CONSENSUS FOLD RECOGNITION BY PREDICTED MODEL QUALITY

JINBO XU, LIBO YU, MING LI

School of Computer Science, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada
Email: {l3yu, j3xu, mli}@uwaterloo.ca

Consensus-based protein structure prediction methods have been proved to be successful in recent CASPs (Critical Assessment of Structure Prediction). By combining several weaker individual servers, a meta server tends to generate better predictions than any individual server. In this paper, we present a Support Vector Machines (SVM) regression-based consensus method for protein fold recognition, which is a key component for high-throughput protein structure prediction and protein functional annotation. Our SVM model extracts the features from a predicted structural model by comparing it to other models generated by all the individual servers and then predicts the quality of this model. Experimental results on several LiveBench data sets show that our consensus method consistently performs better than individual servers. Based on this approach, we have developed a meta server, Alignment by Consensus Estimator (ACE), which is participating in CASP6 and CAFASP4 (Fourth Critical Assessment of Fully Automated Structure Prediction). ACE is available at http://www.cs.uwaterloo.ca/~l3yu/consensus.htm.

1. Introduction

Protein three-dimensional structure determination has been a fundamental challenge in molecular biology. The experimental approaches like X-ray crystallography or nuclear magnetic resonance spectroscopy (NMR) turn out to be costly and low throughput. Protein structure prediction by computational methods has been addressed for more than three decades and only limited progress has been made. Recently, with the enlargement of protein databases and the advances in high-performance computing facility, great progress has been achieved in this area and some community-wide experiments such as CASP[1,2,3] and LiveBench[4,5] have been carried out. Since the first CAFASP in 1998, great progress in automatic structure prediction has been made, and more and more fully automatic structure prediction servers have been developed. For example, the number of prediction servers in CAFASP3[6] almost doubled, compared to that in CAFASP2.[7]

In CAFASPs, it has been observed that for different targets, the best predictions are often made by different servers.[6] No single server can reliably generate the best models for all the targets. In contrast, consensus predictors, also called meta servers, can consistently produce better results than individual servers. A meta server takes the top models generated by a set of individual servers as its input and chooses the best model or assembles a new hybrid model as the prediction. Along with CASPs, several meta servers, such as PCON,[8] 3D-Jury,[9] Pmodeller[10] and 3D-SHOTGUN,[11,12] have been developed.

The consensus method was first applied in fold recognition by some individual servers rather than meta servers. INBG[13] threads a sequence to a template by using five different scoring functions. The template with the highest average score is chosen as the final prediction. 3DPS[14] aligns a query sequence to each protein structural template using three different scoring functions and chooses the alignment with the maximum score as the final prediction.

Making a consensus prediction based on the results of several individual servers was first successfully applied in CASP4 by a group named CAFASP-CONSENSUS.[2] This group derived predictions by inspecting and analyzing the outputs from the automated fold recognition servers running in the CAFASP2. As a result, the CAFASP-CONSENSUS outperformed any individual server in CAFASP2 and ranked seventh among the human predictors in CASP4.[2] This led to the development of the first automated consensus server PCON, which made use of the results of six individual servers. PCON uses a neural network to predict the quality of one model by comparing it with other models and reports the model with the highest quality score as the final prediction. PCON performs better than any of its component servers, especially in specificity. Pmodeller is a new version of PCON. It predicts the quality of a model by combining PCON and ProQ[15] together. ProQ is a neural network-based tool to predict the quality of a protein model based on the structural characteristics of one model. With ProQ, certain amount of improvement is achieved.

Unlike PCON, 3D-Jury does not use machine learning methods and no training procedure is required, which makes it simple and flexible. 3D-Jury compares input models with each other using MaxSub[16] and a similarity score is obtained for each pair of models. Then a quality index for each model is calculated based on pairwise MaxSub scores. The 3D-Jury can be operated in two modes in which quality indices are calculated in two different ways.

Both PCON and 3D-Jury only choose the best possible model from all the input models. Some meta servers go beyond this selection-only method. 3D-SHOTGUN[11,12] is such an example and capable of assembling a new hybrid model from the input models. It is believed that this feature makes it usually more sensitive and specific than other meta servers. In spite of the success with this approach, it has also been observed that the hybrid models assembled by 3D-SHOTGUN sometimes contain nonnative-like structural fragments.

In this paper, we are to explore the possibility of developing a selection-only meta server by utilizing more effective features and applying state-of-the-art machine learning techniques. We will present a new consensus method based on SVM regression, which turns out to be quite effective at boosting the performance, especially when not many high-performance individual prediction servers are available. By using SVM regression approach, for each target, we can predict the quality of each model, which in turn is used to rank all the input models generated by individual servers.

The rest of this paper is organized as follows. Section 2 briefly introduces the idea of SVM regression. Section 3 discusses how to extract some effective features from each model. Section 4 describes some experiments and discusses various factors that may influence the performance of the meta server. Finally, section 5 draws some conclusions from the experimental results.

2. SVM Regression

Approximating a real-valued function from a finite set of samples is a central problem in many areas. The commonly used techniques for such tasks are linear regression, or logistic regression, which are often not sufficient to approximate complex functions with a high degree of nonlinearity. In such cases, nonlinear regression methods should be used to improve the approximation accuracy. For our application, we are to use SVM regression to approximate the functional relationships between the features of a predicted structural model and its structural quality.

Support Vector Machine was developed by Vapnik et al. in 1970's.[17] SVM classifiers turned out to have excellent generalization performance and were successfully applied in many areas like pattern recognition and information retrieval.[18] When applied in regression applications, SVM also gives excellent performance.[19] We will start with linear SVM regression which is simple and straightforward.

Linear SVM Regression Given training data $\{x_i, y_i\}$, $i = 1, \cdots, n$, where for our application, $x_i \in R^m$ is the model feature vector and y_i is the model quality score. An ϵ-insensitive loss function is used, that is, we look for a function $f(x) = w \cdot x + b$ that has at most ϵ deviation from the actual obtained y_i for all the data points. Here w is a vector in R^m. By looking for a minimized $||w||$, a unique solution can be defined. This can be formulated as a convex optimization problem as shown below [18]:

$$\begin{array}{ll} \text{Minimize} & \frac{1}{2}||w||^2 \\ \text{Subject to} & \begin{cases} y_i - w \cdot x_i - b \le \epsilon \\ w \cdot x_i + b - y_i \le \epsilon \end{cases} \end{array} \tag{1}$$

There is no guarantee that such $f(x)$ always exists. So we can introduce slack variables ξ_i, ξ_i^* to cope with otherwise infeasible solutions of the optimization problem (1).

$$\begin{array}{ll} \text{Minimize} & \frac{1}{2}||w||^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i) \\ \text{Subject to} & \begin{cases} y_i - w \cdot x_i - b \le \epsilon + \xi_i \\ w \cdot x_i + b - y_i \le \epsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \end{cases} \end{array} \tag{2}$$

where C penalizes the amount to which deviations larger than ϵ are tolerated.

By introducing Langrangian multipliers λ_i and λ_i^* ($i = 1, \cdots, l$), we can obtain the dual formulation of the original optimization problem (1).

$$\begin{array}{ll} \text{Maximize} & \begin{cases} -\frac{1}{2}\sum_{i,j=1}^{l}(\lambda_i - \lambda_i^*)(\lambda_i - \lambda_i^*)(x_i \cdot x_j) \\ -\epsilon\sum_{i=1}^{l}(\lambda_i + \lambda_i^*) + \sum_{i=1}^{l} y_i(\lambda_i - \lambda_i^*) \end{cases} \\ \text{Subject to} & \begin{cases} \sum_{i=1}^{l}(\lambda_i - \lambda_i^*) = 0 \\ \lambda_i, \lambda_i^* \in [0, C] \end{cases} \end{array} \tag{3}$$

Solving (3), we have:

$$w = \sum_{i=1}^{l} (\lambda_i - \lambda_i^*) x_i$$
$$f(x) = \sum_{i=1}^{l} (\lambda_i - \lambda_i^*)(x_i \cdot x) + b \tag{4}$$

Note that w is a linear combination of support vectors x_i. And the support vectors appear only in the form of dot product with x in the trained SVM machine.

Nonlinear SVM Regression To make SVM nonlinear, the straightforward way is to map x_i to a higher dimension space. The drawback is that it can easily become computationally prohibitive. A cheaper way to achieve this is to make an implicit mapping via kernel functions. Instead of defining $\phi(\cdot)$ explicitly, we can make an implicit mapping by defining $\kappa(x, x_i) = \phi(x) \cdot \phi(x_i)$ directly without knowing $\phi(\cdot)$.[18] $\kappa(x, x_i)$ is called the kernel function. Now the SVM has a form like:

$$f(x) = \sum_{i=1}^{l} (\lambda_i - \lambda_i^*) \kappa(x, x_i) + b \tag{5}$$

Various kernels can be used for different applications. The most commonly used ones are polynomial kernels and Radial Basis Function (RBF) kernels.

3. Feature Extraction

Features are critical for the performance of machine learning based meta predictors. For our meta server, all the features are extracted from the structural comparisons of the input models. After the top 10 models reported by each server are collected, all the models are compared with each other and a similarity score is obtained for each pair. The similarity between two models is calculated by MaxSub,[16] a program originally designed to measure the quality of a single model. The quality score of a model serves as the objective function of our SVM regression method. MaxSub is a sequence-dependent quality assessment tool that identifies the maximum superimposable subset of C_α atoms of two protein structures.

Let $M_{a,b}$ denote the b^{th} model reported by server a and $sim(M_{a,b}, M_{i,j})$ the similarity score between model $M_{a,b}$ and $M_{i,j}$. Let N denote the total number of servers and n the number of top models reported by one server. For each model, we extract three groups of features as follow.

Feature I This feature is specific to each model and can be calculated by Formula 6. That is, for a given model $M_{a,b}$, we calculate its similarity with all the models generated by all the servers excluding server a.

$$\frac{1}{(N-1)n} \sum_{i=1, i \neq a}^{N} \sum_{j=1}^{n} sim(M_{a,b}, M_{i,j}) \tag{6}$$

Feature II This feature is also specific to each model and can be calculated by Formula 7. Given a model $M_{a,b}$, we compare it with all the models from another one server and pick out the maximum similarity score. This procedure is repeated for each of all the rest servers. Finally, we calculate the average of those selected maximum scores.

$$\frac{1}{N-1}\sum_{i=1,i\neq a}^{N}\max_{j=1}^{n}\{sim(M_{a,b},M_{i,j})\} \tag{7}$$

Feature III Feature III is different from feature I and II and composed of a set of subfeatures rather than one number. In addition, it is not model-specific but target-specific. That is, for different target proteins, feature III is different and all the models predicted for the same target protein have the same feature III. To obtain feature III, the similarity between predictions made by every two servers must be calculated. So for N servers, there are C_N^2 subfeatures altogether. The disadvantage is that if N is very large, this number grows quickly. But later we will show that a large N may not be a good choice. For server a and i, the similarity between the models generated by them can be calculated by Formula 8.

$$\frac{1}{n^2}\sum_{b=1}^{n}\sum_{j=1}^{n}sim(M_{a,b},M_{i,j}) \tag{8}$$

Feature I and II can be viewed as two methods measuring the degree to which each model is supported by other models. It is observed that the most accurate model usually have more similarity with other models than a less accurate model does. Based on this assumption, one approach to estimate the quality of a model is to compare it with all the models from the other servers, which is the basis for feature I and II. Note that for feature I and II, the models from the same server as $M_{a,b}$ are ignored. The reason is that it has been observed in the experiments that the models reported by the same server are more likely to be similar to each other. So including them in the sum may introduce bias and degrade the performance. Instead, 3D-Jury[9] takes into consideration the models generated by the same server in calculating the support of one model. 3D-Jury-All uses a formula similar to Eq. 6 to calculate the support of one model and 3D-Jury-Single uses a formula similar to Eq. 7.

Feature I and II are the main driving force of our approach and feature III is an auxiliary feature. Feature III represents the similarity between the two sets of structural predictions made by any two servers for a particular target protein. This can help in some cases to estimate the performance of different servers with respect to the same target protein, which indirectly helps to distinguish models in some cases. For instance, suppose there are three servers, named a, b, c. Assume at any time, the majority of servers make correct predictions. If we know that for a particular target, server a and b have similar predictions, but server b and c, server c and a do not have similar predictions. Then it is quite possible that server c makes poor predictions on this target. Thus, feature III can help to estimate the model quality in this case. Note that all these features are calculated by averaging many similarity scores. From this point of view, these features are obtained from the raw scores in a statistical way. In this sense, if more models are involved in structural comparisons the variations of the features can be reduced and the performance can be improved. That is why all the top 10 models from each server are used.

As mentioned, the structural comparisons can be performed in different ways and different features can be extracted accordingly. For instance, we can compare one model with the top model from each of the other servers, or compare each model with all the models from other servers. In addition, some other features can be derived from the sequence alignments, such as the gap numbers, gap length, target length, and template length. It is also possible to generate some features by using some software such as PROCHECK[20] and WHATIF[21] to measure the stereochemical quality of each model. These software can calculate some structural parameters from the coordinates of a protein structure, such as torsion angles, hydrogen bond energy, which have been shown to correlate with the model quality. Based on the distribution of these parameters, some statistics about the model's stereochemical quality can be calculated, which provide a simple guide as to the reliability of the structure.[20] A similar idea has also been implemented in ProQ by using machine learning techniques to predict the quality of a model. ProQ uses neural network to predict the Maxsub score or LGscore of a protein model based on the intrinsic features of a model, such as atom-atom contact, solvent accessibility surfaces.

In spite of the abundance of the available features, not all the features are equally effective. The features mentioned above were tested and refined through a trial-and-error process. By testing the combinations of different features, we explored the powers of different features and eventually arrived at the features reported in this paper.

4. Experimental Results

To test the performance of our new consensus algorithm, we downloaded publicly available LiveBench 5-8 data to train the SVM regression model and test it. When we were doing experiments, LiveBench 8 was incomplete and only had 148 targets. In addition, since LiveBenchs are not totally blind, some servers might report the experimental structure of a target as the prediction. In this situation, we just removed this prediction to avoid bias. Thus we have four sets of data corresponding to LiveBench 5-8. In the following context, we will call the four data sets LiveBench5, Livebench6, LiveBench7, and LiveBench8 respectively. To test the performance of our meta server, we used the four-fold cross validation in the experiments. Specifically, we used one data set to train the SVM model and tested it with the other three data sets. This was repeated four times, each time with a different set used for training.

4.1. *Sensitivity*

The sensitivity is defined as the sum of the MaxSub scores of the top models for all the targets. We make use of the results of three individual servers FFA3,[22,23] 3DPS,[14] and FUG2.[24] To assure objectivity, we have avoided using our own RAPTOR server. The top 10 models from each server were collected so there were 30 models for each target. By comparing models with each other, we generated all the features. We used a RBF kernel in the SVM. There are several tunable parameters in the SVM regression model. We tried different settings of the parameters and selected the one with best performance. As shown in Table 1, no matter which training set is used, our SVM regression approach has a very stable sensitivity on all the test sets. The total MaxSub score on the four LiveBench

Table 1. MaxSub scores of ACE with three component servers used. The number of targets is shown under the name of each data set. One data set is used for training and the other three for testing. The average testing result for each data set is calculated and summed.

Training	LiveBench5	LiveBench6	LiveBench7	LiveBench8	sum of
data set	78	98	115	148	average
LiveBench 5	—–	77.31	93.18	257.53	—–
LiveBench 6	73.59	—–	91.73	256.57	—–
LiveBench 7	73.15	75.35	—–	256.87	—–
LiveBench 8	73.15	74.47	91.89	—–	—–
Average	73.30	75.71	92.27	256.99	498.27

Table 2. Sensitivity (MaxSub score) comparison with three component servers and other meta servers. The results of 3D-Jury are derived from three component servers FFA3, 3DPS and FUG2. The results of all the other servers are taken from LiveBench. PCON's results are only available for LiveBench 5-7.

Training	LiveBench5	LiveBench6	LiveBench7	LiveBench8	sum
data set	78	98	115	148	score
FFA3	69.68	66.30	88.97	234.52	459.97
3DPS	58.97	61.62	81.47	252.17	454.23
FUG2	59.53	63.54	79.71	233.59	436.37
PCON	62.79	68.65	83.77	—–	—–
3D-Jury-all	44.24	57.80	64.52	191.38	357.94
3D-Jury-single	64.09	65.36	88.26	250.08	467.79
ACE	73.30	75.71	92.27	256.99	498.27

Table 3. MaxSub scores of ACE obtained with six component servers.

Training	LiveBench5	LiveBench6	LiveBench7	LiveBench8	sum of
data set	78	98	115	148	average
LiveBench 5	—–	63.15	92.57	253.44	—–
LiveBench 6	69.22	—–	90.37	253.39	—–
LiveBench 7	68.47	68.52	—–	255.81	—–
LiveBench 8	67.30	69.11	90.94	—–	—–
Average	68.33	66.93	91.29	254.21	480.76

data sets is 498.27. Comparison with 3D-Jury and individual servers is shown in Table 2. We can see that 3D-Jury-All has poor performance, even not as good as some component servers. Both ACE and 3D-Jury-Single are better than any individual server. For ACE, its sensitivity is above that of any individual server by about 8%. And for the same set of three component servers, the sensitivity of ACE is higher than that of 3D-Jury-Single by 6%. ACE is also approximately 10% better than PCON in LiveBench 5-7. More important is that the performance of any individual server is not as stable as that our meta server. For example, FFA3 performs very well in LiveBench 5-7, but very badly in LiveBench 8. In contrast, 3DPS performs not as well as FFA3 in LiveBench 5-7, but much better than FFA3 in LiveBench 8. By using consensus method, our meta server ACE can generate a very stable output.

Based on the three servers we used, we did another experiment with three additional servers included, namely, INBG, SFPP,[25] MGTH.[26] The result is shown in Table 3. Even

Table 4. Sensitivity (MaxSub score) comparison between ACE, six individual servers and meta servers PCON and 3D-Jury. The results of 3D-Jury are derived from these six individual servers. The results of all the other servers are taken from LiveBench.

data	LiveBench5	LiveBench6	LiveBench7	LiveBench8	sum
set	78	98	115	148	score
FFA3	69.68	66.30	88.97	234.52	459.97
3DPS	58.97	61.62	81.47	252.17	454.23
FUG2	59.53	63.54	79.71	233.59	436.37
INBG	61.56	46.63	79.25	219.22	406.66
SFPP	40.34	50.02	58.67	186.72	335.75
MGTH	58.52	68.04	70.98	237.40	434.94
PCON	62.79	68.65	83.77	—	—
3D-Jury-all	64.09	65.36	88.26	250.08	467.79
3D-Jury-single	68.56	65.59	91.52	256.41	482.08
ACE	68.33	66.93	91.29	254.17	480.76

though theoretically we should be able to achieve better performance by using more servers. The experimental result shows that using six servers is surprisingly worse than using three servers. In spite of that, the meta server is still better than any individual server. When more servers are included, the meta server has more chance to collect even better models in its input. But at the same time, number of models increases. This increase may bring up two problems. First, if some poor quality models are included, they will contaminate the features extracted, which will result in performance degradation. Secondly, the capability of the machine learning method is not unlimited. When more candidates are to be considered, it becomes more difficult for the machine learning method to pick out the best one.

Comparisons with 3D-Jury method and six individual servers are listed in Table 4. In this case, ACE and 3D-Jury-Single have equivalent performance and have higher sensitivity than any individual server. ACE does not have obvious advantage over 3D-Jury-Single. Note that for 3D-Jury-All, when more servers are included, its performance increases a lot.

4.2. *Specificity*

In addition to the sensitivity of servers, the specificity is also important for high-throughput automated structure prediction servers. High sensitivity and specificity are desired goals but it is hard to achieve the two goals at the same time. We used the method applied by CAFASP3 to calculate the specificity of a server. The specificity is calculated according to the following procedures: (1) Rank the models by the confidence scores (SVM outputs). Note that only the top one model for each target is considered here; (2) Count the number of correct predictions before the first K false positives $TP(K)$; (3) Calculate the average of $TP(K)$, K=1, 2, $\cdots$, 5 as the specificity of the server. Here a correct model is defined as a model which has at least 40 C_α atoms that can be superimposed to the native structure within 3.0 $\dot{A}$ by using the MaxSub program.

Following the above steps, we calculated the specificity when there were three component servers. Here we also used the four fold cross validation. The results are listed in Table 5. Comparisons of specificity with 3D-Jury, PCON and individual servers are shown

Table 5. Specificity of ACE obtained with three component servers.

Training data set	LiveBench5	LiveBench6	LiveBench7	LiveBench 8
	78	98	115	148
LiveBench 5	—–	18.20	24.00	59.80
LiveBench 6	18.00	—–	24.00	59.80
LiveBench 7	18.00	19.00	—–	59.80
LiveBench 8	18.00	19.00	24.00	—–
Average	18.00	18.73	24.00	59.80

Table 6. Specificity comparison between ACE and its three component servers and other meta servers.

Training data set	LiveBench5	LiveBench6	LiveBench7	LiveBench 8
	78	98	115	148
FFA3	18.00	17.00	23.00	56.60
3DPS	15.80	16.80	20.40	57.00
FUG2	18.00	16.60	17.60	57.79
PCON	16.00	19.00	22.00	—–
3D-Jury-All	12.00	15.20	16.00	54.00
3D-Jury-Single	15.40	15.60	17.80	59.60
ACE	18.00	18.73	24.00	59.80

Table 7. Specificity of ACE obtained with six component servers.

Training data set	LiveBench5	LiveBench6	LiveBench7	LiveBench8
	78	98	115	148
LiveBench 5	—–	14.80	20.60	59.60
LiveBench 6	15.00	—–	21.20	59.40
LiveBench 7	15.00	15.60	—–	59.60
LiveBench 8	15.00	15.60	21.20	—–
Average	15.00	15.33	21.00	59.53

in Table 6. We can see that the specificity of ACE is significantly higher than that of 3D-Jury-All and 3D-Jury-Single. Also, the specificity of ACE is higher than any individual server and PCON. We also calculated the specificity of our meta server using six individual servers as shown in Table 7 and 8. As shown in these two tables, when six servers are used, the specificity of ACE drops as the sensitivity does. Even though the specificity is

Table 8. Specificity comparison between ACE and its six component servers and other meta servers.

Training data set	LiveBench5	LiveBench6	LiveBench7	LiveBench 8
	78	98	115	148
FFA3	18.00	17.00	23.00	56.60
3DPS	15.80	16.70	20.40	57.00
FUG2	18.00	16.60	17.60	56.20
INBG	18.00	19.00	24.00	55.79
SFPP	18.00	18.73	24.00	59.80
MGTH	18.00	18.73	24.00	59.80
3D-Jury-All	16.40	14.8	18.20	58.60
3D-Jury-Single	15.00	15.20	21.20	58.80
ACE	15.00	15.33	21.00	59.53

still better than 3D-Jury-All, it is no longer higher than that of any individual server.

5. Conclusion and Future Work

In this study, we have presented a SVM regression approach to build a protein fold recognition meta server ACE (Alignment by Consensus Estimator). ACE extracts features of each protein structure model through structural comparisons and predicts the model quality using the SVM regression. All the structural models generated by individual servers are ranked, based on the predicted model quality. Testing experiments were done with the LiveBench data. Experimental results show that our meta server is more sensitive and specific than individual servers, and slightly better than meta servers 3D-Jury and PCON, when not many individual servers are available for consensus. This feature is very desirable since collecting prediction results of many servers is not a trivial task. There are not many structure prediction servers that provide unlimited and consistent service to the community. ACE is running in CAPS6 and CAFASP4 right now for further testing. A remaining problem is how to find out the best combination of individual servers to produce the best prediction possible. This topic has not been studied before in the community and is our future research topic.

6. Acknowledgement

The authors would like to thank the developers of all the servers FFA3, SFPP, 3DPS, FUG2, INBG, MGTH, PCON and 3D-Jury, and the developers of LiveBench, which provides an excellent platform for benchmarking a new server. Our research is supported by the National Science and Engineering Research Council of Canada, CITO's Champion of Innovation Program, the Killam Fellowship, Canada Research Chair Program, and the Human Frontier Science Program.

References

1. J. Moult, T. Hubbard, F. Fidelis, and J. Pedersen. Critical assessment of methods on protein structure prediction (CASP)-round III. *Proteins: Structure, Function and Genetics*, 37(S3):2–6, December 1999.
2. J. Moult, F. Fidelis, A. Zemla, and T. Hubbard. Critical assessment of methods on protein structure prediction (CASP)-round IV. *Proteins: Structure, Function and Genetics*, 45(S5):2–7, December 2001.
3. J. Moult, F. Fidelis, A. Zemla, and T. Hubbard. Critical assessment of methods on protein structure prediction (CASP)-round V. *Proteins: Structure, Function and Genetics*, 53(S6):334–339, October 2003.
4. J.M. Bujnicki, A. Elofsson, D. Fischer, and L. Rychlewski. Livebench-2: Large-scale automated evaluation of protein structure prediction servers. *Proteins: Structure, Function and Genetics*, 45:184–191, 2001.
5. L. Rychlewski, D. Fischer, and A. Elofsson. Livebench-6: Large-scale evaluation of protein structure prediction servers. *Proteins: Structure, Function and Genetics*, 53:542–547, 2003.
6. D. Fischer, L. Rychlewski, R.L. Dunbrack, A.R. Ortiz, and A. Elofsson. CAFASP3: The third critical assessment of fully automated structure prediction methods suppl. *Proteins: Structure, Function and Genetics*, S6(53):503–516, October 2003.

7. D. Fischer, A. Elofsson, and L. Rychlewski. The 2000 olympic games of protein structure prediction. *Protein Engineering*, 13(10):667–670, October 2000.
8. J. Lundström, L. Rychlewski, J. Bunnicki, and A. Elofsson. PCONS: A neural-netwrok-based consensus predictor that improves fold recognition. *Protein Science*, 10:2354–2362, 2001.
9. K. Ginalski, A. Elofsson, D. Fischer, and L. Rychlewski. 3D-Jury: a simple approach to improve protein structure predictions. *Bioinformatics*, 19(8):1015–1018, 2003.
10. B. Wallner, H. Fang, and A. Elofsson. Automatic consensus-based fold recognition using pcons, proq, and pmodeller. *Proteins: Structure, Function and Genetics*, pages 534–541, 2003.
11. D. Fischer. 3DS3 and 3DS5: 3D-SHOTGUN meta-predictors in CAFASP3. *Proteins: Structure, Function and Genetics*, 53:517–523, 2003.
12. I. Sasson and D. Fischer. Modeling three-dimensional protein structures for CASP5 using the 3D-SHOTGUN meta-predictors. *Proteins: Structure, Function and Genetics*, 53:389–394, 2003.
13. D. Fischer. Hybrid fold recognition: Combining sequence derived properties with evolutionary information. pages 119–130, Hawaii, 2000. Biocomputing: Proceedings of the 2000 Pacific Symposium, World Scientific Publishing Co.
14. L.A. Kelley, R.M. MacCallum, and M.J.E. Sternberg. Enhanced genome annotation using structural profiles in the program 3D-PSSM. *Journal of Molecular Biology*, 299(2):499–520, 2000.
15. B. Wallner and A. Elofsson. Can correct protein models be identified ? *Protein Science*, 12(5):1073–1086, 2003.
16. N. Siew, A. Elofsson, L. Rychlewski, and D. Fischer. Maxsub: An automated measure for the assessment of protein structure prediction quality. *Bioinformatics*, 16(9):776–785, 2000.
17. V.N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.
18. Alex J. Smola and Bernhard Schölkopf. A tutorial on support vector regression. Technical report, October 1998.
19. T. Joachims. *Making Large-scale SVM Learning Practical*. MIT Press, 1999.
20. R.A. Laskowski, M.W. MacArthur, D.S. Moss, and J.M. Thornton. PROCHECK: a program to check the stereochemical quality of protein structures. *Journal of Applied Crystography*, 26:283–291, 1993.
21. G. Vriend. WHATIF: a molecular modeling and drug design program. *Journal of Molecular Graphics*, 8:52–56, 1990.
22. L. Rychlewski, L. Jaroszewski, W. Li, and A. Godzik. Comparison of sequence profiles: strategies for structural predictions using sequence information. *Protein Science*, (9):232–241, 2000.
23. L. Jaroszewski, L. Rychlewski, and A. Godzik. Improving the quality of twilight-zone alignments. *Protein Science*, (9):1487–1496, 2000.
24. J. Shi, L. B. Tom, and M. Kenji. FUGUE: Sequence-structure homology recognition using environment-specific substitution tables and structure-dependent gap penalties. *Journal of Molecular Biology*, 310:243–257, 2001.
25. J. Gough, K. Karplus, R. Hughey, and C. Chothia. Assignment of homology to genome sequences using a library of hidden markov models that represent all proteins of known structure. *Journal of Molecular Biology*, 313(4):903–919, 2001.
26. L.J. McGuffin and D.T. Jones. Improvement of the GenTHREADER method for genomic fold recognition. *Bioinformatics*, (19):874–881, 2003.