Probabilistic Ensembles for Improved Inference in Protein-Structure Determination

Ameet Soni
Dept. of Computer Sciences
Dept. of Biostatistics and Medical Informatics
University of Wisconsin–Madison
soni@cs.wisc.edu

Jude Shavlik
Dept. of Computer Sciences
Dept. of Biostatistics and Medical Informatics
University of Wisconsin–Madison
shavlik@cs.wisc.edu

ABSTRACT

Protein X-ray crystallography – the most popular method for determining protein structures – remains a laborious process requiring a great deal of manual crystallographer effort to interpret low-quality protein images. Automating this process is critical in creating a high-throughput protein-structure determination pipeline. Previously, our group developed ACMI, a probabilistic framework for producing protein-structure models from electron-density maps produced via X-ray crystallography. ACMI uses a Markov Random Field to model the 3D location of each non-hydrogen atom in a protein. Calculating the best structure in this model is intractable, so ACMI uses approximate inference methods to estimate the optimal structure. While previous results have shown ACMI to be the state-of-the-art method on this task, its approximate inference algorithm remains computationally expensive and susceptible to errors. In this work, we develop Probabilistic Ensembles in ACMI (PEA), a framework for leveraging multiple, independent runs of approximate inference to produce estimates of protein structures. Our results show statistically significant improvements in the accuracy of inference resulting in more complete and accurate protein structures. In addition, PEA provides a general framework for advanced approximate inference methods in complex problem domains.

Categories and Subject Descriptors

G.3 [**Probability and Statistics**]: Probabilistic algorithms; J.3 [**Life and Medical Sciences**]: [Biology and genetics]

General Terms

Algorithms, Experimentation, Performance

Keywords

ensembles, statistical inference, protein-structure determination, computational biology

1. INTRODUCTION

Over the past decade, the field of machine learning has seen a large increase in the use and study of probabilistic graphical models due to their ability to provide a compact representation of complex, multidimensional problems [13]. Graphical models have applications in many areas, including natural language processing, computer vision, gene regulatory network modeling, and medical diagnosis. Recently, the complexity of problems posed in many domains, such as statistical relational learning, has stressed the ability of algorithms to reason in graphical models. New techniques for *inference* are essential to meet the demands of these problems in an efficient and accurate manner.

One such application is our group's work on ACMI (Automated Crystallographic Map Interpretation), a three-phase, probabilistic method for determining protein structures from electron-density maps [7, 8, 9, 20]. The task of determining protein structures has been a central one to the biological community, with recent years seeing significant investments in structural-genomic initiatives[21]. X-ray crystallography, a molecular-imaging technique, is at the core of many of these initiatives as it is the most popular method for determining protein structures. In creating a high-throughput crystallography pipeline, however, the final step of constructing an all-atom protein model from an *electron-density map* – a three-dimensional image of a molecule produced as an intermediate product of X-ray crystallography – remains a major bottleneck in need of automation. In difficult cases, this can take months of manual effort by a crystallographer.

Previous results show that ACMI outperforms other automated density-map interpretation methods on difficult protein structures, producing complete and accurate protein structures where other methods fail [7]. ACMI models the probability of all possible configurations of a protein structure using a graphical model known as pairwise Markov random field (MRF) [11]. ACMI's MRF combines visual features derived from the electron-density map with biochemical constraints in order to effectively identify the most probable locations for each amino acid in the electron-density map. Unfortunately, exact inference (i.e., finding the best protein structure model) is computationally infeasible due to the complexity of the MRF. ACMI, instead, employs *approximate* inference techniques to produce a probability distribution for each amino acid's location in the density map. While ACMI has shown promising results, its inference is not guaranteed to arrive at the correct solution and, more importantly, is computationally expensive. This limits ACMI's ability to produce sufficiently accurate solutions in many cases.

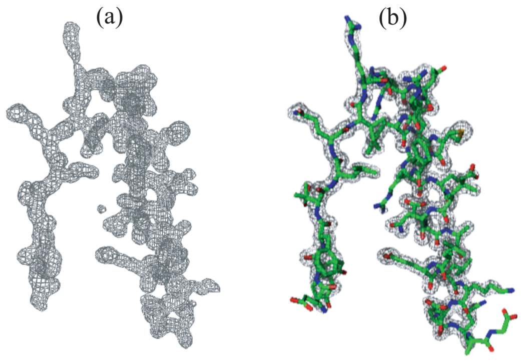


Figure 1: The last step in the protein X-ray crystallography pipeline takes a) the electron-density map (a 3D image) of the protein and finds b) the most likely protein structure that explains the map. Here, the electron density is contoured and the chemical structure of the protein is designated with a stick model showing all of the non-hydrogen atoms.

To overcome these limitations we propose Probabilistic Ensembles in Acmi (Pea), a general framework for performing approximate inference in complex domains. Pea borrows the concept of ensemble learning methods from the supervised machine learning literature [2, 6]. While traditional methods build a single model to estimate the underlying true model to a problem, ensembles leverage a collection of estimates to produce a more accurate solution. We extend this idea to our task by performing multiple, independent runs of approximate inference in Acmi to produce multiple probability estimates of the protein's locations. Our results show Pea dramatically outperforms Acmi in both the quality of inference and accuracy of protein structures produced.

Section 2 provides background on protein structures and automated interpretion of low-quality electron-density maps. Section 3 details Acmi, our existing framework for determining protein structures, including a discussion of approximate inference. Section 4 describes our proposed technique, Probabilistic Ensembles in Acmi (Pea). Lastly, in Section 5, we compare Pea to our previous work using a set of difficult electron-density maps.

2. BACKGROUND

2.1 Protein X-Ray Crystallography

Amino acids form the building blocks of proteins, linking end-to-end to form the linear protein sequence. This sequence folds to form the three-dimensional protein structure. The main chain of atoms linking amino acids is known as the *backbone* of the protein and the variably sized groups hanging off of the backbone are called *side chains*. Amino acids come in twenty different varieties, with each amino-acid type having a unique side-chain molecule and the same set of backbone atoms. All side chains connect to the backbone via the Cα atom – the central atom in an amino acid.

X-ray crystallography is the most popular wet-lab technique for determining protein structures, producing $\sim$88% of protein structures in the Protein Data Bank [19]. The final step in the X-ray crystallography process is taking an electron-density map – a fuzzy, three-dimensional image of a protein – and determining (or *interpreting*) the underlying protein molecular model that produced the image. Figure 1 shows a sample density map and the resulting interpretation. Figure 1a is a contoured electron-density map, similar to what a crystallographer would see at the beginning of interpretation. In b) we see the resulting protein structure with all non-hydrogen atoms in a stick representation. The crystallographer's task is: given a protein's amino-acid sequence and an electron-density map of the protein, produce the underlying protein structure.

Several factors make determining the protein structure a difficult and time-consuming process, mainly by affecting the quality of the electron-density map. The most significant factor is crystallographic resolution which describes the highest spatial frequency terms used to assemble the electron-density map. Resolution is measured in angstroms (Å), with higher values indicating poorer-quality maps with less detail. Another factor making automation difficult is the *phase problem*; crystallographer's can only estimate the phases needed to calculate the electron-density map, reducing the interpretability of the image. Lastly, imperfections in the crystal structure and the stochastic nature of protein structure can create areas of distortion or smeared density in the image that contain very little or unreliable features.

2.2 Automated Approaches

The most popular method for automated density-map interpretation is ARP/wARP [15, 18], an algorithm which iteratively performs two steps. First, it fits atomic structure to a density map and second, it refines (or improves) the quality of the map so more structure can be identified on the next iteration. ARP/wARP efficiently finds solutions in maps with 2.7 Å resolution or better. Another approach, Textal [12], uses rotation-invariant features derived from regions of the density map to train two pattern-recognition algorithms: Capra, a neural-network classifier for identifying Cα locations and Lookup for matching side-chain templates to a region of the density map around a Cα. A third algorithm, Resolve [22], uses a top-down procedure in which secondary-structure elements are located in the map as starting points for building large segments of the protein structure. Using a large library of protein fragments, Resolve first extends these secondary-structure elements before attempting to connect them with loop elements. A recent method, Buccaneer [4], takes a similar approach as Textal but utilizes orientation-based features. Buccaneer currently only performs a backbone trace, and thus does not provide a complete protein model. Textal, Resolve, and Buccaneer have successfully interpreted density maps up to 3.2 Å in quality.

3. AUTOMATED CRYSTALLOGRAPHIC MAP INTERPRETATION

In previous work, our group developed Acmi (Automated Crystallographic Map Interpretation) [7, 8, 9, 20], a probabilistic method for determining protein structures from low-quality electron-density maps ($\sim$3 to 4 Å resolution). This section provides an overview of our existing framework.

3.1 ACMI Overview

Figure 2 provides an overview of Acmi and its three-phase process. At the heart of Acmi is a probabilistic model known

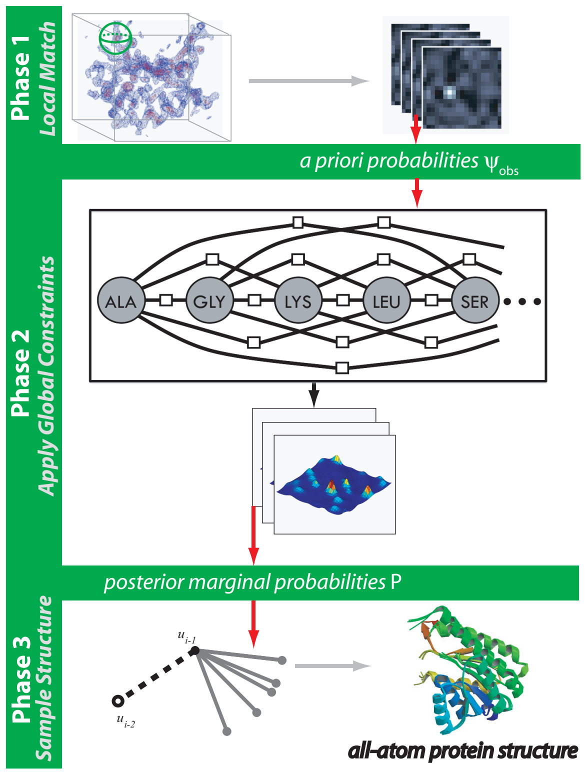


Figure 2: The three-phase ACMI pipeline. Given an electron-density map and amino-acid sequence, Phase 1 performs a local-match search independently for each amino acid. Phase 2 combines these local-search results with global constraints to create posterior probabilities of each amino acid's location. Finally, Phase 3 uses these marginals to sample physically feasible, all-atom protein structures. The upper box in Phase 2 shows a portion of a Markov Random Field for an example protein sequence.

as a *pairwise Markov Random Field* (MRF) [11]. An MRF is an undirected graphical model that defines a probability distribution on a graph. MRF's compactly represent a full-joint probability by factorizing the complex, large joint-probability into smaller factors. *Vertices* (or nodes) in an MRF are associated with random variables, and *edges* enforce pairwise constraints on those variables. In our task, ACMI seeks to probabilistically represent all possible structures of a protein in a compact manner. Given an electron-density map and the linear amino-acid sequence of a protein, ACMI constructs a graph where each vertex describes the location, $\vec{u_i}$, of the Cα atom for the amino acid at position i in the sequence. Edges exist between every two amino acids in the sequence and model the interactions between the pair of connected amino acids. A sample MRF is show in the upper Phase 2 box in Figure 2.

Formally, ACMI's pairwise Markov Random Field model $G = (V, E)$ consists of vertices $i \in V$ connected by undirected edges $(i, j) \in E$. We define the full-joint probability of all amino-acid locations, $\mathbf{U}$, as

$$P(\mathbf{U}|\mathbf{M}) = \prod_{i \in V} \psi_i(\vec{u_i}|\mathbf{M}) \times \prod_{(i,j) \in E} \psi_{i,j}(\vec{u_i}, \vec{u_j}). \quad (1)$$

The first term, $\psi_i(\vec{u_i}|\mathbf{M})$, is associated with vertex i and is known as the *observation* potential function. It can be thought of as prior probability on the location of an amino acid given the map, $\mathbf{M}$, and ignoring all other amino acids in the protein. ACMI calculates this function in Phase 1. Briefly, ACMI looks up instances of amino acid i in a database of previously solved structures[1] to serve as templates (i.e., potential configurations) for amino acid i. Each returned instance is compared to the density map at each grid point, with the maximum correlation coefficient across all rotations being stored [9].

The second term, $\psi_{i,j}(\vec{u_i}, \vec{u_j})$, is associated with edges. This function represents one of two *conformation* potentials which define pairwise chemical constraints on the protein structure. Edges between neighboring amino acids in the linear sequence (i.e., $|i-j| = 1$) are represented by the *adjacency potential*, which encodes the constraint that adjacent amino acids must maintain an approximate 3.8 Å spacing as well as proper angles[2]. Edges between non-neighboring amino acids (i.e., $|i-j| > 1$) contain an *occupancy potential*, reflecting the constraint that no two amino acids can occupy the same space.

Given this MRF, we construct a three-phase pipeline (Figure 2) to calculate the most probable protein structure for a given protein sequence and electron-density map. As mentioned, Phase 1 estimates the observation potential, or the probable location of each amino acid in the density map independent of information about other amino acids. Phase 2 then takes these results and combines them with chemical constraints using the MRF outlined above. Inference on the MRF distributes the evidence and produces a posterior marginal probability of each amino acid's location in the electron-density map. Phase 3 uses these probabilities to sample physically-feasible, all-atom (i.e., side chain and backbone atoms) protein structures.

ACMI owes much of its success to two distinguishing properties. First, ACMI simultaneously ties local feature detection in the density map with global biochemical constraints to infer possible locations of amino acids. Second, while other techniques represent an amino acid with a small set possible locations, ACMI represents each amino acid's location probabilistically; that is, each possible location in the electron-density map is considered with certain weight. This allows the algorithm to overcome poor, early decisions while also allowing weaker evidence to persist and possibly be utilized in later stages.

3.2 Inference in ACMI

The model in Equation 1 represents the full-joint probability distribution over all possible locations for each amino acid in the target protein. Calculating this probability exactly, however, is intractable due to the cyclical nature and large size of ACMI's graph. ACMI, instead, employs loopy belief propagation (BP) [16, 17], a fast approximate-inference algorithm, to calculate an approximate marginal probability distribution for the location of each amino acid's Cα atom. This task takes place as Phase 2 in the ACMI pipeline.

Belief propagation calculates marginal probabilities by utilizing an iterative, local message-passing scheme to prop-

[1]ACMI uses a non-redundant subset of the Protein Data Bank (PDB) [24].

[2]Bond and angle distributions are obtained from protein structures in the Protein Data Bank (PDB).

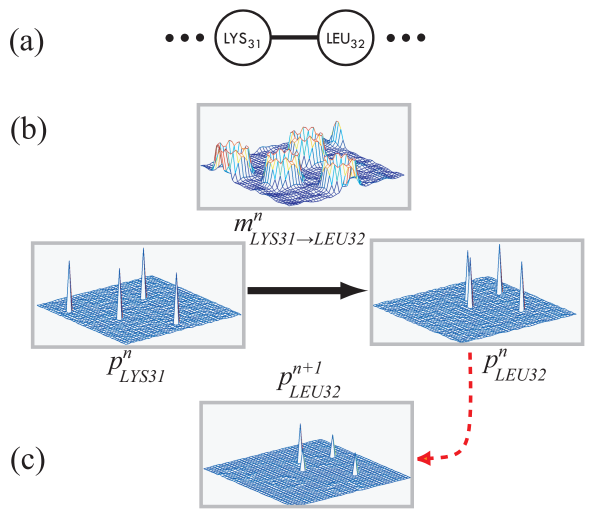


Figure 3: A sample message being sent in ACMI's belief-propagation algorithm. In a) we show the portion of interest in the protein's MRF. In b) we show the current state of beliefs and the calculated message to be sent from lysine (amino acid 31 in the sequence) to leucine (32). Finally, c) shows leucine's updated belief after receiving the message, which has boosted the confidence in one of the original four peaks. For simplicity, these distributions represent probabilities on a 2-dimensional plane.

agate information across a graphical model [17]. Here, a *marginal* probability represents the posterior probability of a single amino acid's location, incorporating all available information. While converging to the exact solution is not guaranteed in cyclical graphs such as ACMI, belief propagation tends to produce good approximations in practice [16]. A message from amino acid i to amino acid j states, "Based on my current belief in my location, you should be here (with weight)."

Formally, BP, at iteration n for vertex i, computes an estimate of $\hat{p}_i^n(\vec{u_i})$, amino acid i's belief:

$$\hat{p}_i^n(\vec{u_i}) = \psi_i(\vec{u_i}|\mathbf{M}) \times \prod_{k \in \Gamma(i)} m^n_{k \to i}(\vec{u_i}) \qquad (2)$$

where $\Gamma(i)$ is the set of vertices connected to vertex i. Messages from amino acid i to amino acid j are calculated by a convolution of the edge potential $\psi_{i,j}(\vec{u_i}, \vec{u_j})$ (i.e., adjacency potential or occupancy potential) with amino acid i's belief

$$m^n_{i \to j}(\vec{u_j}) = \int_{\text{EDM}} \psi_{i,j}(\vec{u_i}, \vec{u_j}) \times \frac{\hat{p}_i^n(\vec{u_i})}{m^{n-1}_{j \to i}(\vec{u_i})} \, d\vec{u_i}. \qquad (3)$$

The convolution occurs over the entire distribution, denoted EDM (for Electron-Density Map). The denominator inside the integral removes the influence of the previous message sent across the edge.

A sample iteration of message passing is shown in Figure 3. This process continues iteratively, one amino acid at a time, until all amino acids converge to a stable solution, or some stopping criteria is met. The belief after the final iteration becomes the posterior probability of each amino acid's location and is fed as input to Phase 3.

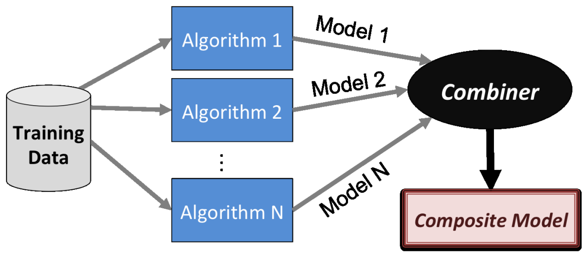


Figure 4: Ensemble learning methods for supervised learning. Multiple algorithms construct models based on the training data. The resulting models are combined to produce a more comprehensive, composite model.

4. METHODS

As mentioned in Section 3.1, ACMI's Phase 2 utilizes an approximate inference technique known as loopy belief propagation (BP) to calculate the location of each amino acid in the protein sequence. Empirically, ACMI's Phase 2 rarely converges to a solution and while ACMI performs well on difficult proteins, results indicate that there are shortcomings in the inference process [7, 20]. This section discusses the major contribution of this paper: a new technique for improving ACMI's performance using a statistical ensemble of approximate inference solutions. Section 4.1 introduces the concept of ensemble learning methods from the supervised machine learning literature and related ideas in probabilistic inference. Section 4.2 presents our technique, Probabilistic Ensembles in ACMI (PEA), in a modified ACMI framework. Lastly, Section 4.3 presents the methodology for experiments we used to evaluate our proposed technique.

4.1 Ensemble Learning Overview

Ensemble learning methods come primarily from the supervised machine learning community. The goal of supervised learning is to develop a model (or classifier) with high predictive performance on future instances of a problem. Traditional learning methods involve a search through a hypothesis space that returns a single-best model, $\hat{f}(x)$, to estimate the underlying (but unknown) true function, $f(x)$. Ensemble learning methods, illustrated in Figure 4, aim to develop a collection of models, $\hat{f}^1(x), \hat{f}^2(x), ..., \hat{f}^N(x)$, that, in aggregate, produce a classifier with better performance than any single constituent model. Empirical evaluations of ensemble learning methods (or *ensembles*) show that such methods outperform the best individual constituent models under two conditions [2, 6, 14]. First, the ensemble must be *diverse*. A lack of diversity means each model will produce the same answer to a given instance and thus the collective performance will mirror individual performance. Second, the individual members must be at least *weakly* accurate; that is, performing better than random guessing.

There are two primary design choices in developing an ensemble learning method. First, the learner must generate models that meet the two criteria above i.e., the learner must generate diverse, accurate models. The machine learning literature contains a variety of techniques for accomplishing this, including the popular methods *bagging* and *boosting*. Second, the learner must aggregate the decisions (or predictions) of each model. This is often accomplished with

Algorithm 1: Probabilistic Ensembles in ACMI (PEA)

```
input  : amino-acid sequence $Seq$ of length $N$,
         electron-density map of protein $\mathbf{M}$,
         number of ensemble components $C$,
         inference protocol for each component $protocol_c$
output: Protein structure $\mathbf{U} = (u_1, u_2, \ldots, u_N)$ where
         $u_i$ is the coordinates of amino acid $i$
// Calculate vector of observation potentials, $\Psi(U)$
for $i = 1 \ldots N$ do
    $\psi_i(\vec{u_i}) \leftarrow$ Phase1($Seq, \mathbf{M}$)
end
// Generate ensemble of posterior probabilities, $\mathcal{P}$
for $c = 1 \ldots C$ do
    $\hat{P}_c \leftarrow$ Phase2($Seq, \Psi(U), protocol_c$)
end
// Sample all-atom protein structure
$\mathbf{U} \leftarrow$ Phase3($Seq, \mathcal{P}$)
```

majority voting, where each model gets a weighted or unweighted "vote" on the answer to a query instance.

While most work on ensembles is on supervised machine learning problems, we are interested in structured-prediction problems such as ACMI. Rather than classification, our task involves performing inference to estimate the probability densities of several unknown and connected variables (e.g., amino-acid locations). Weiss et al. [25] proposed Structural Ensemble Cascades (SEC), an iterative, hierarchical method for structured prediction problems. Their model learns a sequence of coarse-to-fine models to filter possible output locations in a vision task (e.g., tracking human pose across frames of video). Ensembles are used in inference, where an intractable graph is converted to a set of tractable (i.e., tree-structured) graphs. Wainwright et al. [23] took a similar approach to the problem of intractable inference by decomposing the graph into a set of overlapping, tractable models. These techniques, however, tie parameter learning between the tractable models, imposing an expensive increase in inference time. SEC loosens this contraint since it is not needed for the filtering task.

4.2 Probabilistic Ensembles in ACMI

With the well-documented success of ensemble methods in classification tasks, we seek to extend the idea of aggregating multiple estimates to probabilistic graphical models. As discussed in the previous section, current efforts in the area rely on simplifying the structure of an intractable graph to create a collection of tractable problems [23, 25]. These techniques, however, do not easily extend to the graph in ACMI, which is fully connected and thus difficult to convert to the necessary number of tree-structured graphs. In addition, previous work in ACMI introduced an approximation that exploited redundancies in messages passing to dramatically reduce the complexity of inference [8]. Converting ACMI to a tree-structured graph loses the gains from this approximation as well as important information encoded in edges. Thus, we are interested in a solution that boosts the accuracy of inference, not that tractability.

We propose Probabilistic Ensembles in ACMI (PEA), shown in Figure 5 and Algorithm 1. PEA is a framework for generating and combining multiple approximate inference solutions to create more accurate protein structures. As with ensemble learning methods in classification, there are two major design components to address. In Section 4.2.1, we discuss how PEA generates a diverse set of solutions (i.e., ensemble components). In Section 4.2.2, we describe our framework for aggregating these solutions to produce a protein structure. Both of these components modify the existing ACMI framework (Figure 2). Specifically, generating diverse solutions amends the inference process in Phase 2 and combining the solutions takes place in structure sampling of Phase 3.

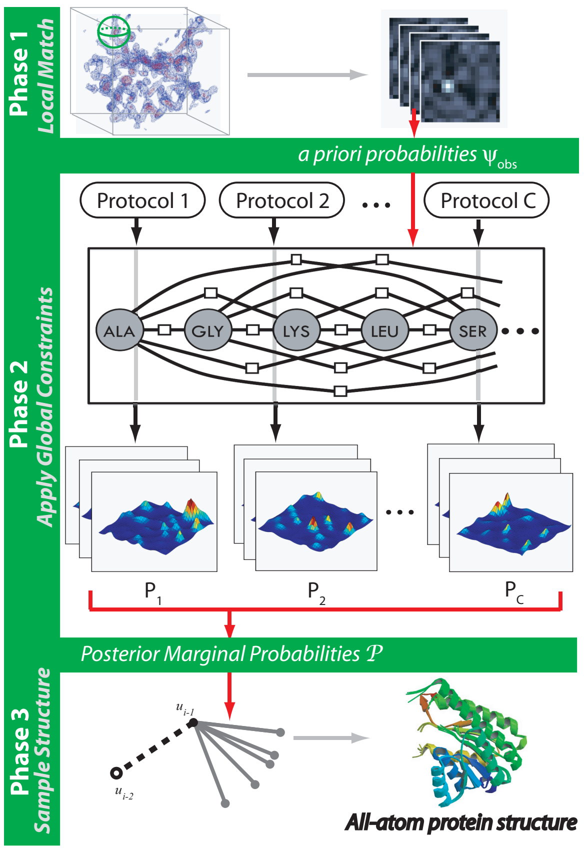


Figure 5: Probabilistic Ensembles in ACMI (PEA). Phase 1 is the same as in the ACMI framework (Figure 2). Phase 2 performs C independent inference runs, each with a unique protocol. This results in a set of C marginal probabilities for each amino acid's location. Phase 3 aggregates the set of marginal probabilities to produce a protein structure.

4.2.1 Generating Ensemble Components

From Section 3.1, Equation 1 calculates $P(U|M)$ – the probability distribution over all possible protein structures given the density map. Since this calculation is intractable, Phase 2 of ACMI produces $\hat{p}_i$, the approximate marginal probability of amino acid i's location for each amino acid in the protein sequence. Rather than performing inference once, our proposed framework, PEA, performs several independent runs of inference. As shown in Figure 5, each run (C in total) uses a unique protocol and outputs its own marginal probability distribution for each amino acid's lo-

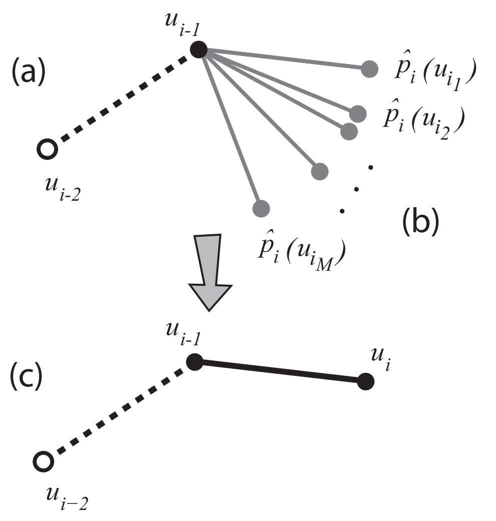


Figure 6: ACMI's Phase 3 sampling step for amino acid i. In a) Phase 3 samples M possible new locations, u_{i_m}. In b) these locations are weighted by their agreement with the Phase 2 probability, $\hat{p}_i$. In c) one location is chosen from the weighted distribution to be amino acid i's location, u_i.

cation. Phase 2, in total, produces a matrix of probability distributions $\mathcal{P} = (\hat{P}_1, \hat{P}_2, \ldots, \hat{P}_C)$ where each ensemble component c produces $\hat{P}_c = (\hat{p}_{1_c}, \hat{p}_{2_c}, \ldots, \hat{p}_{i_c})$. Here, $\hat{p}_{i_c}$ represents the probability of amino acid i's location in the density map according component c of the ensemble.

As mentioned in Section 4.1, a desired property of an ensemble is that the individual components are diverse. Fortunately, Elidan et al. [10] showed the choice of a message-passing protocol (i.e., what order to send and receive messages between nodes) has a large effect on the outcome of belief propagation, particularly in complex, loopy graphs such as Acmi. We previously examined the effect of using various message-passing schedules on Acmi's performance and found that this decision choice has a significant impact on the final solution of Phase 2 [20]. Section 4.3 provides example message-schedules for generating ensemble components in Pea.

4.2.2 Aggregating Ensemble Components

In DiMaio et al. [7], we developed Phase 3 of Acmi which utilizes *particle filtering* [1], a sampling algorithm, to generate all-atom protein structures given the posterior marginal probabilities from Phase 2. Briefly, Phase 3 is an iterative process which sequentially grows a protein structure one amino acid at a time[3]. Figure 6 shows how, at a given iteration, Phase 3 samples the location, u_i, of amino acid i. Phase 3 first samples M potential locations for the new amino acid based on the location of already placed amino acids in the sequence and a distribution of known angles and distances between neighboring amino acids (i.e., the adjacency potential function $\psi_{adj}(u_{i-1}, u_i)$ from Section 3.1). Next, Phase 3 assigns a weight, w_m, to each sampled estimate, u_{i_m}, correlated with the likelihood of amino acid i being in that location. This is calculated from the Phase 2 posterior marginal probabilities:

$$w_m \propto \hat{p}_i(u_{i_m}). \tag{4}$$

Lastly, Phase 3 chooses one of the M samples as its prediction for amino acid i's location in the structure. The sample is chosen with probability proportional to w_m.

In Pea, we must adjust Phase 3 to deal with the existence of multiple Phase 2 posterior probabilities for each amino acid. We propose several functions for combining these probabilities into a single weight measurement. First, we look at the average score over all ensemble components using a mixture model:

$$w_m \propto \sum_c \pi_c \cdot \hat{p}_{i_c}(u_{i_m}) \tag{5}$$

where π_c is the mixture weight, representing the confidence that component c provides the correct distribution.[4] The average score should perform well if the true location tends to have a high score across all runs of inference while false positives are uncorrelated between runs. False positives would be smoothed out and consistent peaks would maintain high probabilities. Conversely, strong divergences between models would smooth away most information. Another proposed weight function is to instead take the maximum score for a given location across all components:

$$w_m \propto \max_c \hat{p}_{i_c}(u_{i_m}). \tag{6}$$

In difficult sections of a protein, it is very likely that most models will miss the correct location since there is very little evidence. Given multiple estimates, it is more likely that one model found the correct answer. Using the maximum score allows Phase 3 to capture the correct location in this situation, although it may also over-emphasize false positives. Lastly, we consider using a subsampling approach where Phase 3 will randomly select one of the ensemble components to score the location:

$$\begin{gathered} c \sim U[1, C] \\ w_m \propto \hat{p}_{i_c}(u_{i_m}) \end{gathered} \tag{7}$$

where $U[1, C]$ returns an integer between 1 and C with uniform weight. Alternatively, c could be drawn from a weighted distribution based on π_c in Equation 5. This technique fits intuitively into the sampling framework of particle filtering where multiple structure estimates exist to explore several different paths to the end state.

4.3 Experimental Methodology

In Section 5, we compare the performance of our original Acmi framework from DiMaio et al. [7] to our proposed algorithm, Pea. Previous results show Acmi is the state-of-the-art technique for low-quality protein images, thus related approaches are not compared in this paper. We use a set of ten *experimentally-phased* electron-density maps described in DiMaio et al. [7] for validation. This data was provided by the Center for Eukaryotic Structural Genomics (CESG) at UW–Madison. The maps were initially phased using either the Solve [22] or Sharp [5] packages, with non-crystallographic symmetry averaging used to improve the

[3]Phase 3 maintains multiple estimates (or particles) during the sampling process and uses separate steps to sample backbone and side-chain atoms. For simplicity, we only consider one particle's backbone placement in this description.

[4]π_c can be set by various measures, such as entropy or prior knowledge. We use uniform weights in our experiments.

map quality where possible. Based on the electron density quality, expert crystallographers selected these maps as the "most difficult" from a larger data set of twenty maps. These structures have been previously solved and deposited to the PDB, enabling a direct comparison with the correct model[5]. However, all ten required a great deal of human effort to build the final atomic model.

Phase 1 (performing an independent search for local features) is the same for both algorithms, meaning Phase 2 for both the original ACMI and proposed PEA algorithms begin with the same input. For the experiments in Sections 5.1 and 5.2, we consider 3 variations of ACMI's belief-propagation protocol for Phase 2:

- ORIG, the original protocol of ACMI which is run for 40 iterations per amino acid in a round-robin fashion starting with amino acid 1, proceeding left to right, and then reversing at the end of each pass.
- EXT, an extended version of the original protocol going for 160 iterations.
- BEST, the top-performing individual version of ACMI from the four protocols considered for PEA (see below).

The BEST protocol provides an overly optimistic estimate of ACMI to see how PEA performs as an ensemble relative to its individual components. For PEA, we generate an ensemble of size 4 with each component having its own protocol:

- Protocol 1 is the same as ORIG above.
- Protocol 2 starts halfway through the sequence.
- Protocol 3 is like ORIG, but runs for 20 iterations
- Protocol 4 employs guided belief propagation [20].

For the learning curve in Section 5.3, 50 protocols were generated. All were based on the standard, round-robin schedule and executed for 40 iterations. Each varies in the starting location and the direction of the first iteration with half going left to right and the other half going right to left.

Phase 3 experiments in Section 5.2 were run with 100 particles (see DiMaio et al. [7] for details). For the aggregators in Section 4.2.2, our shown results reflect a uniform mixture weight. We considered a weight based on the entropy of the distributions, but the results were statistically equivalent to using a uniform weight. We determine statistical significance using a paired t-test.

5. RESULTS

Using the methodology described in Section 4.3, we compare the performance of our new approach of using Probabilistic Ensembles in ACMI (PEA) against the original ACMI algorithm [7]. We compare the results across a set of ten difficult protein structures. These solutions were previously solved utilizing significant human effort. Since Phase 1 remains unchanged between the two algorithms, we do not detail the results of generating the observation potentials. Section 5.1 first assesses the quality of approximate inference by comparing the accuracy of the Phase 2 outputs by the two approaches. In Section 5.2, we feed these Phase 2 probabilities to Phase 3 to measure the accuracy of the all-atom protein structure models produced by PEA and ACMI. Lastly, Section 5.3 shows how the accuracy of PEA changes as the number of ensemble components increases.

[5]Test-set solutions were removed from ACMI's fragment library to blind all methods from the true result.

5.1 Approximate Inference

Our first experiment assesses the quality of approximate inference solutions produced in Phase 2 for both ACMI and PEA by examining the accuracy of posterior marginal probabilities. In this experiment, ACMI and PEA use the same Phase 1 outputs to run their respective Phase 2 algorithms and halt before executing Phase 3. PEA runs Phase 2 with four ensemble components using the protocols specified in Section 4.3. We consider the maximum score aggregator from Equation 6 and the average score aggregator from Equation 5 (MAX and AVG, respectively). The sampling algorithm from Equation 7 performs aggregation as a step in Phase 3 and cannot be compared here. For ACMI, we test the original, round-robin protocol (ORIG), an extended run of inference (i.e., 160 iterations) in ACMI (EXT), and the best-performing *individual* component of PEA (BEST).

Figures 7a and 7b show the results of running these techniques on a set of difficult protein images. Figure 7a shows the percentile rank which represents how highly ranked the correct solution (i.e., location from the deposited structure in the PDB) is in the posterior marginal probabilities. To calculate the percentile for a given amino acid, we sort all of the probabilities in the posterior from highest to lowest and then calculate the percent of locations lower than the true location. The optimal score of 100 means the true location had the highest probability value in the map. In Figure 7b, the negative log-likelihood is the probability value for the true location, transformed as a negative-log score. Here, we desire lower values as they indicate higher probabilities. In both figures, one column represents the average score across all 10 test-set proteins and across all amino acids in those proteins. Columns are color coded based on whether they are instance of ACMI (light green) or PEA (dark red).

Both figures show that the ensemble method, PEA, drastically outperforms the existing, single inference version of ACMI across all protocols. Both the maximum and average aggregators obtain scores in the 89th percentile compared to the original ACMI protocol which averages scores in the 66th percentile. This implies that, on average, there are three times as many false positives in ACMI versus PEA. The negative log-likelihoods tell a similar story; the probability scores improve by over three orders of magnitude by using ensembles. The results for the best individual component of PEA are only slightly better than standard ACMI, showing that PEA benefits from combining multiple, good models rather than from generating one very good model. The extended run of standard ACMI shows minor improvements as well, but comes nowhere near the performance of PEA, showing that the gains of our ensemble method cannot be explained away by an increase in CPU resources. In fact, the results of all pairwise differences between the PEA variations and the three ACMI variations are statistically significantly at scores of $p < 0.01$ for both metrics in Figure 7 based on a paired t-test.

5.2 Protein Structures

While the previous results indicate our ensemble technique improves the accuracy of approximate inference probabilities, biochemists are more interested in the actual protein structures produced. As a follow-up experiment, we used the marginal probabilities from Section 5.1 as the input for Phase 3 of the ACMI and PEA algorithms, respectively, to produce all-atom protein structures for all 10 of our test-set

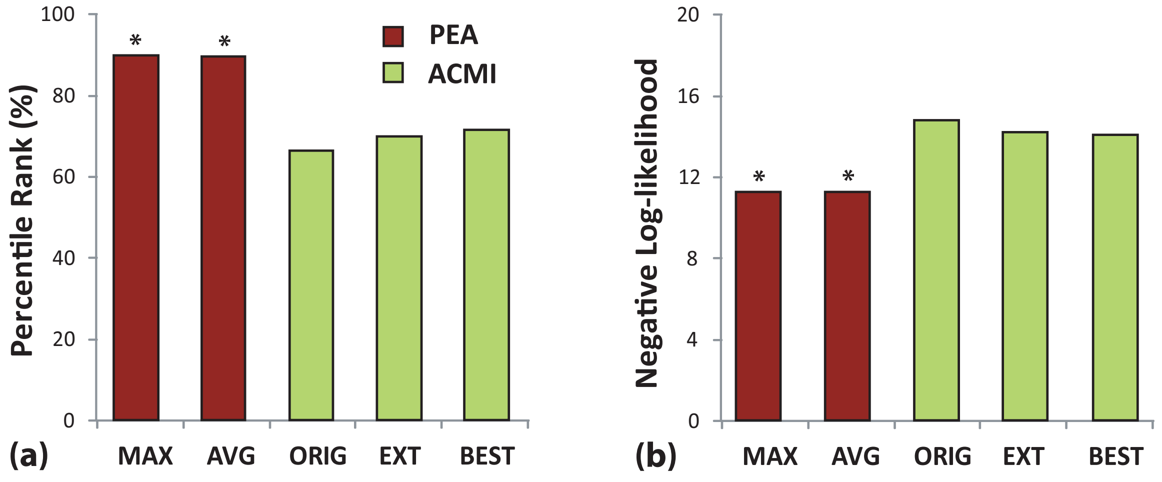


Figure 7: Accuracy of inference solutions. In a) percentile rank of the true solution's probability. A higher percentile means the algorithm puts the true location of an amino acid closer to the top of a list of sorted probabilities. In b) the negative log-likelihood of the true solution. Lower scores mean a higher probability value for the correct answer. In both, columns are the average score over all amino acids in all test-set proteins. Dark (red) bars represent variations of PEA, while light (green) bars represent variations of ACMI. An * denotes a statistically significant difference with ORIG at $p < 0.01$.

proteins. We use the completeness and correctness of the resulting protein structures to compare our proposed aggregators for Pea against Acmi.

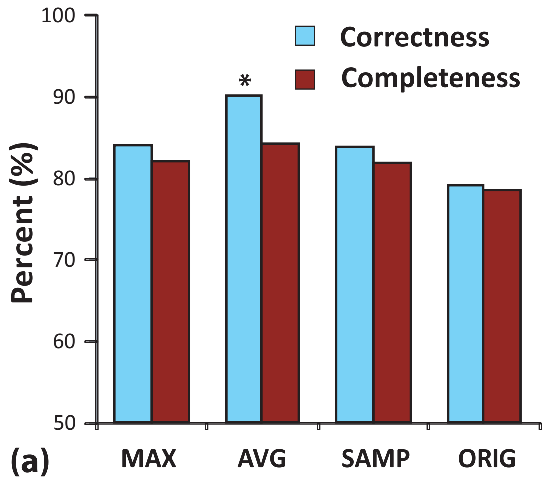
Figure 8a shows the averaged results of our experiments. The first three pairs of columns represent, respectively, the maximum (MAX), average (AVG), and sampling (SAMP) aggregators for Pea presented in Section 4.2.2. The fourth pair of columns represent the original Acmi protocol. Within each pair, the first column represents the *correctness* of the predicted protein structure – that is, what percentage of amino acids predicted were within 2 Å of their corresponding true-solution location. This is similar to the *precision* metric used in information retrieval. The second column represents the *completeness* of the predictions – what percent of amino acids available in the PDB solution were accurately predicted (within 2 Å). This is akin to a *recall* metric. Each column represents an average over all ten test proteins. The top performer across both metrics was Pea using the averaging function to aggregate ensemble components. On average, 90.3% of its predicted amino-acid locations were correct (compared to 79.3% for the original Acmi algorithm) while completing 84.3% of the real structure (78.6% for Acmi). Importantly, all three Pea methods outperform Acmi in both correctness and completeness measures.

Figure 8b provides a closer comparison of Pea versus Acmi. Here, each datapoint represents the results of one protein in our test set. The x-axis value is the accuracy of the original Acmi algorithm and the y-axis is the accuracy of Pea using the average aggregator. To assess accuracy, we use an F-measure to combine the correctness and completeness metrics from Figure 8a. The F-measure is commonly used in the information retrieval community to balance both the need for high precision and high recall. Here, we use the traditional F_1 metric, which is the harmonic mean of correctness and completeness:

$$F_1 = 2 \cdot \frac{correctness \cdot completeness}{correctness + completeness} \tag{8}$$

The line represents equivalent performance, and the shaded region represents values where Pea outperforms ACMI. In every test case, Pea performs better than or equal to Acmi in the F_1 metric, affirming the results from Figure 8a. The largest improvement comes in the most difficult test case, with the F_1-score improving from 0.25 to 0.66. This corresponds to an extra 41 percentage points of the true structure being built and 42 percentage points of extra predictions being correct. Overall, Pea shows substantial improvement in 6 of the 10 proteins with equal performance in the other 4, although these values are not statistically significant. The variance in overall performance is not correlated with either the size of the protein or image, but indicative of the range in image quality in our test set.

Figure 8b only considers the average aggregator for Pea since it performed better than the alternative options. As hypothesized in Section 4.2.2, the averaging aggregator's main advantage is that it can smooth away "noisy" probabilities. That is, as seen in Figure 7, Acmi's individual inference runs contain many incorrect locations (i.e., false positives) with higher probabilities than the correct solution. This makes it more difficult for Phase 3 to find the correct location for the structure during sampling. If an ensemble produces diverse solutions, however, than the "noisy" locations should be independent between runs. When averaged together, these incorrect peaks are smoothed away since they are low in probability in most components, and the signal from the true location will be boosted since its signal is detected by multiple inference runs. Thus, while each individual run produces moderate results, the average together reveals only a handful of possible solutions. The maximum aggregator and sampling aggregator also produced improved inference probabilities, but did not translate into the same



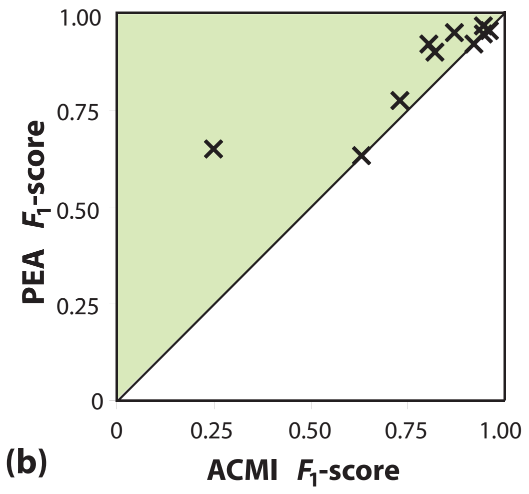


Figure 8: Protein-structure prediction accuracy. In a) correctness (light blue) specifies the percent of predicted structure within 2 Å of the true answer, averaged over all proteins. Completeness (dark red) is the percent of the true structure a method predicted within 2 Å. ORIG is the standard ACMI algorithm and MAX, AVG, and SAMP are variations of PEA. An * indicates statistically significant difference compared to ORIG at $p < 0.05$. In b) we show a detailed comparison of F_1-scores for ACMI (x-axis) and PEA using the averaging aggregator (y-axis). F_1 is the harmonic mean between the correctness (precision) and completeness (recall) metrics. Each point represents one protein and the shaded region indicates better scores for PEA.

level of improvement in structure quality as the averaging aggregator. It is difficult to pinpoint the exact reason, but the areas of major difference happened to be in regions of the map with this least amount of signal, implying the averaging aggregator handles noise the best.

5.3 Ensemble Learning Curve

As a last experiment, we consider how the size of an ensemble effects the accuracy of inference in PEA. Due to resource limitations, we could not run larger ensembles sizes for the previous experiments. Instead, for the seven smallest test-set proteins, we generated ensembles with various number of components, ranging from 1 to 50. We assessed each using percentile scores as described for Figure 7a. Figure 9 shows the learning curve for seven of our test-set proteins as the number of ensemble components increases (the values past 30 are not shown since no change occurred). PEA uses the mixture-model average aggregator to combine posteriors. The light, dashed lines represent the inference results for one test-set protein while the thick, black line represents the average performance across the seven proteins. As the figure shows, PEA gains accuracy from adding more components, making its largest leap in performance with the first 10 ensemble components before seeing very little improvement after 20 component ensembles.

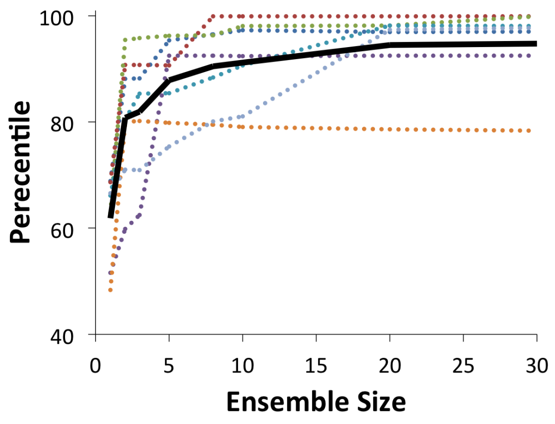


Figure 9: Learning curve for ensemble inference. Each dashed line represents one protein's percentile scores for Phase 2 posteriors as the number of ensemble components increases. The solid black line represents the average learning curve.

6. CONCLUSIONS AND FUTURE WORK

While ACMI was previously shown to outperform other automated density-map interpretation methods in building all-atom protein structures in low quality electron-density maps [7], performing approximate inference in ACMI's model is an expensive process in need of advanced inference methods. In this work, we developed a new approximate inference method based on the concept of ensemble learning methods from the supervised machine learning community. Our new framework, Probabilistic Ensembles in ACMI (PEA), executes several independent runs of inference to provide multiple, diverse solutions to the problem. We suggest several protocols for generating unique solutions for each component of the ensemble as well as different techniques for aggregating these models to produce a single, accurate prediction of the protein structure.

Our results show PEA provides improved performance on a test-set of 10 difficult protein images. This improvement is seen in the accuracy of the inference process, where the probability distributions from PEA were statistically significantly better in terms of both percentile rank and probability value assigned to the correct location of each amino acid. The results show that this improvement could not be explained by either extra CPU resources or by using the single-best component of PEA. More importantly, PEA's improved inference translates into more complete and correct protein structures. In the future, we seek to test ACMI on a larger set of proteins, including membrane proteins which present many difficulties for crystallographers [3].

While we presented ensembles of approximate inference solutions for the task of protein-structure determination, our method can generally be applied to difficult inference problems where the complexity of probabilistic graphical models limits the accuracy of current methods. In future work, we look to find such applications, and to provide an in-depth comparison to related inference techniques that rely on simplifying the graph structure [23].

7. ACKNOWLEDGMENTS

This work is supported by NLM grant R01-LM008796 and NLM training grant T15-LM007359. Support for our collaborators at the University of Wisconsin Center for Eukaryotic Structural Genomics (CESG) is provided by NIH Protein Structure Initiative Grant GM074901. The ACMI system is a collaboration of this paper's authors and Frank DiMaio, George N. Phillips, Jr., and members of the Phillips Laboratory at the University of Wisconsin–Madison. ACMI and the data set of experimentally-phased density maps is available online at `http://www.cs.wisc.edu/acmi/`.

8. REFERENCES

[1] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters. *IEEE Transactions of Signal Processing*, 50, 2001.

[2] E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36(1-2), 1999.

[3] E. P. Carpenter, K. Beis, A. D. Cameron, and S. Iwata. Overcoming the challenges of membrane protein crystallography. *Current Opinion in Structural Biology*, 18(5):581–586, 2008.

[4] K. Cowtan. The BUCCANEER software for automated model building. *Acta Crystallographica*, D62, 2006.

[5] E. de La Fortelle and G. Bricogne. Maximum-likelihood heavy-atom parameter refinement for the multiple isomorphous replacement and multiwavelength anomalous diffraction methods. *Methods in Enzymology*, 276, 1997.

[6] T. G. Dietterich. Ensemble methods in machine learning. *Lect. Notes in Comp. Science*, 1857, 2000.

[7] F. DiMaio, D. Kondrashov, E. Bitto, A. Soni, C. Bingman, G. Phillips, and J. Shavlik. Creating protein models from electron-density maps using particle-filtering methods. *Bioinformatics*, 23, 2007.

[8] F. DiMaio and J. Shavlik. Belief propagation in large, highly connected graphs for 3D part-based object recognition. In *Proceedings of the Sixth IEEE International Conference on Data Mining (ICDM 06)*, Hong Kong, 2006.

[9] F. DiMaio, A. Soni, G. N. Phillips, and J. Shavlik. Spherical-harmonic decomposition for molecular recognition in electron-density maps. *Int. J. of Data Mining and Bioinformatics*, 3(2), 2009.

[10] G. Elidan, I. McGraw, and D. Koller. Residual belief propagation: Informed scheduling for asynchronous message passing. In *Proceedings of the Twenty-second Conference on Uncertainty in Artificial Intellignece (UAI 06)*, 2006.

[11] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. Pattern Anal. and Mach. Intell.*, 6, 1984.

[12] T. Ioerger and J. Sacchettini. The TEXTAL system: Artificial intelligence techniques for automated protein model building. *Methods in Enzymology*, 374, 2003.

[13] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. MIT Press, 2009.

[14] R. Maclin and D. W. Opitz. An empirical evaluation of bagging and boosting. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI 97)*, 1997.

[15] R. Morris, A. Perrakis, and V. Lamzin. ARP/wARP and automatic interpretation of protein electron density maps. *Methods in Enzymology*, 374, 2003.

[16] K. Murphy, Y. Weiss, and M. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of Uncertainty in Artificial Intelligence (UAI 99)*, 1999.

[17] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufman Publishers, San Mateo, CA, 1988.

[18] A. Perrakis, R. Morris, and V. Lamzin. Automated protein model building combined with iterative structure refinement. *Nature Structural and Molecular Biology*, 6(5), 1999.

[19] Protein Data Bank (PDB). PDB current holdings breakdown, August 2011. `http://www.rcsb.org/pdb/statistics/holdings.do`.

[20] A. Soni, C. Bingman, and J. Shavlik. Guiding belief propagation using domain knowledge for protein-structure determination. In *Proceedings of the ACM International Conference on Bioinformatics and Computational Biology (ACM-BCB 2010)*, Niagara Falls, NY, USA, August 2010.

[21] T. C. Terwilliger. Structural genomics in North America. *Nature Structural Biology*, 7:935–939, 2000.

[22] T. C. Terwilliger. Automated main-chain model building by template matching and iterative fragment extension. *Acta Crystallographica*, D59, 2003.

[23] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Tree-reweighted belief propagation algorithms and approximate ML estimation by pseudo-moment matching. In *Workshop on Artificial Intelligence and Statistics (AISTATS)*, 2003.

[24] G. Wang and R. J. Dunbrack. PISCES: A protein sequence culling server. *Bioinformatics*, 19, 2003.

[25] D. Weiss, B. Sapp, and B. Taskar. Sidestepping intractable inference with structured ensemble cascades. In *Advances in Neural Information Processing Systems 23*. 2010.