Journal of Bioinformatics and Computational Biology
Vol. 3, No. 3 (2005) 645–676
© Imperial College Press

Imperial College Press
www.icpress.co.uk

DETERMINING RELEVANT FEATURES TO RECOGNIZE ELECTRON DENSITY PATTERNS IN X-RAY PROTEIN CRYSTALLOGRAPHY

KRESHNA GOPAL
Department of Computer Science, Texas A&M University
301 H.R. Bright Building, College Station TX 77843-3112, USA
kgopal@cs.tamu.edu

TOD D. ROMO
Department of Biochemistry & Biophysics, Texas A&M University
103 Bio-Bio Building, College Station TX 77843-2128, USA
tromo@tamu.edu

JAMES C. SACCHETTINI
Department of Biochemistry & Biophysics, Texas A&M University
103 Bio-Bio Building, College Station TX 77843-2128, USA
sacchett@tamu.edu

THOMAS R. IOERGER
Department of Computer Science, Texas A&M University
301 H.R. Bright Building, College Station TX 77843-3112, USA
ioerger@cs.tamu.edu

Received 31 August 2004
Revised 1 December 2004
Accepted 10 December 2004

High-throughput computational methods in X-ray protein crystallography are indispensable to meet the goals of structural genomics. In particular, automated interpretation of electron density maps, especially those at mediocre resolution, can significantly speed up the protein structure determination process. TEXTAL$^{\text{TM}}$ is a software application that uses pattern recognition, case-based reasoning and nearest neighbor learning to produce reasonably refined molecular models, even with average quality data. In this work, we discuss a key issue to enable fast and accurate interpretation of typically noisy electron density data: what features should be used to characterize the density patterns, and how relevant are they? We discuss the challenges of constructing features in this domain, and describe SLIDER, an algorithm to determine the weights of these features. SLIDER searches a space of weights using ranking of matching patterns (relative to mismatching ones) as its evaluation function. Exhaustive search being intractable, SLIDER adopts a greedy approach that judiciously restricts the search space only to weight values that cause the ranking of good matches to change.

We show that SLIDER contributes significantly in finding the similarity between density patterns, and discuss the sensitivity of feature relevance to the underlying similarity metric.

Keywords: Protein structure determination; X-ray crystallography; pattern recognition; case-based reasoning; nearest neighbor learning; feature relevance.

1. Introduction

X-ray diffraction methods account for over 80% of proteins whose three-dimensional (3D) structures have been determined.[1] However, high cost and low throughput of X-ray crystallography and other experimental methods (like NMR) result in solving only a small fraction of the new proteins being discovered. In fact, the ratio of solved crystal structures to the number of discovered proteins is about 0.15.[2] At the same time, DNA-sequencing projects are producing an overwhelming amount of sequencing information; keeping up the protein structure determination rate with this growth of genomic information has become a major challenge. The structural genomics initiative[3] is a worldwide effort aimed at solving protein structures in a high-throughput mode, primarily by X-ray crystallography and NMR methods. There is also a surge of interest in predicting structure through *ab initio* methods (based directly on physical principles) and comparative modeling techniques (based on comparison to known structures).

High-throughput structure determination requires automation to reduce human intervention, especially in bottleneck steps. One such step in X-ray crystallography is the *electron density map* interpretation, where crystallographers have to recognize 3D patterns of electron density around the protein, and fit amino acids into the density patterns in the right orientation (Fig. 1). Although crystallographers use molecular graphics programs, the process is nonetheless tedious and time-consuming, especially if the data is noisy or at low resolution.[4] Map interpretation may take of the order of weeks or months, and can often be subjective in nature.[5,6]

TEXTAL$^{\mathrm{TM}}$ automates this process of map interpretation — it takes an electron density map as input and outputs a model (atoms and their 3D coordinates) of the macromolecule in a few hours. One of the salient features of TEXTAL$^{\mathrm{TM}}$ is that it has been designed to work even with average or poor quality density data (in the 2.5–3.0Å resolution range). Existing density interpretation programs[7–10] typically require good quality data and/or human intervention.

TEXTAL$^{\mathrm{TM}}$ uses case-based reasoning[11,12] and nearest neighbor learning[13] to recognize patterns of electron density (in small spherical regions in a density map) by comparing them to existing patterns whose structures are known and stored in a database. Matching patterns are identified in the database, corresponding solved local structures are retrieved and assembled together to model the protein incrementally, guided by domain knowledge, either explicitly stated (like typical stereochemical constraints) or implicitly encoded in the solved structures.

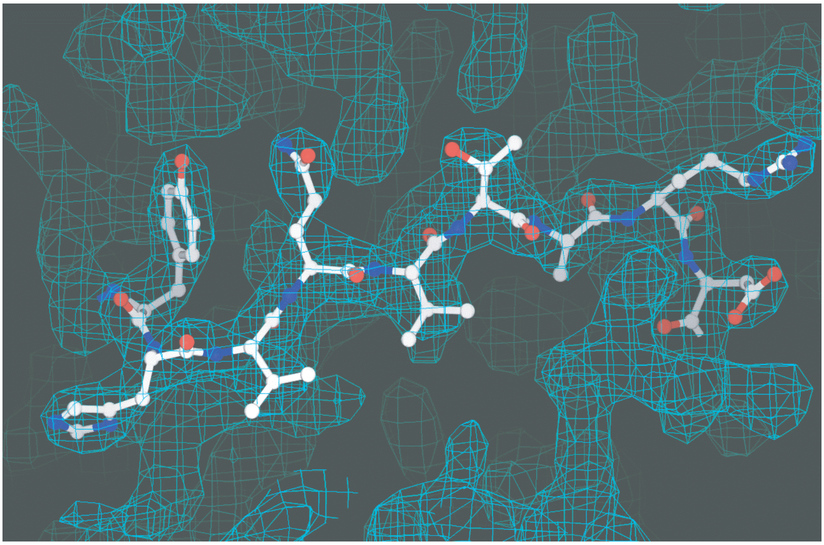


Fig. 1. Example of electron density around a fragment of a protein structure. The fragment shown consists of nine residues (144-122) of a β-strand in 1HQZ, an actin-binding protein from Yeast. The electron density map has been calculated (back-transformed) from the solved structure at 2.8 Å, and is shown at a contour level of $\sim 1\sigma$. This stereo view has been made with Spock, a graphics program written by Dr. Jon A. Christopher (http://quorum.tamu.edu).

A salient characteristic of nearest neighbor learning and case-based reasoning methods is their sensitivity to the distance (or similarity) metric being used to compare patterns. One of the key requirements for accurate determination of similarity is the correct choice of features to be used in the metric. In TEXTAL$^{\mathrm{TM}}$, 76 numeric features were manually extracted based on domain knowledge, as well as intuitions on what would be relevant in discriminating electron density patterns. But not all the features may be relevant, in all situations. Irrelevant features effectively introduce noise in the data, and can mislead the matching of patterns. In this paper, we focus our attention on the challenges of designing features in this domain, the importance of determining relevance of features, and the methods we use to address these problems in TEXTAL$^{\mathrm{TM}}$. We present SLIDER, an algorithm that assigns numeric weights to features such that the similarity measure is improved, which leads to better case retrieval and ultimately enhanced model-building.

Feature selection and weighting can be viewed as a search over a space of weight vectors, with an evaluation function to assess each weight vector. In SLIDER we propose the following evaluation function: given an instance (i.e. a spherical region of electron density), we first find a matching region and a set of mismatching ones (using an independent, objective measure). Then we evaluate how well a given feature-based distance measure (that uses the weights) ranks the match relative to the mismatches. This is done for a set of instances, and the average ranking of the matches reflects how good the weight vector is.

The space of weight vectors is very large, and an exhaustive search is clearly intractable; even if we consider two possible weights (0 and 1) for each feature i.e. it is either irrelevant or relevant, then there are 2^n possible subsets of n features. Therefore we search only those weights at which matching and mismatching patterns switch as nearer neighbors to query instances. The evaluation function based on ranking of matches will in fact change at those specific weights, and thus the search becomes very effective. But the algorithm is greedy in that decisions are based on finding the "optimal" weight of one feature at a time, which may lead to solutions that are only locally optimal. Nonetheless, SLIDER outputs weights that significantly contribute to the accuracy of case matching, and improves on the initial (non-weighted) set of features as defined by human experts. We also observe that the best weight vector as determined by SLIDER varies for different distance metrics. We argue that feature relevance can be, to a certain extent, sensitive to the underlying distance metric it is used for. Although SLIDER was motivated by the crystallographic map interpretation problem, the algorithm is based on general principles, and can be applied to many other applications, especially those with high-dimensional and noisy data.

The rest of this paper is organized as follows:

- Section 2 provides more details on the principles and challenges of the X-ray crystallography method to determine protein structures, and summarize other work related to automated interpretation of electron density maps.
- The TEXTAL$^{\text{TM}}$ system is briefly described in its entirety in Sec. 3; this provides the context and motivation for feature weighting.
- Section 4 discusses the general feature selection and weighting problem, and presents the main approaches proposed in the literature on artificial intelligence and machine learning.
- Section 5 describes the features that experts defined in this domain, and the rationale behind the choices.
- The SLIDER algorithm is presented in details in Sec. 6, followed by empirical results and a discussion in Secs. 7 and 8 respectively.

2. X-Ray Protein Crystallography

X-ray crystallography is the most widely used technique to accurately determine the structure of proteins and other macromolecules. It is based on the fact that X-rays can be diffracted by crystals. In fact, X-rays are scattered by the electrons around atoms, and this scattering from periodic arrangements of atoms in a crystal results in diffraction patterns. These patterns are detected and used to reconstruct the electron density, from which the macromolecular model (i.e. atoms and their coordinates) can be determined. X-ray crystallography usually produces accurate molecular structures, from global folds to atomic-level bonding details.

Crystallographic structure determination involves many steps: first the protein has to be isolated, purified and crystallized. Protein molecules being long chains of typically hundreds of amino acids, they fold in irregular shapes, and are not suited to be stacked in a regular crystal lattice. Thus protein crystals are generally small, fragile, and contain about 50% solvent, on average — this makes crystallization a challenge, and usually requires experimenting with many conditions.

After crystallization, diffraction data has to be collected; diffraction spots are obtained when X-rays are shone through the crystal, based on the arrangement of the atoms. The intensities of the diffracted spots are determined, and an electron density map is calculated by the Fourier transformation of the intensities and corresponding estimated phases. In fact, the spots contain information only about the intensity of diffracted waves; the phase information, which is also required for inferring the map, is lost and has to be approximated by other methods, such as multiwavelength anomalous diffraction (MAD)[14,15] or multiple isomorphous replacement (MIR).[16,17] This is known as the *phase problem*. Furthermore, the sample of points at which intensities can be collected is limited, which constrains the degree to which atoms can be distinguished from one another. This imposes limits on the *resolution* of the map, measured in Å ($1\,\text{Å} = 10^{-10}\,\text{m}$).

Solving the structure essentially means fitting *amino acids* or *residues* in the density map, where each amino acid has several rotational degrees of freedom and can adopt various conformations. The solved structure can then be used to obtain better phase information and generate an improved map, which can then be reinterpreted. This process can go through many cycles, and it may take weeks or sometimes months of effort for an expert crystallographer to produce a refined structure, even with the help of molecular 3D visualization programs. Manual structure determination can be laborious and inaccurate, depending on factors like the size of the structure, resolution of the data, the complexity of the crystal packing, etc. There can be many sources of errors and noise, which distort the electron density map, making interpretation difficult.[4] There is also a subjective component to model building;[5,6] decisions of an expert are often based on what seems most reasonable in specific situations, with oftentimes little scope for generalization.

Various tools and techniques have been proposed for automated protein model building: treating modeling and phase refinement as one unified procedure using free atom insertion in ARP/wARP,[7] expert systems,[18,19] molecular-scene analysis,[20,21] database search,[22–24] using templates from the Protein Data Bank,[25] template convolution and other FFT-based approaches,[8] maximum-likelihood density modification,[26] heuristic methods based on optimizing the fit of rotamers or backbone in the density,[9,10,27] etc. Common problems with many of these approaches include dependency on user-intervention and/or high quality data (i.e. with 2 Å resolution, or better). In contrast, TEXTAL$^{\text{TM}}$ has been designed to be fully automated, and to work with average and even low quality data (around 2.8 Å resolution); most maps are, in fact, noisy and fall in the low-medium resolution category[28] due to difficulties in protein crystallization and other limitations of the data collection methods.

3. The TEXTAL$^{\mathrm{TM}}$ System

TEXTAL$^{\mathrm{TM}}$ fully automates electron density map interpretation, thereby saving considerable effort required by human experts. TEXTAL$^{\mathrm{TM}}$ attempts to mimic the way a crystallographer approaches the problem — first the backbone is modeled i.e. the positions of the Cα carbon atoms are determined. Side chains are then fitted into the density based on how the density looks around the Cα atoms. After side chain modeling, the structure is refined through a number of post-processing routines to improve the fit to the density[29] and to align with the protein sequence.[30] The model obtained can then be manually improved by the crystallographer, and used to obtain better phases. An improved map can thus be generated, and fed again to TEXTAL$^{\mathrm{TM}}$.

Figure 2 gives the overall architecture of TEXTAL$^{\mathrm{TM}}$. In this section, we provide a very brief description of the system, with emphasis on issues related to feature relevance. For a more detailed discussion on TEXTAL$^{\mathrm{TM}}$ and its sub-systems, refer to previous work[31–34] and http://textal.tamu.edu:12321.

3.1. *Backbone modeling*

The backbone determination is done by a system called CAPRA, or C-Alpha Pattern Recognition Algorithm.[32] As shown in the architecture of the TEXTAL$^{\mathrm{TM}}$ system (Fig. 2), CAPRA comprises of the following steps:

- SCALE MAP: the input map, in XPLOR[35] format, is scaled i.e. density values are normalized to have a mean and a standard deviation comparable to those in other maps. This is done to enable meaningful comparison between regions from different maps.
- TRACE MAP: this is analogous to skeletonization programs;[36,37] given an electron density map, this routine creates a chain of grid points along the medial axis of the contours of the map. These trace points essentially represent the shape of the density contours in a compact form.
- CALCULATE FEATURES: takes a scaled map as input, and computes numeric features of spherical regions defined around each of the trace points from TRACE MAP. These features are subsequently used to determine the positions of Cα atoms, and to model side chains.
- PREDICT Cα POSITIONS: uses a neural network to determine the distances of a set of points in a map to true Cα's. The inputs to the network are 38 numeric features in a 5 Å sphere centered on each trace point. Points with the lowest predicted distances to Cα's are picked as an initial set of Cα's, with preference to those that are about 3.8 Å apart (this is the typical distance between Cα atoms in proteins).
- BUILD CHAINS: heuristic search methods are used to link putative Cα's in a map into a set of linear chains. The heuristic favors those chains that conform to secondary structures and other typical motifs in proteins. The output is a PDB

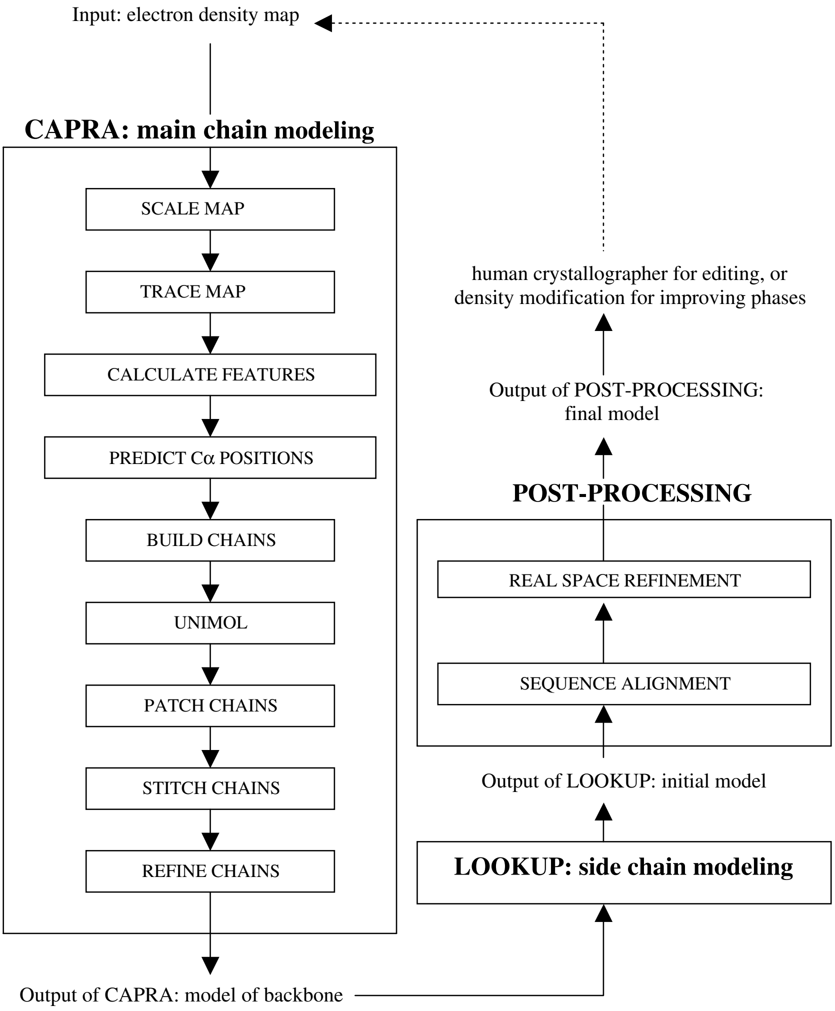

Input: electron density map
CAPRA: main chain modeling
SCALE MAP
TRACE MAP
CALCULATE FEATURES
PREDICT Cα POSITIONS
BUILD CHAINS
UNIMOL
PATCH CHAINS
STITCH CHAINS
REFINE CHAINS
Output of CAPRA: model of backbone
LOOKUP: side chain modeling
Output of LOOKUP: initial model
POST-PROCESSING
SEQUENCE ALIGNMENT
REAL SPACE REFINEMENT
Output of POST-PROCESSING: final model
human crystallographer for editing, or density modification for improving phases

Fig. 2. Architecture of TEXTAL^{TM}, showing the three major components: CAPRA, LOOKUP and POST-PROCESSING. The crystallographer may use the model produced by TEXTAL^{TM} to improve phases and create a better map, which can be re-inputted to TEXTAL^{TM}; this process can go through several iterations.

file, which represents information about the coordinates of Cα's and how they are linked together.

- UNIMOL: reduces the complexity of a set of chains by removing redundant symmetry copies. UNIMOL keeps chains that are near the center of the model and eliminates symmetric chain copies around the periphery. The result is a simplified model that is easier to interpret.
- PATCH CHAINS: connects chains together in regions where density is weak by adjusting the contour level.

- STITCH CHAINS: uses a case-based reasoning approach to stitch together nearby ends of disconnected chains, especially in regions where the backbone makes a loop and/or the density appears distorted or missing.
- REFINE CHAINS: improves the geometry of Cα chains with respect to typical atomic bond lengths and angles.

3.2. *Side chain modeling*

Modeling of side chains is done by a sub-system called LOOKUP — the program takes a set of Cα chains and an electron density map as inputs, and uses case-based reasoning and nearest neighbor learning to effectively and efficiently retrieve, from a database, spherical regions (of 5 Å radius) that are structurally similar to regions from the unsolved map. In fact, the 5 Å spherical regions centered around Cα's in the backbone model produced by CAPRA are compared to a large database of $\sim$50,000 regions from $\sim$200 maps of proteins (the local structures of the regions are known and cover a very wide range of structural motifs in proteins). The matching local structures are retrieved and assembled together to gradually produce a preliminary model, which can be further refined by post-processing routines. Matching regions are found based on a similarity metric that uses 76 numeric features to locally characterize the spherical regions.

Like many other case-based reasoning systems, LOOKUP needs a large database of cases for wide problem coverage and high quality solutions. But large databases may be very inefficient, especially if the case matching function to determine similarity between two cases is expensive.[38] Given an unsolved spherical query pattern q of electron density, the distance between q and each case c_i in the database can be determined, and the most similar (smallest distance) can be returned as the best match. In TEXTAL$^{\text{TM}}$, the similarity between q and the c_i's is measured by *density correlation*, a metric that involves the computation of the optimal superposition between two patterns. Since the number of possible 3D rotations is very large, the computation of density correlation is too expensive, which we cannot afford to run over the whole database. Thus, we use an approximate, inexpensive, feature-based distance metric to select a small subset of k potential matches, and the density correlation procedure then makes the final ranking. In previous work,[39] we evaluated and compared various feature-based distance metrics for this approach.

It should be noted that a good feature-based match need not be the absolute best one according to the objective metric; it can be the top few matches (based on a tolerance on how high we wish the density correlation value to be to qualify for being a match). Given a query pattern, our aim is to try to get as many good matches (anywhere) in the top k, since the expensive objective will be employed to re-rank the top k matches and identify the truly good ones. In an earlier work,[40] we examined the effectiveness of this filtering scheme and how it depends on the level of tolerance of matching. We also discussed how to choose the value of k, based on a loss function that represents the extent to which the feature-based distance measure approximates the objective metric (density correlation).

3.3. *Post-processing*

There are two main post-processing routines in TEXTAL^{TM}:

- SEQUENCE ALIGNMENT: takes an initial solved model as input (i.e. both the backbone and side chains have been determined), and corrects the identity of amino acids through alignment of the sequence determined by LOOKUP to the known sequence.[30] These corrections are necessary because LOOKUP fits residues that are structurally similar to correct ones, oftentimes erring on the exact identity. Furthermore, noise in the density often deceives LOOKUP in its choices.
- REAL SPACE REFINEMENT: takes a solved structure and the electron density map as inputs, and moves atoms slightly to improve the fit to the density, subject to the preservation of stereo-chemical constraints.[29]

3.4. *Deployment of TEXTAL^{TM}*

The TEXTAL^{TM} project was initiated in 1998 as an inter-disciplinary effort with researchers from computer science and structural biology. TEXTAL^{TM} has been deployed in various ways:

(1) Linux-based distributions, available since September 2004, can be downloaded from the TEXTAL^{TM} website at http://textal.tamu.edu:12321.
(2) Through WebTex, a web-based interface (http://textal.tamu.edu:12321), where registered users can upload their maps that are processed on our servers, and the generated models are automatically sent back in an email. WebTex was launched in June 2002. Currently it is used in 61 research institutions in 17 countries, both from the industry and academia.
(3) As the model determination component of PHENIX[41] (http://www.phenix-online.org), an integrated crystallographic computing environment, based on the Python scripting language. PHENIX was first released in March 2003.

4. Feature Relevance

Automated reasoning and learning systems need information to work effectively — but too much information may cause accuracy and efficiency to degrade. Thus determining what information is relevant is an important endeavor in machine learning. Assessment of relevance in learning systems can be made for various types of information — a training example, a proposition, an inference rule, an attribute (or feature), etc.

Relevance of features is widely studied in learning tasks[42] like pattern classification,[43] instance-based learning[44] and case-based reasoning,[11,12] where patterns or examples have to be compared to detect similarities. Potentially useful features are generally defined by an expert or extracted by automated techniques,[45] and a subset of these features is automatically selected (or weighted), based on the relevance to the task at hand.[46] Irrelevant features tend to mislead pattern matching; the problem is particularly acute in the nearest neighbor method, where irrelevant features can seriously hamper learning.[47] There are several algorithms that have been proposed to address the problem. These approaches can be categorized into two major groups:

(i) *filter* methods try to build classifiers that take into account some properties of the features involved, such as correlations, dependencies and other information;[48] the features are considered independently of the induction algorithm (i.e. the general classifier being built). In fact, the feature selection is done before the induction step; thus irrelevant features are filtered out before induction occurs.[48]

(ii) *wrapper* methods use part of the data to iteratively evaluate the subset of selected features using performance on the induction algorithm for evaluation; this is done by techniques such as cross-validation. In wrapper methods, features are selected by taking the bias of the induction algorithm into account.[42,49]

Feature selection is a specific case of feature weighting i.e. selection is essentially using only two alternative weights, 0 and 1, whereas general weighting assigns degrees of perceived relevance. Blum and Langley[42] suggest that feature selection is most natural when the result is expected to be understood and interpreted by humans, or fed into another algorithm. Feature weighting, on the other hand, are generally purely motivated to enhance the performance of the induction algorithm. But it has also been argued that there may be very little benefit in increasing the number of possible weights beyond two (0 and 1). More fine-grained weighting may, in fact, degrade performance.[50]

A different type of criterion for determining feature or attribute relevance is sensitivity to the context. Different features may be relevant for different instances, making attribute relevance a function of the instance and sensitive to the location in the feature space. This motivates local feature weighting methods.[51–53] Nonetheless, in this work we assume that relevance of features is global, independent of the instance.

Another facet of the feature weighting problem is feature interaction. Sometimes information is shared among attributes, and one attribute is effectively meaningful when considered in conjunction with other attributes.[54,55] Thus, an attribute may appear irrelevant when analyzed independently, but its relevance manifests itself when combined with other attributes.

5. Features in TEXTAL$^{\text{TM}}$

In this section we describe the features that were defined by domain experts to characterize small spherical regions of electron density. One important limitation that was imposed on the design is that the features have to be rotation-invariant, since patterns to be compared can occur in any 3D orientation. This eliminates many candidate features that may contain relevant information — for example Fourier coefficient amplitudes, which are translation-invariant, but not rotation-invariant. The density pattern to be characterized is effectively a 3D grid of electron density values around the protein macromolecule. The relevant information we attempt to capture includes general statistics of the density distribution, moments of inertia, and various geometric measures designed to capture typical shapes of amino acids. Thus, four classes of features have been defined (Table 1):

(i) Statistical features like mean, standard deviation, skewness and kurtosis of electron density distribution for a set of grid points in the spherical region.

(ii) Features based on moments of inertia, which gives the distribution of density in three dimensions. The primary moment lies along the path around which the density is most widely distributed; the secondary and tertiary moments are orthogonal to the primary moment (and to each other) and describe directions in space that have narrower density distributions. The magnitudes of the three moments of inertia are taken as features. The various ratios of eigenvalues

Table 1. Definition of features used to describe spherical electron density patterns in TEXTAL$^{\text{TM}}$. The features are grouped into four classes; each feature has four versions for different radii of the sphere (3, 4, 5 and 6 Å, where 1 Å = 10^{-10} m).

Feature class	Description of feature	Method of computation (ρ_i is the electron density value at the ith of n grid points in a region)
Statistical	Mean	$P = (1/n)\Sigma\rho_i$
	Standard deviation	$[(1/n)\Sigma(\rho_i - \rho)^2]^{1/2}$
	Skewness	$[(1/n)\Sigma(\rho_i - \rho)^3]^{1/3}$
	Kurtosis	$[(1/n)\Sigma(\rho_i - \rho)^4]^{1/4}$
Moments of Inertia	Magnitude of primary moment	Compute inertia matrix, diagonalize and sort eigenvales
	Magnitude of secondary moment	
	Magnitude of tertiary moment	
	Ratio of primary to secondary moment	
	Ratio of primary to tertiary moment	
	Ratio of secondary to tertiary moment	
Symmetry	Distance from center of sphere to center of mass	$\lvert\langle x_c, y_c, z_c\rangle\rvert$, where $x_c = (1/n)\Sigma x_i\rho_i$, $y_c = (1/n)\Sigma y_i\rho_i$, $z_c = (1/n)\Sigma z_i\rho_i$
Shape	Minimum angle between spokes	Find three "spokes" i.e. three distinct vectors with highest density summation, and compute the min, max, median, and sum of angles
	Maximum angle between spokes	
	Median angle between spokes	
	Sum of spoke angles	
	Radial sum of first spoke	
	Radial sum of second spoke	
	Radial sum of third spoke	
	Spoke triangle area	

for the three mutually perpendicular moments of inertia are also defined as features.

(iii) A feature that captures how symmetric or balanced the region is, based on the distance from the center of the sphere to its center of mass.

(iv) Features that capture information about the shape of the pattern: typically an amino acid has three "spokes" emanating from its Cα (one to the side chain and two to the main chain in opposite directions). These spokes are identified, and various features are calculated based on the angles between these spokes. We specifically look at the minimum, median and maximum angle among the spokes. The three spokes are defined as vectors from the center to the surface of the sphere with maximum radial sum, where the radial sum is calculated as the sum of the densities evaluated at points sampled evenly along the spoke. Computation of all possible spoke directions is too expensive; thus a finite number of trial spokes are sampled, and the best triplet of spokes is then computed. Besides the minimum, median and maximum spoke angles, other features include the sum of spoke angles, radial sum for each spoke, and the area of the triangle formed by the endpoints of the three spokes.

Furthermore, for each region, we calculate these features at four different radii (3, 4, 5 and 6 Å); this is necessary since amino acids vary in shape and size, and each feature captures slightly different information for different sizes. Thus, the total number of features that we use is $19 * 4 = 76$.

6. The SLIDER Algorithm

SLIDER[34,56] is a supervised machine learning method to optimize weights such that the accuracy of a distance metric that uses the weights (like weighted Euclidean distance) is improved. It is essentially a *filter* approach that uses the following measure to evaluate how good a set of weights is: given an instance x, we look at how well the given distance metric ranks an instance y truly similar to x, relative to a set of instances known to be different to x. True similarity or difference is determined by an objective, and typically expensive, metric; in this domain, we use *density correlation*, which involves a rotational search to find the best superposition between two regions.

This ranking is averaged over a set of instances. Trying to optimize the ranking is intuitive since similarity is measured by a continuous variable (density correlation). Thus, in searching our database, we are not necessarily looking for one perfect match, since the latter may not exist, and the notion of a match here is fuzzy. Instead, we attempt to select a group of potential matches (by ranking them highly) that will be scrutinized in subsequent steps.

A central idea in SLIDER is that evaluation is done specifically where there is a switch in relative distance between a match and a mismatch to a given instance. These weights are the ones that will influence the accuracy of ranking the most. Thus, by limiting the space of weights to be searched, and identifying only the weights that are more likely to make a difference, the efficiency and effectiveness of learning are largely ensured.

In our empirical analysis, we compare three different weighting schemes: (1) uniform i.e. all features are selected, and weighted equally; (2) binary i.e. feature weights can be either 0 or 1; (3) continuous weights. The latter two schemes are derived from the output of the SLIDER algorithm. We also analyze the sensitivity of the weighting methods to different distance measures. In this work, we look at the Minkowsky family of distance metrics (of order 1, 2 and 3) i.e. Manhattan (L_1), Euclidean (L_2) and L_3.

The SLIDER algorithm was very briefly described in previous work.[56,34] In this section, we provide a more detailed explanation of how weights are tuned by SLIDER. We first consider two-component mixtures (i.e. involving two features, where their weights sum up to one) and then extend it to an arbitrary number of features. The weighted Minkowsky distance of order n between two instances x and y, using two features i and j (with weights w_i and w_j respectively, where $w_i + w_j = 1$) is defined as:

$$D_{i,j}(x,y) = (w_i|x_i - y_i|^n + w_j|x_j - y_j|^n)^{1/n} \tag{1}$$

If $n = 1$, we get the Manhattan distance; if $n = 2$, the metric is called Euclidean, and in general it is known as the Minkowsky distance of order n. We can drop the nth root, since it is a monotonic transformation, and we use distances as a relative measure (their absolute values are not meaningful). Thus $D_{i,j}$ can be re-defined as:

$$\begin{aligned} D_{i,j}(x,y) &= w_i|x_i - y_i|^n + w_j|x_j - y_j|^n \\ &= (1-w)|x_i - y_i|^n + w|x_j - y_j|^n \end{aligned} \tag{2}$$

where w is set to w_j, the weight of feature j. One approach to approximate the optimal weight w is to use a test set to exhaustively evaluate accuracy for various weights defined over a grid, such as $\{0.0, 0.1, 0.2, \ldots, 1.0\}$. Then the induction algorithm would have to be run on the training data to evaluate which weight gives the highest accuracy, as in a wrapper method. This approach is inefficient and is limited by the coarseness of the grid sampling. SLIDER utilizes a more efficient method.

Consider an instance x that has y as its closest neighbor according to feature f_i, and z as its closest neighbor according to feature f_j i.e. the nearest neighbor of x is y when $w = 0$, and it is z when $w = 1$ (w is the weight of feature j), assuming that $y \neq z$. If w "slides" from 0 to 1, then there is a weight w_c at which $D_{i,j}(x,y) = D_{i,j}(x,z)$; this point is called a "crossover". Re-writing $D_{i,j}(x,y) = D_{i,j}(x,z)$, we get:

$$(1-w)|x_i - y_i|^n + w|x_j - y_j|^n = (1-w)|x_i - z_i|^n + w|x_j - z_j|^n \tag{3}$$

Solving for w, and setting it to w_c, we get:

$$w_c = \frac{|x_i - z_i|^n - |x_i - y_i|^n}{|x_j - y_j|^n - |x_i - y_i|^n + |x_i - z_i|^n - |x_j - z_j|^n} \tag{4}$$

In other words, w_c is a weight where there is a net increase (or decrease) in accuracy, depending on which of y and z is truly closer to x. The concept of a

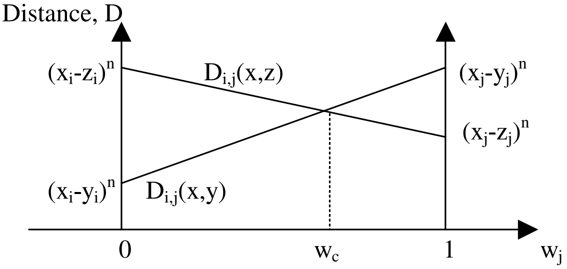


Fig. 3. As the weight of feature j, w_j, slides from 0 to 1, the Minkowsky distance between x and y $[D_{i,j}\ (x, y)]$ changes linearly from lesser to greater than that between x and z $[D_{i,j}\ (x, z)]$. The "crossover" occurs at w_c i.e. there is a change in accuracy of prediction at w_c, depending on whether y or z is truly more similar to x.

crossover point is illustrated in Fig. 3. When there is an increase in accuracy (i.e. the match is closer to x than the mismatch, for all weights above w_c), it is referred to as "positive crossover", and "negative crossover" otherwise. It should be noted that not all three-tuples of instances will have a crossover for a given pair of features; in fact, there will not be a crossover point if for all values of w_j, the distance between x and its match is always larger (or smaller) than the distance between x and its mismatch (i.e. the lines representing distances $D_{i,j}(x, y)$ and $D_{i,j}(x, z)$ in Fig. 3 do not intersect).

Crossover points can also be determined by considering two subsets of features (instead of just two features). Consider two feature subsets A and B, with corresponding Minkowsky distances D_A and D_B respectively. A composite metric, D_{A+B}, can be defined as $D_{A+B}(x, y) = wD_A(x, y) + (1 - w)D_B(x, y)$. As w slides from 0 to 1, it may cause a switch of neighbors, as described earlier. Thus w can be used to determine the new weight vector that increases accuracy, based on crossover points. Currently, SLIDER randomly chooses one feature (singleton set A) and evaluates it against all remaining features (set B). The approach can be extended to compare feature sets of arbitrary size and composition.

SLIDER determines the crossovers for a set T of training examples by sliding over the weight of one feature at a time to determine the "optimum" weight value at which the overall accuracy increases the most. SLIDER uses a greedy approach[57] by iteratively choosing a (random) feature, adjusting its weight based on the above criterion, and stopping when there is no net increase in accuracy. In each iteration of the algorithm, we randomly choose a feature and find all crossover points from the training examples. Then we find the optimum weight w^* (of the chosen feature) that maximizes the difference between the number of positive crossovers and negative crossovers; w^* is computed as follows: For a given set of crossover weights w_i, we define a score $d(w)$ for each crossover weight w as follows:

$$d(w) = \sum_{w_i \leq w} \text{crossover\_type}(w_i) \qquad (5)$$

where crossover_type$(w_i) = 1$ if w_i is a positive crossover weight, and –1 if w_i is a negative crossover weight. Given the set of crossover weights for the randomly chosen feature, we determine the optimal weight w^* of that feature as follows:

$$w^* = \underset{w_i}{\operatorname{argmax}}\ d(w_i) \tag{6}$$

The procedure that we use to evaluate whether overall accuracy has improved by updating the weights is as follows: we define a set S of instances (independent of the training set T to find crossover weights), and for each instance in S we find a match (high density correlation), and a set of mismatches (average or low density correlation). Given a weight vector and an instance, we compute and rank the distances of that instance to the known match and mismatches; the rank of the match relative to the mismatches gives an estimate of the optimality of the weights. Given a weight vector w, a set S of m cases, and for each case, one match and n mismatches, we define the ranking consistency of w, $\mathrm{RC}(w, S)$ as follows:

$$\mathrm{RC}(w, S) = 1/(mn) \sum_{i=1}^{m} [n - \mathrm{rank}\,(i, S)] \tag{7}$$

where $\mathrm{rank}(i, S)$ is the rank of the match of i (relative to all n mismatches of i) in S; note that lower rank implies more similar to the query instance (i.e. the match should ideally have rank = 1).

The pseudo-code for the SLIDER algorithm is given in Fig. 4. After the weights are determined, they can be directly used in the distance metric as continuous

```
Inputs: 1. Sets S and T of spherical regions of density centered on C$\alpha$ atoms.
        2. For each instance in T, one match and one mismatch.
        3. For each instance in S, one match and $n$ mismatches.
        4. $F$ features.
Output: Optimized weight vector $w = \langle w_1, w_2, \ldots, w_F \rangle$
*for* each feature $f_i$
  $w_i \leftarrow 1/F$ //initialize weights of all features uniformly s.t. $\Sigma w_i = 1$
*repeat*
  Select feature $f$ randomly
  Find all crossover points in T // i.e. by sliding $w_f$ from 0 to 1
  Find the "optimum" weight of $f$, $w_f{}^*$
        // The optimum weight maximizes the difference between positive
        // and negative crossovers — Eq. (6)
  *for* all features $i, i \neq f, w_i \leftarrow w_i + (w_f - w_f{}^*) w_i / \sum w_k,\ k \neq f$
                    // Other weights are proportionally adjusted
                    // (according to weight) s.t. all weights add up to 1
  $w_f \leftarrow w_f{}^*$
  Compute Ranking Consistency of $w$, RC($w$, S) using Eq. (7).
*until* $\mathrm{RC}(w, S)$ does not improve *or* number of iterations exceeds a threshold
*return* $w$
```

Fig. 4. The SLIDER algorithm.

weights. Alternatively, the features can be "selected" by converting their corresponding weights to 0 or 1 if the weights are below or above a certain threshold i.e. negligibly small weights (less than 0.01, for instance) are converted to 0, and all other weights are converted to 1. We refer to this weighting scheme as "binary". It should be noted that our objective function in this problem is a continuous metric (density correlation). But this approach can be extended to handle discrete classification problems as well.

7. Results

In this section, we empirically evaluate SLIDER's performance, analyze its impact on TEXTAL^{TM}, and explore various aspects of the feature weighting algorithm. We compare the three different weighting schemes (uniform, continuous and binary), with weights optimized by SLIDER for three different distance metrics (Manhattan, Euclidean and L_3). We closely look at which of the 76 features get how much weight, and analyze the dependence of weighting on the distance metric.

We assess SLIDER's performance on both "ideal" protein maps (i.e. maps that have been artificially generated by back-transforming from their correct structures at 2.8 Å) as well as real, experimental maps. Analysis of the performance of SLIDER on ideal maps is insightful since it reduces the confusion that noise, variance in resolution, and phase error in real maps may introduce. Of course, performance on real maps is what matters ultimately; we later show how SLIDER helps in solving real maps as well.

We use a database of ideal maps for our case-based reasoning approach; a database of ideal maps is more suitable than one of real maps, since it enables better case matching and retrieval in solving real maps. In general, we use ideal maps for the various machine learning tasks in TEXTAL^{TM}: training the neural network to determine Cα positions, case-based reasoning method to "stitch" chains in CAPRA, and tuning feature weights in SLIDER.

Our evaluation of SLIDER on ideal maps involves three sets of data: (1) a set to train SLIDER i.e. to determine crossover points and tune the weights; (2) a test set of query regions to evaluate whether using weights determined by SLIDER improve on retrieval of matching density regions; and (3) a database from where matches are actually retrieved. These three sets were constructed using maps from three independent sets of proteins in PDBSelect,[58] a subset of the PDB[59] database. As mentioned earlier, the maps were artificially generated at 2.8 Å from their correct structures, and the regions were defined as 5Å spheres centered on Cα atoms.

The training set consisted of 200 example regions (drawn randomly from 48 proteins in PDBSelect) that were used to determine the weights. For each example, one match and 200 mismatches were pre-determined by the calculation of density correlation. An instance is defined as a match (mismatch) if its density correlation to the example in question is above (below) 0.7, a threshold above which local patterns of density look similar.

We evaluate the effectiveness of SLIDER in determining a final set of appropriate global weights in the context of the filtering scheme where we essentially try to get as many matches as possible in the top k, and let an expensive measure make the final choice. For that purpose, the following test procedure is used: we chose 100 regions from our test set obtained from 51 proteins in PDBSelect (independent of the proteins used for tuning the weights). For each test region, we exhaustively searched the database of ~30,000 regions (obtained from yet another 137 proteins in PDBselect) to find their true, objective matches (based on density correlation). We then used the weighted feature-based distance metrics to rank all the ~30,000 regions according to similarity, and find out how many *good* matches are found in the top k by the feature-based metric.

We are willing to accept more than just the single best match, since the second, third, and other subsequent matches will often do. So we use a more relaxed notion of *good match*, which is defined as follows: given an example x, a database D and an objective similarity metric *obj*, an example y in D is said to be good match of x if:

$$[obj(x,b) - obj(x,y)]/obj(x,b) < \delta$$

where δ ($\delta \geq 0$) is a tolerance, and $b = \text{argmax}_{d \in D}\ obj(x,d)$, i.e. b is the absolute best match of x in D. We here assume that $obj > 0$, and increases with similarity. This definition accepts multiple hits (in the database) as reasonable matches if they have a similarity within some threshold of the best possible score. In fact, for any given tolerance δ, let the number of good matches be λ. In this domain, the objective similarity metric *obj* is density correlation between two spherical regions, which involves an expensive rotational search for the best possible superposition; density correlation ranges from 0 to 1. Figure 5 shows how the average number of matches λ (over the test set of 100 regions) varies with tolerance δ, using a database of ~30,000 instances. Domain experts found that a tolerance of about 0.02 is reasonable to produce useful models of side chains.

Since SLIDER is greedy and non-deterministic, we run it ten times for each of the three Minkowsky distance metrics. The average weight for each of the 76 features was calculated and proportionally adjusted so that all weights sum up to 1. For all three metrics, between 23 and 31 features (out of 76) were selected i.e. those features with weights greater than 0.01. Moreover, there is strong tendency to choose the same features, and even weigh them similarly. There are 34 features for which all three metrics have yielded zero weight. Table 2 shows all features that were found to be irrelevant (for all three metrics) by SLIDER. These features need not be irrelevant in absolute terms; they might just be useless in the context of other features deemed relevant, especially in situations where some features are redundant. Table 3 shows those features that were found relevant for any of the three metrics — the average weights over ten runs are shown with their standard errors.

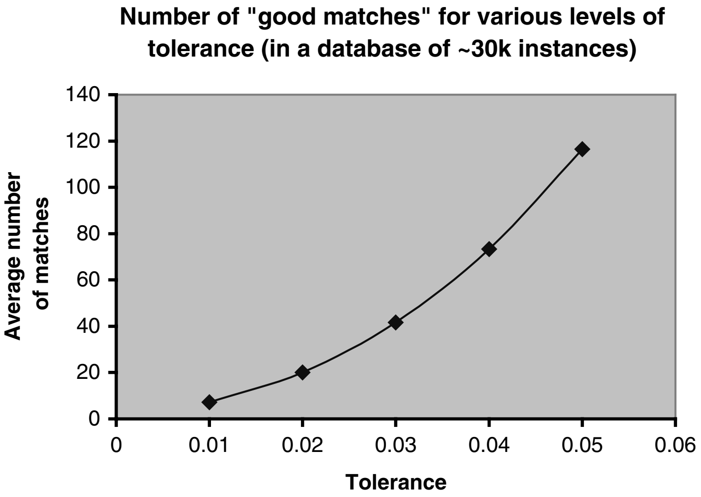


Fig. 5. The number of "good matches" (λ) grows exponentially with tolerance, δ.

Table 2. List of 34 features found irrelevant by SLIDER for all 3 metrics.

Name (and radii in Å) of features with weight = 0
Mean (3, 4)
Standard deviation (4, 5, 6)
Skewness (4, 5, 6)
Kurtosis (5)
Primary moment of inertia (6)
Secondary moment of inertia (3, 4, 5, 6)
Tertiary moment of inertia (4)
Ratio of primary to secondary moment of inertia (5, 6)
Ratio of primary to tertiary moment of inertia (6)
Ratio of secondary to tertiary moment of inertia (5, 6)
Distance to center of mass (6)
Maximum angle between spokes (3, 6)
Minimum angle between spokes (4, 6)
Median angle between spokes (6)
Radial sum of first spoke (3, 4, 5, 6)
Radial sum of second spoke (3)
Radial sum of third spoke (4, 5, 6)

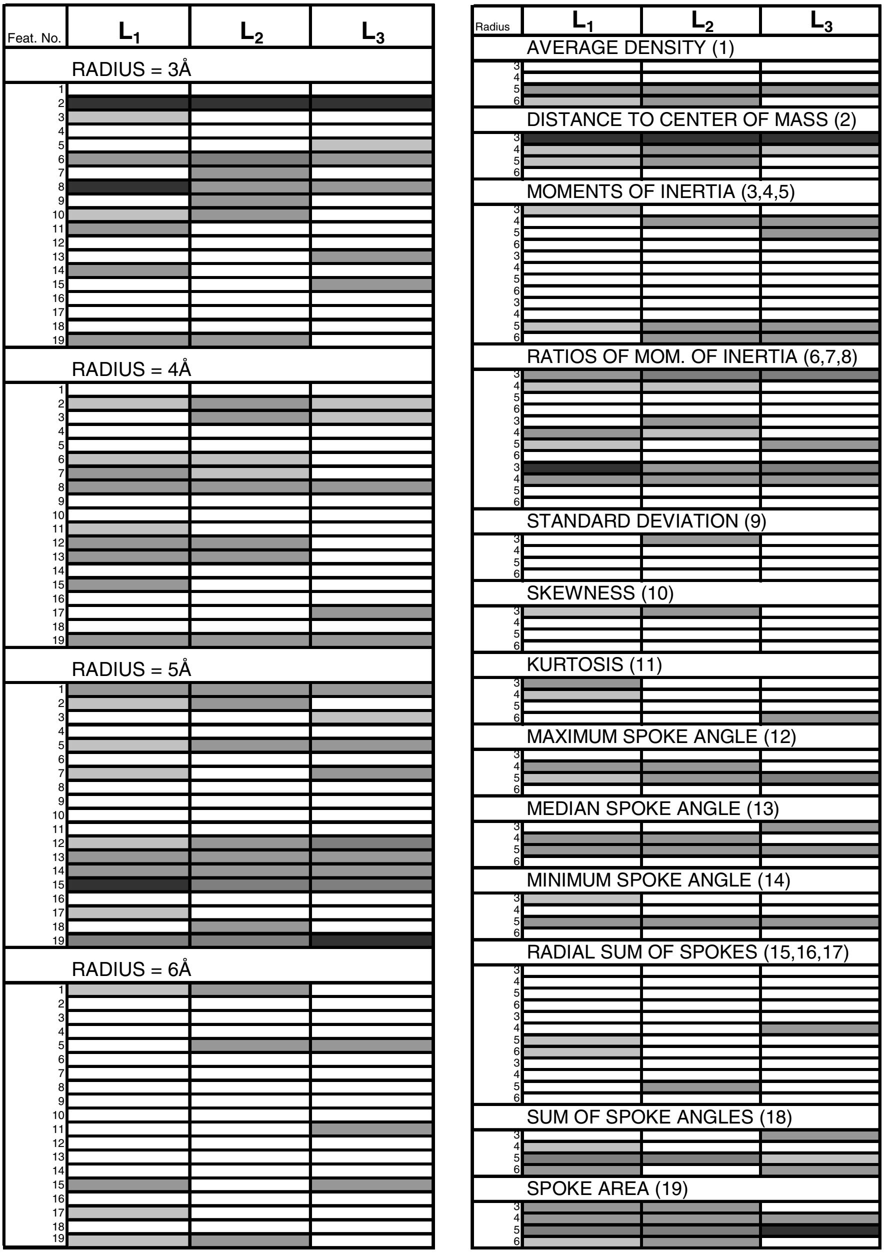
It should be noted that when different SLIDER runs "select" different features, the features selected may actually be quite similar, in two ways:

(i) The features could be closely related e.g. standard deviation, skewness and kurtosis, or the three moments of inertia.
(ii) They could be the same feature (like mean density) but calculated over different radii, especially those radii close to each other.

Table 3. Features found relevant (i.e. non-zero weights) for *any* of the three Minkowsky distance metrics. The weights are averaged over multiple runs. The standard errors are also shown.

Feature name (radius in Å)	Manhattan	Euclidean	L_3
Mean (5)	.046 ± .015	.028 ± .008	.036 ± .012
Mean (6)	.020 ± .011	.026 ± .007	.000 ± .002
Standard deviation (3)	.000 ± .008	.042 ± .007	.000 ± .005
Skewness (3)	.017 ± .007	.030 ± .006	.000 ± .006
Kurtosis (3)	.033 ± .007	.000 ± .005	.000 ± .004
Kurtosis (4)	.016 ± .006	.000 ± .005	.000 ± .008
Kurtosis (6)	.000 ± .004	.000 ± .004	.026 ± .009
Primary moment of inertia (3)	.021 ± .005	.000 ± .006	.000 ± .009
Primary moment of inertia (4)	.000 ± .006	.031 ± .008	.024 ± .008
Primary moment of inertia (5)	.000 ± .000	.000 ± .000	.024 ± .012
Tertiary moment of inertia (3)	.000 ± .000	.000 ± .007	.023 ± .009
Tertiary moment of inertia (5)	.017 ± .009	.029 ± .009	.025 ± .010
Tertiary moment of inertia (6)	.000 ± .006	.023 ± .006	.025 ± .006
Ratio of primary to secondary MOI (3)	.029 ± .006	.056 ± .004	.062 ± .009
Ratio of primary to secondary MOI (4)	.014 ± .005	.023 ± .003	.000 ± .002
Ratio of primary to tertiary MOI (3)	.000 ± .002	.030 ± .008	.000 ± .004
Ratio of primary to tertiary MOI (4)	.026 ± .008	.024 ± .006	.000 ± .005
Ratio of primary to tertiary MOI (5)	.018 ± .007	.000 ± .006	.022 ± .005
Ratio of secondary to tertiary MOI (3)	.087 ± .011	.040 ± .007	.067 ± .015
Ratio of secondary to tertiary MOI (4)	.035 ± .007	.036 ± .007	.029 ± .005
Distance to center of mass (3)	.109 ± .005	.114 ± .006	.116 ± .012
Distance to center of mass (4)	.017 ± .006	.026 ± .007	.024 ± .006
Distance to center of mass (5)	.021 ± .007	.028 ± .005	.000 ± .006
Maximum angle between spokes (4)	.032 ± .007	.031 ± .005	.000 ± .004
Maximum angle between spokes (5)	.022 ± .007	.049 ± .007	.055 ± .013
Median angle between spokes (3)	.000 ± .003	.000 ± .005	.030 ± .005
Median angle between spokes (4)	.026 ± .006	.026 ± .006	.000 ± .003
Median angle between spokes (5)	.034 ± .005	.031 ± .005	.041 ± .007
Minimum angle between spokes (3)	.024 ± .006	.000 ± .004	.000 ± .003
Minimum angle between spokes (5)	.026 ± .006	.044 ± .004	.026 ± .007
Sum of spoke angles (3)	.000 ± .000	.000 ± .003	.026 ± .008
Sum of spoke angles (4)	.018 ± .006	.000 ± .006	.000 ± .006
Sum of spoke angles (5)	.088 ± .019	.054 ± .013	.106 ± .017
Sum of spoke angles (6)	.039 ± .006	.000 ± .006	.044 ± .014
Radial sum of second spoke (4)	.000 ± .000	.000 ± .004	.025 ± .011
Radial sum of second spoke (5)	.020 ± .009	.000 ± .005	.000 ± .001
Radial sum of second spoke (6)	.015 ± .007	.000 ± .002	.000 ± .000
Radial sum of third spoke (5)	.000 ± .006	.025 ± .009	.000 ± .000
Spoke triangle area (3)	.037 ± .010	.034 ± .006	.000 ± .006
Spoke triangle area (4)	.024 ± .005	.034 ± .006	.045 ± .012
Spoke triangle area (5)	.050 ± .016	.062 ± .013	.097 ± .015
Spoke triangle area (6)	.018 ± .007	.025 ± .005	.000 ± .002

Figure 6(a) tries to capture this concordance in returned weights by first sorting the features based on radius and then listing them in a particular order (such that related features are as close together as possible). Figure 6(b) groups the features the other way round i.e. first it lists all features [in the same order as in Fig. 6(a)],



(a) (b)

Fig. 6. The relative weights of 76 features returned by SLIDER for L_1 (Manhattan), L_2 (Euclidean), and L_3 are shown. (a, left): the features are first sorted on radius, and for each radius, 19 features are listed in a specific order. (b, right): the 19 features are first listed in the same order as in Fig. 6(a), and then sorted on radius. Darker shade implies higher weight. The white cells represent features with zero weight.

Table 4. Sum of all feature weights at four different radii; there are 19 features for each radius.

Radius in Å (1 Å = 10^{-10} m)	Sum of weights for three Minkowsky metrics		
	Manhattan	Euclidean	L_3
3	0.36	0.35	0.32
4	0.21	0.23	0.15
5	0.34	0.35	0.43
6	0.09	0.07	0.10

and then sorts them based on radius, in ascending order. The weights have been linearly graded on a five-level scale, where darker shade implies higher weight.

We make the following remarks regarding the feature weights computed by SLIDER:

- The consistency in features selected (and weighted) across the three metrics shows that the algorithm converges. But the risk of local minima still exists; this is partially addressed by the randomized choice of a feature in each iteration. The major cause of local minima is the fact that the weight of only one feature is greedily adjusted at a time.
- Table 4 shows the sum of weights for each radius; we can again observe significant similarity of weights for the three metrics. Furthermore, we can note that the total weights for radii 3 Å and 5 Å are the highest, and comparable to each other, and the total weights for radius 6 Å is significantly lower. The 3D spherical patterns are expected to cover amino acids of various shapes and sizes, which justifies the choice of feature values at different radii. A 3 Å radius sphere centered on a Cα atom is expected to contain significant information about the side chain; but this information may not be adequate to recognize the side chain, especially for large amino acids that may not be totally encapsulated in the 3 Å sphere. But at 6 Å, we face the problem of having noise due to density of neighboring residues or long-range contacts, and hence we expect their relevance to be lower; this trend is captured by our weight optimization algorithm.
- The absolute moments of inertia seem irrelevant individually, but their ratios provide more information related to the shape of the density pattern (e.g. spherical, ellipsoidal, etc.). This exemplifies the feature interaction problem,[54,55] where several features may not appear relevant on an individual basis, but when looked at in combination, they contribute significantly to the description of the pattern.
- The strong similarity of weights across the three Minkowsky metrics is largely expected. Some weights are relevant, irrespective of the underlying metric. For example, the distance between the center of sphere and its center of mass at 3 Å is weighted highly for all three metrics. Nonetheless, there are differences, and interestingly, these differences do capture the sensitivity of "optimum" weights to the metric being used.

Figures 7–9 plot the number of good matches (averaged over the test set) that the three Minkowsky metrics manage to obtain (from the database of ∼30,000 regions) in the top k, for various values of tolerance (at a constant $k = 500$).

Figures 10–12 plot the average number of good matches in the top k for the three metrics, this time varying k (keeping tolerance fixed at 0.02). We can observe that

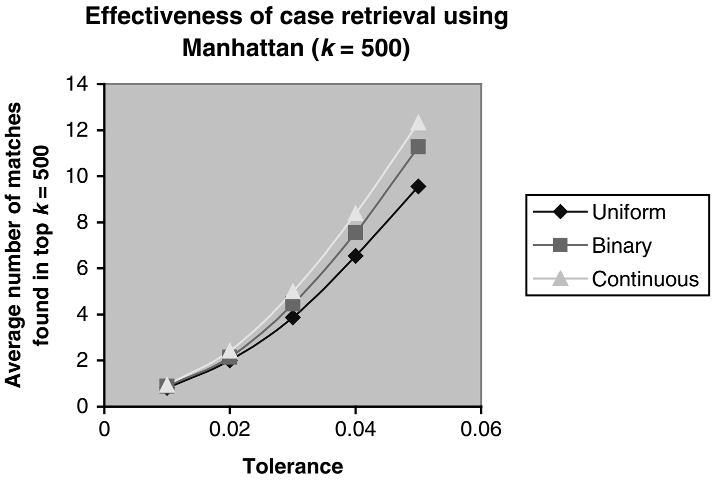


Fig. 7. Weighted Manhattan metrics find more matches than non-weighted Manhattan distance in top $k = 500$ from a database of ∼30,000 regions (for various levels of tolerance). Similar results are obtained for Euclidean (L_2) and L_3.

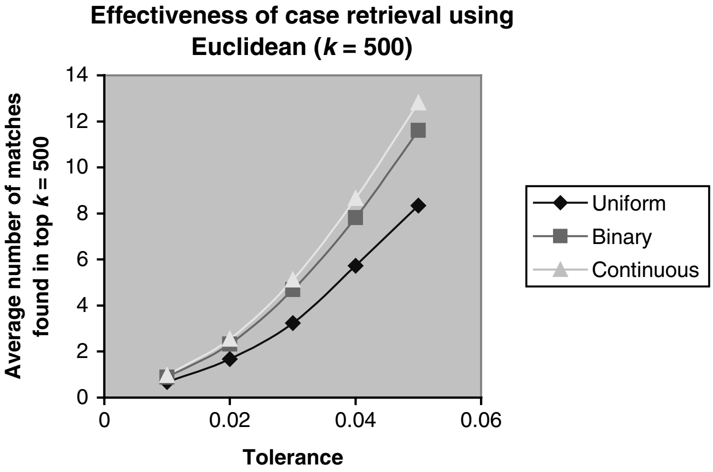


Fig. 8. Continuous weights are more effective than binary weights (using Euclidean distance) in retrieval of matching electron density patterns from a database. Similar results are obtained for Manhattan and L_3.

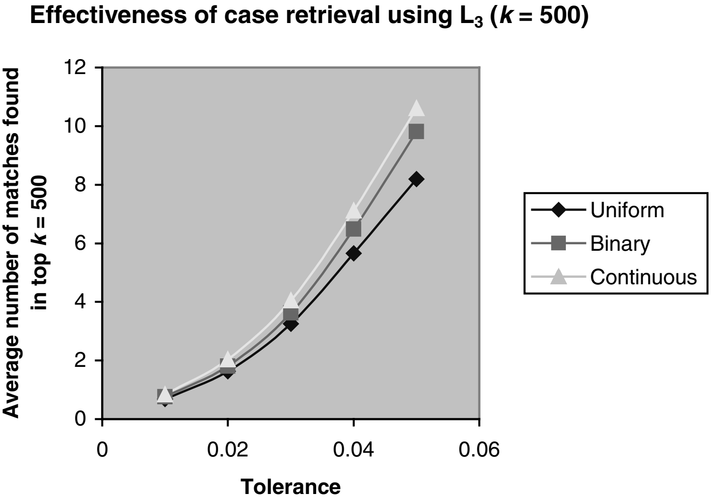


Fig. 9. Weighted L_3 metrics outperform the non-weighted (uniform) L_3 metric.

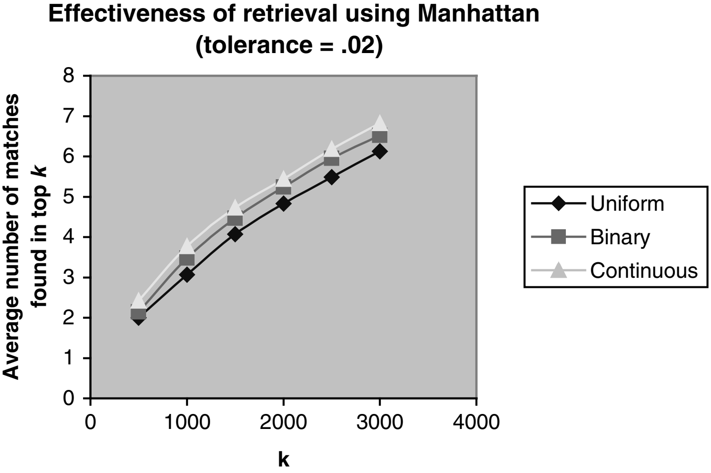


Fig. 10. Weighted Manhattan metrics find more matches than non-weighted Manhattan metric for various values of k from a database of ~30,000 regions (tolerance = .02).

both feature selection (binary weights) and feature weighting (continuous weights) improve over uniform weights (all 76 features equally weighted). Furthermore, continuous weights are better than binary weights for all three metrics.

Figures 7–9 show that as tolerance increases, more matches are found, but at a higher rate for the weighted metrics compared to the non-weighted one. Of course, it can be argued that only one good match needs to be found, which can be achieved

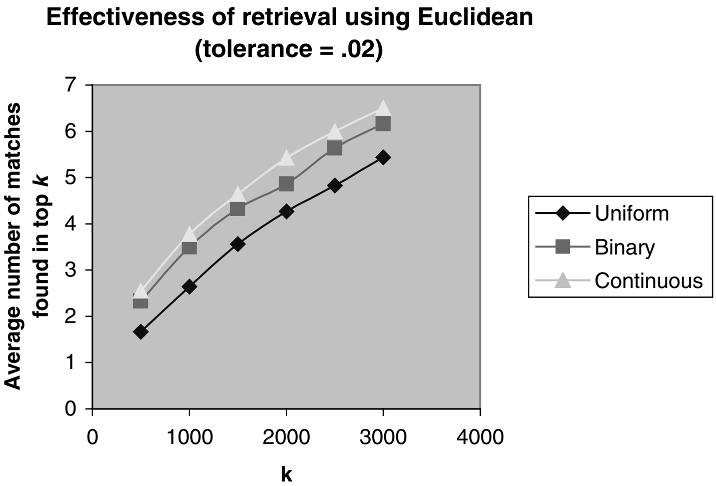


Fig. 11. Continuous weights are more effective than binary weights (using Euclidean distance) in retrieval of matching electron density patterns from a database of ∼30,000 regions for various values of k. Similar results are obtained with Manhattan and L_3.

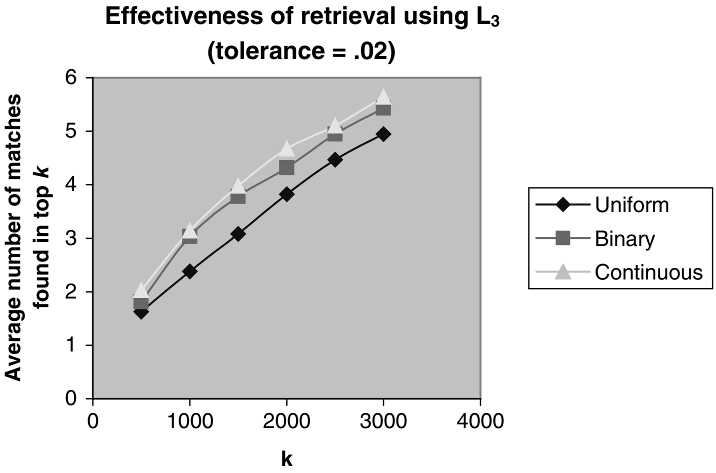


Fig. 12. Weighted L_3 metrics outperform the non-weighted (uniform) L_3 metric.

by setting k sufficiently high. But the improved retrieval allows us to use a lower k, and hence improve efficiency by reducing the database search time.

Figures 10–12 examine how performance scales with k, at constant tolerance. The results again show that using weights determined by SLIDER gives the best performance, since it needs the smallest value of k for the same number of hits (good matches).

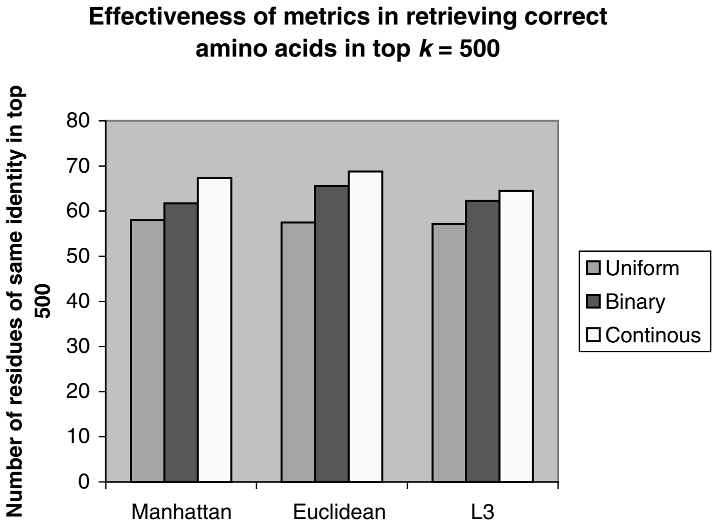


Fig. 13. Weighted metrics retrieve more matches with the same residue identity than non-weighted metrics (for $k = 500$). Similar results are obtained with other values of k.

Figure 13 shows the effectiveness of case retrieval in a different way: how many instances retrieved in the top 500 have same amino acid identity as that of the test region? We can see that the weighted schemes help in getting more side chains of similar identity in the top 500. Similar results are obtained for other values of k (not shown). Selecting as many correct residues as possible in the top k is helpful in the sequence alignment step of TEXTAL$^{\text{TM}}$ — it is often rewarding to look at several top matches (rather than just the first), and see a better alignment with the sequence can be obtained with the second, third, or subsequent matches. Note, however, that even regions with the correct amino acid identity are not necessarily good matches, as they might represent a different side chain conformation. Conversely, sometimes residues of alternative amino acid identity can be good matches, due to structural similarity (e.g. between Valine and Threonine, or Glutamate and Glutamine).

Figure 14 compares the effectiveness of retrieval of the Euclidean metric to the *Mahalabonis distance*,[43] which is a distance metric that takes into account some statistical properties of the data. The Mahalanobis distance is based on the correlations between the feature values; it effectively selects or weights features based on feature covariances. From Fig. 14, it can be observed that, while the Mahalabonis metric outperforms the non-weighted Euclidean measure, both binary and continuous weighting schemes based on SLIDER outperform the Mahalabonis distance.

An important point to note is that, given a distance metric, using continuous weights does not improve pattern matching (as compared to binary weights) if the

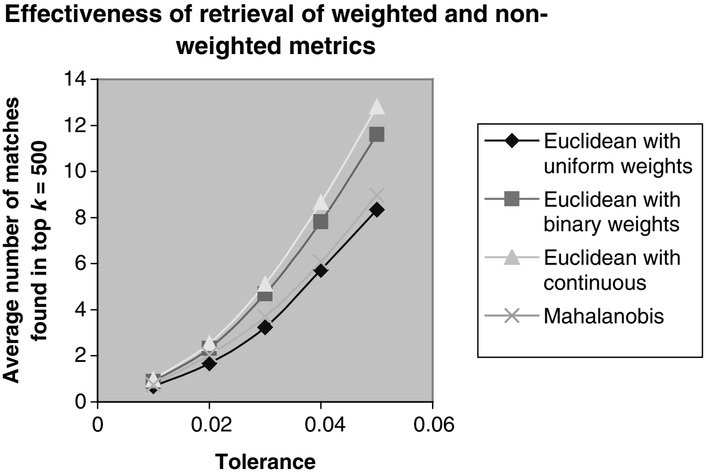


Fig. 14. Mahalanobis distance outperforms the non-weighted (uniform) Euclidean metric in obtaining matches in the top $k = 500$ for various levels of tolerance. But weighted Euclidean distances (using weights determined by SLIDER) are more effective than the Mahalanobis metric in case matching and retrieval.

weights used are those optimized for another metric e.g. the weights determined by SLIDER for Euclidean will not make continuous weights outperform binary weights for Manhattan.

We now evaluate the impact of SLIDER by running TEXTAL$^{\text{TM}}$ (including the sequence alignment and real-space refinement steps) on real maps, using Euclidean distance with uniform weights as well as with weights determined by SLIDER. The distance metric is invoked in the LOOKUP system, which performs the database search. The other components (CAPRA and POST-PROCESSING) do not depend on the distance metric. As mentioned earlier, the database we use in TEXTAL$^{\text{TM}}$ is made up of ideal (back-transformed) maps. The training set that SLIDER uses to determine the weights is also generated from ideal maps. Experimental maps are noisy and have high variance in terms of quality and resolution; thus we achieve better case retrieval and tuning of weights if ideal maps are used.

We ran TEXTAL$^{\text{TM}}$ on four experimentally determined maps: (i) CzrA[60] (chromosome-determined zinc-responsible operon A) is a dimer consisting of 96 amino acids in four α-helices; (ii) IF-5A[61] (translation initiation factor 5a) consists of a pair of β-barrel domains; it has 137 amino acids; (iii) MVK[62] (mevalonate kinase) is a medium-sized protein with 317 amino acids, including both α and β secondary structures; and (iv) PCA[63] (mycolic acid cyclopropane synthase) has 262 amino acids, and is made up of both α-helices and β-sheets.

Table 5 shows the percentage of amino acids that were correctly determined by TEXTAL$^{\text{TM}}$, using the weighted and non-weighted Euclidean measure, and setting k to 100. Given an unsolved query region, the top 100 potential matches are

Table 5. Performance of TEXTAL$^{\text{TM}}$ with and without SLIDER weights.

		% of residues correctly identified by TEXTAL$^{\text{TM}}$	
Protein	No of residues	Euclidean distance with uniform weights	Euclidean distance with weights determined by SLIDER
CzrA	96	98.9	95.6
IF-5A	137	78.1	79.7
MVK	317	25.4	54.7
PCA	262	23.9	57.7

retrieved from a database (of ~50,000 regions) by the Euclidean distance metric, and the final selection is done by choosing the one with the highest density correlation with the query region. We can observe that feature weighting by SLIDER seems to contribute little to the performance of TEXTAL$^{\text{TM}}$ for the first two (smaller) proteins (CzrA and IF-5A). But in the last two cases (MVK and PCA), the percentage of amino acids correctly identified more than doubles when SLIDER weights are used. The wide variation of performance of TEXTAL$^{\text{TM}}$ on different real maps can most likely be attributed to differences in qualities of the maps. The accuracy with which CAPRA places the Cα atoms is also influenced by the quality of the map, and the performance of LOOKUP is sensitive to that of CAPRA. Thus the "interpretability" of maps varies widely, depending on resolution and degree of phase error.

8. Discussion

The SLIDER system has been successfully applied to determine the weights of features for the complex problem of recognizing patterns of electron density in protein crystallography. But the techniques employed are general and potentially useful in other domains, especially those with high-dimensional, noisy data. The salient aspects of our approach are:

- SLIDER is a filter method that avoids searching a large space of possible weight vectors. Instead, the evaluation is performed at weight values that matter i.e. at "crossover" weights, where there is a change in accuracy of matching. Furthermore, locating these weight values can be done efficiently, since it involves solving linear equations applicable to many metrics (like the Euclidean distance). The benefits of restricting the number of weights searched and used for nearest neighbor classification are emphasized by Kohavi *et al.*;[49] they also argue that there are probably no benefits in using weights beyond two possible values (0 and 1) — but the SILDER algorithm does manage to compute finer weight values that improve matching and case retrieval.
- SLIDER was used to optimize weights for three different Minkowsky distance metrics, and proved to be successful in improving pattern matching and retrieval for each of the three metrics, in the context of case-based reasoning and nearest

neighbor strategies to efficiently retrieve matches. The weights as determined by SLIDER were largely similar for the various metrics; nonetheless, the slight differences were significant in capturing the sensitivity of relevance to the distance metric being used. We argue that the relevance of features in describing a pattern is not absolute; it depends on how the features are used to determine similarity, especially since similarity itself is often a fuzzy concept, with multiple ways of determining it.

As future work, there is considerable scope for improvement and investigation. In particular, we are looking at the following:

- SLIDER is currently limited to distance metrics for which crossovers weights can be calculated by solving simple linear equations. This may not be possible for other metrics, like those based on probabilistic and statistical methods.[39,64,65] We are currently investigating approaches where crossover points for such metrics can be efficiently determined (by binary search over the space of weights, for instance).
- SLIDER can be extended to optimize more than one weight at a time, based on the same geometric principles in higher dimensions. It can be shown that each ⟨*instance, match, mismatch*⟩ three-tuple represents a line in a 2D space of two features, with one side representing an improvement in accuracy, and the other side a loss in accuracy. If many such three-tuples are considered, the problem is to find an optimal 2D region (a convex polygon, actually) that represents optimum pairs of weights for these two features. This will make the algorithm less greedy.
- One aspect that necessitates closer scrutiny is the definition of match and mismatch to assess if the updated weights improve accuracy. We use a simple strategy where two patterns are said to match/mismatch if their density correlation in above/below a threshold. We observed that the final set weights returned by SLIDER is sensitive to this threshold. What would be an appropriate threshold, and how can it be determined? Or is there a better way of assessing similarity in this context? Should we use "perfect" matches/mismatches in our training set, or do we need to allow for near-matches/near-mismatches as well, which will enable us capture the nuances in the information that is required to confidently say how different two instances are?
- More generally, we are also working on other strategies to weight features, including analyzing the sensitivity of feature relevance to the context[51–53] and methods based on Singular Value Decomposition (SVD) and Principal Component Analysis (PCA).

Acknowledgements

This work was partially supported by grant P01-GM63210 from the National Institutes of Health.

References

1. Protein Data Bank (PDB) Annual Report 2003, http://www.rscb.org/pdb/annual_report03.pdf.
2. Tsigelny I (ed.), *Protein Structure Determination: Bioinformatic Approach*, International University Line, La Jolla, 2002.
3. Burley SK, Almo SC, Bonanno JB, Capel M, Chance MR, Gaasterland T, Lin D, Sali A, Studier W, Swaminathian S, Structural genomics: beyond the human genome project, *Nature Gen* **232**:151–157, 1999.
4. Richardson JS, Richardson DC, Interpretation of electron density maps, *Method Enzymol* **115**:189–206, 1985.
5. Mowbray SL, Helgstrand C, Sigrell JA, Cameron AD, Jones TA, Errors and reproducibility in electron-density map interpretation, *Acta Cryst* **D55**:1309–1319, 1999.
6. Branden CI, Jones TA, Between objectivity and subjectivity, *Nature* **343**:687–689, 1990.
7. Perrakis A, Morris R, Lamzin V, Automated protein model-building combined with iterative structure refinement, *Nature Struc Biol* **6**:458–463, 1999.
8. Kleywegt GJ, Jones TA, Template convolution to enhance or detect structural features in macromolecular electron density maps, *Acta Cryst* **D53**:179–185, 1997.
9. Levitt DG, A new software routine that automates the fitting of protein X-ray crystallographic electron density maps, *Acta Cryst* **D57**:1013–1019, 2001.
10. Turk D, Towards automatic macromolecular crystal structure determination, in Turk D, Johnson L (eds.), *Methods in Macromolecular Crystallography*, NATO Science Series I, Vol. 325, pp. 148–155, 2001.
11. Kolodner J, *Case-Based Reasoning*, Morgan Kaufmann Publishers, San Mateo, 1993.
12. Leake DB (ed.), *Case-Based Reasoning — Experiences, Lessons and Future Directions*, MIT Press, Cambridge, 1996.
13. Fix E, Hodges J, Discriminatory analysis, nonparametric discrimination: consistency properties, Technical Report 4, USAF School of Aviation Medicine, Randolph Field, Texas, 1951.
14. Okaya Y, Pepinsky R, New formulation and solution of the phase problem in X-ray analysis of non-centric crystals containing anomalous scatterers, *Phys Rev* **103**:1645–1647, 1956.
15. Hendrickson WA, Determination of macromolecular structures from anomalous diffraction of synchrotron radiation, *Science* **254**:51–58, 1991.
16. Blow DM, Crick FHC, The treatment of errors in the isomorphous replacement method, *Acta Cryst* **12**:794–802, 1959.
17. Ke H, Overview of the isomorphous replacement phasing, *Method Enzymol* **276**:448–461, 1997.
18. Feigenbaum EA, Engelmore RS, Johnson CK, A correlation between crystallographic computing and artificial intelligence research, *Acta Cryst* **A33**:13–18, 1997.
19. Terry A, The CRYSALIS project: hierarchical control of production systems, Technical Report HPP-83-19, Stanford University, 1983.
20. Glasgow J, Fortier S, Allen F, Molecular scene analysis: crystal structure determination through imagery, in Hunter L (ed.), *Artificial Intelligence and Molecular Biology*, MIT Press, Cambridge, 1993.
21. Fortier S, Chiverton A, Glasgow J, Leherte L, Critical-point analysis in protein electron density map interpretation, *Method Enzymol* **277**:131–157, 1997.
22. Jones TA, Thirup S, Using known substructures in protein model building and crystallography, *EMBO J* **5**(4):819–822, 1986.

23. Diller DJ, Redinbo MR, Pohl E, Hol WGJ, A database method for automated map interpretation in protein crystallography, *Proteins* **36**:526–541, 1999.
24. Holm L, Sander C, Database algorithm for generating protein backbone and side chain coordinates from a Cα trace, *J Mol Biol* **218**:183–194, 1991.
25. Jones TA, Zou JY, Cowtan SW, Improved methods for building models in electron density maps and the location of errors in these models, *Acta Cryst* **A47**:110–119, 1991.
26. Terwilliger TC, Maximum-likelihood density modification, *Acta Cryst* **D56**:965–972, 2000.
27. Oldfield TJ, A semi-automated map fitting procedure, in Bourne PE, Watenpaugh K (eds.), *Crystallographic Computing 7, Proceedings from the Macromolecular Crystallography Computing School*, Oxford University Press, Corby, UK, 1996.
28. Jones TA, Kjeldgaard M, Electron density map interpretation, *Method Enzymol* **277**:173–208, 1997.
29. Diamond R, A real-space refinement procedure for proteins, *Acta Cryst* **A27**:436–452, 1971.
30. Smith TF, Waterman MS, Identification of common molecular subsequences, *J Mol Biol* **147**:195–197, 1981.
31. Ioerger TR, Sacchettini JC, The TEXTAL system: artificial intelligence techniques for automated protein model-building, in Sweet RM, Carter CW (eds.), *Method Enzymol* **374**:244–270, 2003.
32. Ioerger TR, Sacchettini JC, Automatic modeling of protein backbones in electron-density maps via prediction of Cα coordinates, *Acta Cryst* **D5**:2043–2054, 2002.
33. Holton TR, Christopher JA, Ioerger TR, Sacchettini JC, Determining protein structure from electron density maps using pattern matching, *Acta Cryst* **D46**:722–734, 2000.
34. Gopal K, Pai R, Ioerger TR, Romo TD, Sacchettini JC, TEXTAL$^{\text{TM}}$: artificial intelligence techniques for automated protein structure determination, in *Proceedings the 8th Conference on Innovative Applications of Artificial Intelligence*, Acapulco, Mexico, pp. 93–100, 2003.
35. Brünger AT, XPLOR manual, version 3.1, Yale University, 1992.
36. Greer J, Computer skeletonization and automatic electron density map analysis, *Method Enzymol* **115**:206–224, 1985.
37. Swanson SM, Core tracing: depicting connections between features in electron density, *Acta Cryst* **D50**:695–708, 1994.
38. Smyth B, Cunningham P, The utility problem analyzed: a case-based reasoning perspective, *3rd European Workshop on Case-Based Reasoning*, Lausanne, Switzerland, Advances in Case-Based Reasoning, *Lecture Notes in Computer Science*, pp. 392–399, 1996.
39. Gopal K, Romo TD, Sacchettini JC, Ioerger TR, Evaluation of geometric & probabilistic measures of similarity to retrieve electron density patterns for protein structure determination, *Proc ICAI-04*, Las Vegas, pp. 427–432, 2004.
40. Gopal K, Romo TD, Sacchettini JC, Ioerger TR, Efficient retrieval of electron density patterns for modeling proteins by X-ray crystallography, *Proceedings of the International Conference on Machine Learning & Applications*, Louisville, pp. 380–387, 2004.
41. Adams PD, Gopal K, Grosse-Kunstleve RW, Hung LW, Ioerger TR, McCoy AJ, Moriarty NW, Pai RK, Read RJ, Romo TD, Sacchettini JC, Sauter NK, Storoni LC, Terwilliger TC, Recent developments in the PHENIX software for automated crystallographic structure determination, *J Synchrotron Rad* **11**:53–55, 2004.

42. Blum AL, Langley P, Selection of relevant features and examples in machine learning, *Artif Int* **97**:245–271, 1997.
43. Duda RO, Hart PE, Stork DG, *Pattern Classification*, John Wiley and Sons Inc, New York, 2001.
44. Aha DW, A study of instance-based algorithms for supervised learning tasks: mathematical, empirical, and psychological observations, Ph. D. Thesis, University of California, Irvine, 1990.
45. Liu H, Motoda H (eds.), *Feature Extraction, Construction, and Selection: A Data Mining Perspective*, Kluwer, Boston, 1998.
46. Aha DW, Feature weighting for lazy learning algorithms, in Liu H, Motoda H (eds.), *Feature Extraction, Construction and Selection: A Data Mining Perspective*, Kluwer, Boston, 1998.
47. Langley P, Iba W, Average-case analysis of a nearest neighbor algorithm, *Proc IJCAI-93*, Chambery, France, pp. 889–894, 1993.
48. Kira K, Rendell LA, A practical approach to feature selection, *Proceedings of the 9th International Conference on Machine Learning*, pp. 249–256, 1992.
49. John G, Kohavi R, Pfleger K, Irrelevant features and the subset selection problem, *Proceedings of the 11th International Conference on Machine Learning*, pp. 121–129, 1994.
50. Kohavi R, Langley P, Yun Y, The utility of feature weighting in nearest-neighbor algorithms, *Proceedings of the European Conference on Machine Learning*, 1997.
51. Domingos P, Context-sensitive feature selection for lazy learners, *Artif Int Rev* **11**:227–253, 1997.
52. Howe N, Cardie C, Examining locally varying weights for nearest neighbor algorithms, *Lect Notes Artif Int* 455–466, 1997.
53. Greiner R, Grove AJ, Kogan A, Knowing what doesn't matter: exploiting the omission of irrelevant data, *Artif Int* **97**:345–380, 1997.
54. Jakulin A, Bratko I, Testing the significance of attribute interactions, *Proceedings of the ICML-04*, Banff, 2004.
55. Ioerger TR, Detecting feature interactions from accuracies of random feature subsets, *Proceedings of the 16th National Conference on Artificial Intelligence*, pp. 49–54, 1999.
56. Holton TR, Christopher JA, Ioerger TR, Sacchettini JC, Determining protein structure from electron density maps using pattern matching, *Acta Cryst* **D46**:722–734, 2000.
57. Russel S, Norvig P, *Artificial Intelligence: A Modern Approach*, Prentice Hall, New Jersey, 1995.
58. Hobohm U, Scharf M, Schneider R, Sander C, Selection of a representative set of structures from the Brookhaven Protein Data Bank, *Protein Sci* **1**:409–417, 1992.
59. Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H, Shindyalov IN, Bourne PE, The Protein Data Bank, *Nucleic Acids Res* **28**:235–242, 1992.
60. Eicken C, Pennella MA, Chen X, Koshlap KM, VanZile ML, Sacchettini JC, Giedroc DP, A metal-ligand-mediated intersubunit allosteric switch in related SmtB/ArsR zinc sensor proteins, *J Mol Biol* **333**(4):683–695, 2003.
61. Peat TS, Newman J, Waldo GS, Berendzen J, Terwilliger TC, Structure of translation initiation factor 5A from Pyrobaculum aerophilum at 1.75 Å resolution, *Structure* **6**:1207–1215, 1998.
62. Yang D, Shipman LW, Roessner CA, Scott IA, Sacchettini JC, Structure of the *Methanococcus jannaschii* mevalonate kinase, a member of the GHMP kinase superfamily, *J Biol Chem* **277**(11):9462–9467, 2002.

63. Huang CC, Smith CV, Glickman MS, Jacobs WR Jr, Sacchettini JC, Crystal structures of mycolic acid cyclopropane synthases from Mycobacterium tuberculosis, *J Biol Chem* **277**:11559–11569, 2002.
64. Aksoy S, Haralick RM, Probabilistic vs. geometric similarity measures for image retrieval, *Proceedings of the Computer Vision and Pattern Recognition (CPRV)*, pp. 112–128, 2001.
65. Kontkanen P, Myllymaki P, Silander T, Tirri H, A Bayesian approach for retrieving relevant cases, in Smith P (ed.), *Artificial Intelligence Applications*, *Proceedings of the EXPERSYS-97*, Sunderland, UK, pp. 67–72, 1997.

Kreshna Gopal is a Ph.D. candidate under the supervision of Dr. Thomas R. Ioerger in the Department of Computer Science at Texas A&M University. Mr. Gopal received a B.Tech. in Computer Science & Engineering from the Indian Institute of Technology, Kanpur in 1990. From 1991 to 1997, he worked as a software engineer and a lecturer in computer science in Mauritius. In 1997, he enrolled in the graduate computer science program at Texas A&M University as a Fulbright fellow, and obtained an M.S. in 2000. His research interests are in the areas of artificial intelligence, machine learning, planning and bioinformatics.

Tod D. Romo is a Research Scientist at the Institute for Biosciences and Technology in the Center for Structural Biology and W. M. Keck Center for Bioinformatics at Houston, Texas. He received a B.S. in Biology and a B.S. in Mathematics from Trinity University in 1991 and a Ph.D. in Biochemistry and Cell Biology from Rice University in 1999, where he was a W. M. Keck Center for Computational Biology Predoctoral Fellow. Dr. Romo's research interests include computational biology and bioinformatics for structural biology, as well as scientific visualization and virtual reality.

James C. Sacchettini is a Professor of Biochemistry and Biophysics and of Chemistry, the Wolfe–Welch Chair in Science, the Director of the Center for Structural Biology, all at Texas A&M University. He received a B.A. from St. Louis University in 1980, Ph.D. from Washington University, St. Louis in 1987. He was a Postdoctoral Fellow at Washington University, St. Louis from 1987 to 1989. He joined the faculty at Texas A&M University in 1996. Dr. Sacchettini is currently involved in research on tuberculosis, malaria, lyme disease and crystallographic protein modeling.

Thomas R. Ioerger is an Associate Professor in the Department of Computer Science at Texas A&M University. He received a B.S. from Penn State University in the area of Molecular and Cellular Biology in 1989, an M.S. and a Ph.D. in Computer Science from the University of Illinois in 1992 and 1996 respectively. Dr. Ioerger's research interests are in the areas of artificial intelligence, multi-agent systems, machine learning and bioinformatics.