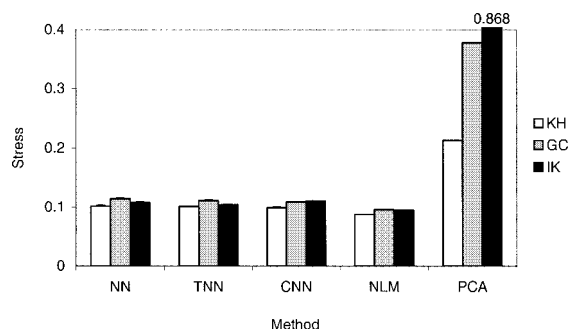
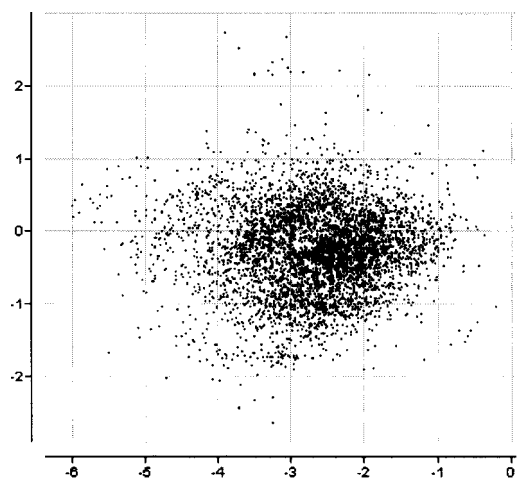
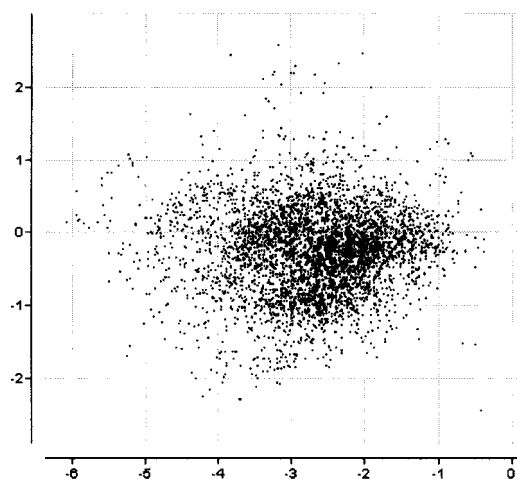


FIGURE 8. Stress of the 4D, 7D, and 6D nonlinear maps of the reductive amination library produced by conventional neural networks (NN), tandem neural networks (TNN), and combinatorial neural networks (CNN) using the Kier–Hall (KH), Ghose–Crippen (GC), and ISIS keys (IK) descriptors, respectively. The columns and their respective error bars represent the average and standard deviation of the stress obtained by each architecture using 10 different random samples of 3000 compounds. All networks had 10 hidden neurons, and were trained for 500 epochs.



a



b

FIGURE 9. 2D nonlinear map for the reductive amination library obtained by (a) NLM, and (b) CNN using the Kier–Hall descriptors. The CNN map was obtained using a network with 10 hidden neurons trained for 500 epochs using a random sample of 3000 compounds.

spectively, suggesting that the predictive ability of CNNs does not show any significant dependence on the composition of the training set, as long as it is randomly chosen and therefore representative of the population from which it is drawn.

In all three cases, CNNs produced maps that were comparable to those derived by the product-based networks (Figs. 9, 10, and 11), with the average relative error being limited to 0.008 (KH), 0.011

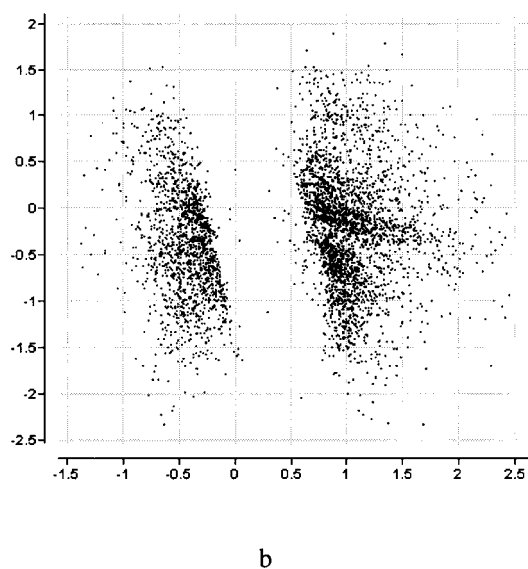
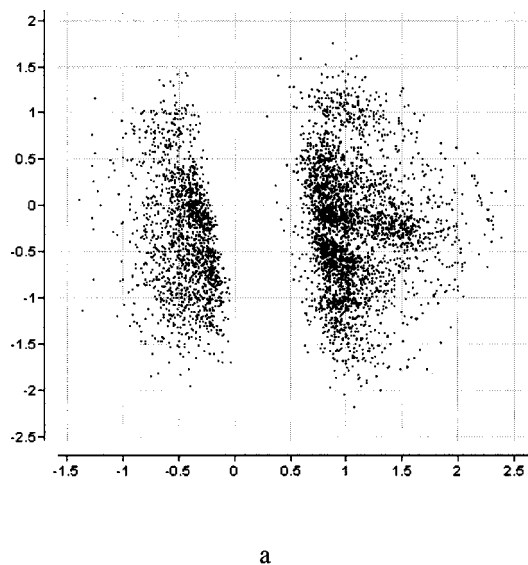


FIGURE 10. 2D nonlinear map for the reductive amination library obtained by (a) NLM, and (b) CNN using the Ghose–Crippen descriptors. The CNN map was obtained using a network with 10 hidden neurons trained for 500 epochs using a random sample of 3000 compounds.

(GC), and 0.007 (IK) stress units relative to TNN, and 0.000 (KH), 0.003 (GC), and 0.004 (IK) relative to NN, respectively. Given the fact that the current CNN implementation involves a single mapping device, it is almost certain that this error could be further reduced with the introduction of local learning. Even though CNNs are substantially more flexible than the product-based NNs (they essentially have r times as many inputs, where r is the

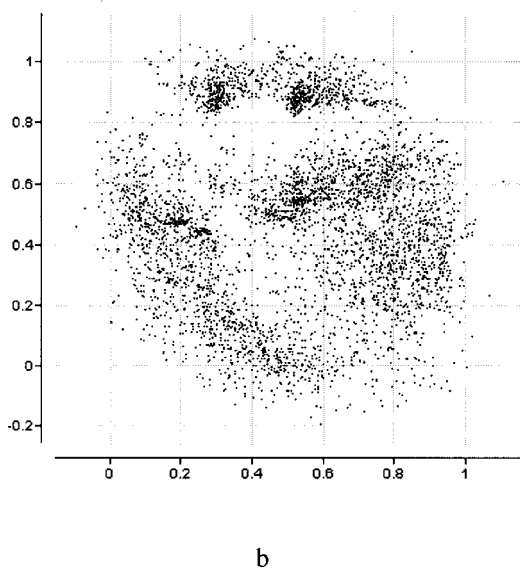
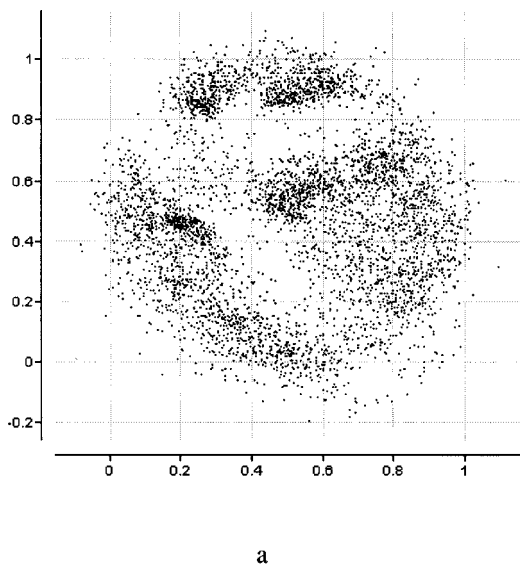


FIGURE 11. 2D nonlinear map for the reductive amination library obtained by (a) NLM, and (b) CNN using the ISIS keys descriptors. The CNN map was obtained using a network with 10 hidden neurons trained for 500 epochs using a random sample of 3000 compounds.

number of variation sites), these results are still impressive, given the fact that the connection between input and output data is less direct.

The preceding discussion focused exclusively on 2D projections. Although visualization is one of the main applications of MDS, the method can also be used for embedding objects into higher dimensions to simplify their analysis by established statistical methods. Martin et al., for example, has often used

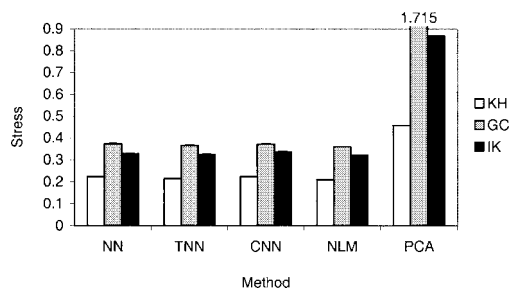


FIGURE 12. Stress of the 2D nonlinear maps of the Ugi library produced by conventional neural networks (NN), tandem neural networks (TNN), and combinatorial neural networks (CNN) using the Kier–Hall (KH), Ghose–Crippen (GC), and ISIS keys (IK) descriptors. The columns and their respective error bars represent the average and standard deviation of the stress obtained by each architecture using 10 different random samples of 3000 compounds. All networks had 10 hidden neurons and were trained for 500 epochs.

this technique to convert binary molecular fingerprints into Cartesian vectors so that they could be used for reagent selection using D-optimal experimental design^{23–25} determine whether our method is effective in higher dimensions, we nonlinearly mapped the KH, GC, and IK descriptors onto 4, 7, and 6 dimensions, respectively, which was the number required to reproduce the respective pairwise distances to within 0.1 stress units (roughly equivalent to an average error of ~10%). The results are summarized in Figure 8. Again, the three algorithms produced comparable maps, with CNN being marginally better for the KH and GC descriptors, and marginally worse for the ISIS keys. The standard deviation was limited to 0.001, 0.0004, and 0.0005 for KH, GC, and IK, respectively, indicating a similar lack of dependence on the choice of the training set.

Finally, to determine whether the method works with different types of functional groups and more complex combinatorial libraries, the algorithm was tested on a four-component library based on the Ugi reaction. A 90,000-member library was enumerated and subjected to the same type of analysis followed in the previous example. As with the amination library, we found 3000 training samples, 10 hidden units, and 500 epochs to be sufficient for reproducing the nonlinear map within 0.01 stress units in two or more dimensions (in this case, 5, 10, and 7 dimensions were required to reproduce the KH, GC, and IK distances to within 0.1 units of stress). The results in Figure 12 (2D) and Figure 13 (5D, 10D, and 7D) reveal identical trends to those established for the amination library, and show compara-

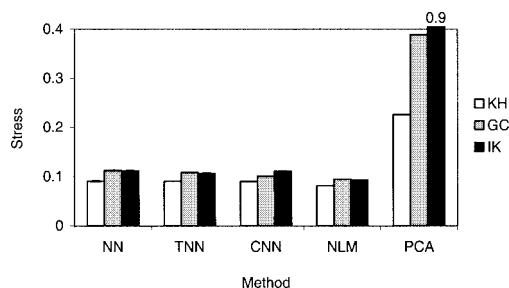


FIGURE 13. Stress of the 5D, 10D, and 7D nonlinear maps of the Ugi library produced by conventional neural networks (NN), tandem neural networks (TNN), and combinatorial neural networks (CNN) using the Kier–Hall (KH), Ghose–Crippen (GC), and ISIS keys (IK) descriptors, respectively. The columns and their respective error bars represent the average and standard deviation of the stress obtained by each architecture using 10 different random samples of 3000 compounds. All networks had 10 hidden neurons and were trained for 500 epochs.

ble performance for the reagent and product-based approaches, with no significant sensitivity to the choice of the training set.

Conclusions

This work described an efficient solution to a long-standing problem in computational data analysis known as multidimensional scaling, as it applies to combinatorial libraries. We have shown that by combining machine learning techniques with probability sampling, it is possible to multidimensionally scale large combinatorial libraries that are intractable with conventional techniques. What makes this approach particularly appealing is the fact that the coordinates of the compounds on the nonlinear map can be inferred from the properties of their building blocks, effectively eliminating the need to enumerate and describe every individual product in the virtual library. This method is fast and broadly applicable, and can be used to analyze libraries of any origin, molecular architecture, and combinatorial complexity.

Acknowledgments

We wish to thank Dr. Dmitrii N. Rassokhin and Dr. Sergei Izrailev of 3-Dimensional Pharmaceuticals, Inc. for many useful discussions, and Dr. Raymond F. Salemme for his insightful comments and support of this work.

References

1. Johnson, M. A.; Maggiora, G. M. *Concepts and Applications of Molecular Similarity*; Wiley: New York, 1990.
2. Torgeson, W. S. *Psychometrika* 1952, 17, 401.
3. Kruskal, J. B. *Psychometrika* 1964, 29, 115.
4. Sammon, J. W. *IEEE Trans Comp* 1969, C-18, 401.
5. Borg, I.; Groenen, P. J. F. *Modern Multidimensional Scaling: Theory and Applications*; Springer: New York, 1997.
6. Chang, C. L.; Lee R. C. T. *IEEE Trans Syst Man Cybern* 1973, SMC-3, 197.
7. Pykett, C. E. *Electron Lett* 1978, 14, 799.
8. Slagle, J. R.; Blum, H.; Lee, R. C. Y. *IEEE Trans Comput* 1977, C-27, 288.
9. Biswas, G.; Jain, A. K.; Dubes, R. C. *IEEE Trans Pattern Anal Machine Intell* 1981, PAMI-3, 701.
10. Agrafiotis, D. K.; Lobanov, V. S. *J Chem Info Comput Sci* 2000, 40, 1356.
11. Rassokhin, D. N.; Lobanov, V. S.; Agrafiotis, D. K. *J Comput Chem* 2001, 22, 373.
12. Agrafiotis, D. K.; Rassokhin, D. N.; Lobanov, V. S. *J Comput Chem* 2001, 22, 488.
13. Mao, J.; Jain, A. K. *IEEE Trans Neural Networks* 1995, 6, 296.
14. Thompson, L. A.; Ellman, J. A. *Chem Rev* 1996, 96, 296.
15. Sittampalam, G. S.; Kahl, S. D.; Janzen, W. P. *Curr Opin Chem Biol* 1997, 1, 384.
16. Agrafiotis, D. K. In *The Encyclopedia of Computational Chemistry*; Schleyer, P. v. R.; Allinger, N. L.; Clark, T.; Gasteiger, J.; Kollman, P. A.; Schaefer, III, H. F.; Schreiner, P. R., Eds.; John Wiley & Sons: Chichester, 1998, p. 742.
17. Agrafiotis, D. K.; Myslik, J. C.; Salemme, F. R. *Mol Diversity* 1999, 4, 1.
18. Agrafiotis, D. K.; Lobanov, V. S.; Rassokhin, D. N.; Izrailev, S. In *Virtual Screening for Bioactive Molecules*; Böhm, H.-J.; Schneider, G., Eds.; Wiley-VCH: Weinheim, 2000, p. 265.
19. Kowalski, B. R.; Bender, C. F. *J Am Chem Soc* 1973, 95, 686.
20. Domine, D.; Devillers, J.; Chastrette, M.; Karcher, W. *J Chemometrics* 1993, 7, 227.
21. Agrafiotis, D. K. *Protein Sci* 1997, 6, 287.
22. Chen, Z.-P.; Jian, J.-H. Li, Y.; Yu, R.-Q. *Chemometrics Intell Lab Syst* 1999, 45, 409.
23. Martin, E. J.; Blaney, J. M.; Siani, M. A.; Spellmeyer, D. C.; Wong, A. K.; Moos, W. H. *J Med Chem* 1995, 38, 1431.
24. Martin, E. J.; Critchlow, R. E. *J Comb Chem* 1999, 1, 32.
25. Martin, E. J.; Wong, A. *J Chem Info Comput Sci* 2000, 40, 215.
26. Martin, E. J.; Hoefel, T. J. *J Mol Graphics Modell* 2000, 18, 383.
27. Agrafiotis, D. K. *J Chem Info Comput Sci* 1997, 37, 841.
28. Agrafiotis, D. K. *J Chem Info Comput Sci* 1997, 37, 576.
29. Rassokhin, D. N.; Agrafiotis, D. K. *J Mol Graphics Modell* 2000, 18, 370.
30. Clark, R. D. Patterson, D. E. Soltanshahi, F.; Balke, J. F.; Matthew, J. B. *J Mol Graphics Modell* 2000, 18, 404.
31. Lobanov, V. S.; Agrafiotis, D. K. *J Mol Graphics Modell* 2001, 19, 571.
32. Haykin, S. *Neural Networks*; Macmillan: New York, 1994.
33. Copyright © 3-Dimensional Pharmaceuticals, Inc., 1994–2001.
34. Agrafiotis, D. K.; Bone, R. F.; Salemme, F. R.; Soll, R. M. US Pat. 5,453,564 (1995); 5,574,656 (1996); 5,685,711 (1997); and 5,901,069 (1999).
35. MDL Information System, Inc., 140 Catalina Street, San Leandro, CA 94577.
36. Lobanov, V. S.; Agrafiotis, D. K. *Combin. Chem. and High-Throughput Screen.*, to appear.
37. Ousterhout, J. K. *Tcl and the Tk Toolkit*; Addison-Wesley: New York, 1994.
38. Bonchev, D.; Trinajstić, N. *J Chem Phys* 1997, 67, 4517.
39. Hall, L. H.; Kier, L. B. In *Reviews of Computational Chemistry*; Boyd, D. B.; Lipkowitz, K. B., Eds.; VCH Publishers: 1991, p. 367, Chap. 9.
40. Patterson, D. E.; Cramer, R. D.; Ferguson, A. M.; Clark, R. D.; Weinberger, L. E. *J Med Chem* 1996, 39, 3049.
41. Lewis, R. A.; Mason, J. S.; McLay, I. M. *J Chem Info Comput Sci* 1997, 37, 599.
42. Lobanov, V. S.; Agrafiotis, D. K. *J Chem Info Comput Sci* 2000, 40, 460.
43. Ghose, A.; Viswandhan, V. N.; Wendoloski, J. J. *J Phys Chem A* 1998, 102, 3762.