The final version of record is available at http://dx.doi.org/10.1109/SECON.2014.6950649

Evaluation of Semi-supervised Learning for Classification of Protein Crystallization Imagery

[1]Madhav Sigdel, [1]Imren Dinç, [1]Semih Dinç, [1]Madhu S. Sigdel, [2]Marc L. Pusey, [1]Ramazan S. Aygün
[1]Department of Computer Science
[1]University of Alabama in Huntsville
[1]Huntsville, Alabama 35899, United States
[2]iXpressGenes, Inc., 601 Genome Way, Huntsville, Alabama 35806, United States
Email: [1]{ms0023, id0002, sd0016, mss0025, aygunr}@uah.edu, [2]marc.pusey@iXpressGenes.com

***Abstract*—In this paper, we investigate the performance of two wrapper methods for semi-supervised learning algorithms for classification of protein crystallization images with limited labeled images. Firstly, we evaluate the performance of semi-supervised approach using self-training with naïve Bayesian (NB) and sequential minimum optimization (SMO) as the base classifiers. The confidence values returned by these classifiers are used to select high confident predictions to be used for self-training. Secondly, we analyze the performance of Yet Another Two Stage Idea (YATSI) semi-supervised learning using NB, SMO, multilayer perceptron (MLP), J48 and random forest (RF) classifiers. These results are compared with the basic supervised learning using the same training sets. We perform our experiments on a dataset consisting of 2250 protein crystallization images for different proportions of training and test data. Our results indicate that NB and SMO using both self-training and YATSI semi-supervised approaches improve accuracies with respect to supervised learning. On the other hand, MLP, J48 and RF perform better using basic supervised learning. Overall, random forest classifier yields the best accuracy with supervised learning for our dataset.**

I. Introduction

In supervised learning, labeled data are used to train a prediction model. In general, supervised learning algorithms perform well only when there is sufficiently large number of training data. For cases where the proportion of labeled data instances is small compared to the unlabeled instances, researchers have proposed semi-supervised learning. Semi-supervised learning targets the common situation where the labeled data is very low and the objective of this technique is to use the unlabeled data to create better learning models.

The situation of having limited labeled data suits very well to the protein crystallization image classification problem. High throughput methods have been developed in recent years trying to identify the best conditions to crystallize proteins [1]. The images are scanned periodically to determine the state change or the possibility of forming crystals. With large number of images being captured, it is necessary to have a reliable classification system to distinguish the crystallization states each image belongs to. It is very tedious to manually label the protein images by an expert since the protein crystal growth rarely happens. We would like to analyze how much the classification accuracy can be improved by using the limited labeled data and then processing the unlabeled data using trained models.

Several research studies have been described in the literature for protein crystallization classification problem using a variety of classification algorithms such as support vector machines (SVMs), decision trees, neural networks, boosting, and random forest [2]. Alternatively, combination of multiple classifiers has also been studied in the literature [3]. Nevertheless, the reported accuracy has not been very reliable, and therefore the classification of crystallization images still remains an important problem. To improve the performance of the classifiers, there has been a trend to increase the size of training data. Cumba et al. built their model based on 165,351 hand-scored images and used random forest for classification [4]. Likewise, Po and Laine used a neural network classifier on a training dataset consisting 79,632 images [5]. Being able to create reliable classifiers using limited labeled data can save a lot of time and effort for expert labeling.

Various semi-supervised techniques have been proposed in the literature. Broadly, there are two types of semi-supervised classification techniques. First, there are generic or wrapper-based techniques which are formulated on top of some supervised techniques. The wrapper-based techniques allow the possibility of using several supervised classification techniques as the base classifier. Self-training is one of the simplest semi-supervised technique where a learner keeps on labeling unlabeled examples and retrain itself on an increased training set. Yet Another Two Stage Idea (YATSI) introduced by Driessens et al. is another wrapper based semi-supervised learning [6]. The second group of semi-supervised classification techniques are the non-generic semi-supervised learning techniques aiming to improve the learning models by taking advantage of the unlabeled data. Examples of non-generic ones include transductive support vector machine (TSVM), semisupervised SVM (S3VM) and their variants, Laplacian SVM, etc. [7].

Semi-supervised techniques have been applied and evaluated for various applications such as software fault detection [8], text classification [9], spam email detection [10], quantitative structure-activity modeling [11], etc. There have been conflicting views about the usability of semi-supervised learning techniques. While some studies have shown this technique to be promising, other studies have shown that the use of unlabeled data does not necessarily improve the performance of the classifier [8], [7]. We try to investigate this scenario for protein crystallization image classification problem.

This study investigates the performance of supervised

versus semi-supervised algorithms for the protein image classification problem with limited labeled data. Firstly, we use sequential minimal optimization (SMO) and naïve Bayesian (NB) to evaluate semi-supervised learning using self-training. Secondly, we evaluate the performance of 5 supervised classification techniques NB, J48, multilayer perceptron neural network (MLP), SMO and random forest (RF)). We use these classification techniques with YATSI to evaluate the performance of semi-supervised learning. We perform our experiments for different proportions of training and test data on a dataset consisting 2250 images with 67% non-crystals, 18% likely leads and 15% crystals.

This paper is arranged as follows. The following section presents background on semi-supervised learning algorithms. Section 3 describes the image categories for protein crystallization image classification. Section 4 provides the features used for the classification. Experimental results and discussion are provided in Section 5. The last section concludes the paper with future work.

II. Semi-supervised learning

In supervised learning, the objective is to derive a prediction model or classification function for the unseen (unlabeled) data. The prediction model is developed on the basis of training data (labeled data) only. Semi-supervised learning aims in combining the labeled data and unlabeled data to create better learners. The general assumption in these algorithms is that data points in a high density region are likely to have same classes and the decision boundary lies in low density regions [12], [7]. The idea is to use labeled data to generate initial training model and determine initial predictions (pre-labels) for the test data. If the labeled and pre-labeled data is combined and retrained, the initial decision boundary can shift which will hopefully improve the performance. Various semi-supervised learning methods have been proposed and shown to be promising [13]. In this study, we investigate the classification performances using two wrapper based semi-supervised learning techniques - self-training and YATSI.

A. Self-training

Self-training is a semi-supervised learning algorithm in which a learner keeps on labeling unlabeled examples and retrains itself on an enlarged labeled training set [14]. This is a generic technique and any supervised technique can be used as the base classifier. One problem with self-training is that the performance is degraded when mistakes reinforce themselves. There are some variants of self-training that try to reduce the number of wrongly predicted instances while re-training. One method uses only high confident predictions from the initial prediction model for retraining. For this method, the classification algorithm is required to generate a confidence value or a probability estimate for the prediction. This confidence value can be used to filter additional pre-labeled data for re-training.

B. YATSI (Yet Another Semi-supervised Idea)

Yet Another Two Stage Idea (YATSI) [6] is a semi-supervised classification algorithm consisting two stages. It is built on top of any supervised classification algorithm and a nearest neighborhood algorithm. In the first stage, prediction model is generated on training set using a supervised classifier and the predictions for unlabeled instances are determined. After the predictions, these previously unlabeled instances are called pre-labeled instances. In the second stage, the nearest neighborhood algorithm is applied using the initial training instances and the pre-labeled instances to determine the actual predictions for unlabeled instances. Besides the initial classification algorithm, the nearest neighborhood algorithm and weight factor corresponding to the trust of correctness for the pre-labeled dataset can be adjusted.

III. Image Categories

In this study, we consider three image categories for protein crystallization images. Description of each of these categories is provided next.

Non-crystals - This category consists of images under the following protein crystallization phases: clear drop (initial state of the crystallization process), phase separation, and regular precipitates. This category indicates that these images do not have crystals. Fig. 1 [a-c] show some sample images under this category.

Likely leads - This category consists of images corresponding to likely lead conditions, and hence, can be a good starting point for optimizing the crystallization conditions. Birefringent precipitate or microcrystals fall under this category. It also includes images with high intensity regions without any clear indication of presence of crystals. Such images can occur due to improper focusing, camera lighting, etc. Since high intensity might indicate the presence of crystals, these images should be reviewed by an expert. Fig. 1 [d-f] provide some sample images under this category.

Crystals - This category consists of images consisting crystals. Crystals can have different shapes and sizes like needle, spherulites, plates, or 3D crystals. Fig. 1[g-i] show some sample images under this category.

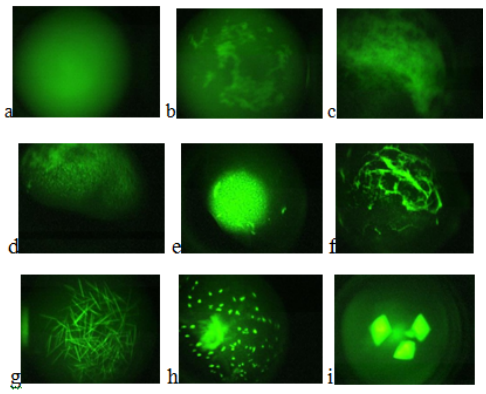

Fig. 1. Sample protein crystallization images: [a-c] Non-crystals [d-f] Likely leads [g-i] Crystals

IV. Feature extraction

For feature extraction, we follow the image processing steps as described in our previous work [2]. Initial pre-processing steps include image resizing to 320x240 pixels, median filtering and applying three dynamic image thresholding methods. Connected component labeling is done on

the thresholded images and corresponding blob features are extracted. From each binary image, we extract 6 intensity related features and 9 blob related features. Therefore, we extract a total of 3*(6+9) = 45 features per image. Short description of each of these features is listed below.

1) Intensity features
   a) Threshold intensity (τ)
   b) No of white pixels in the binary image (N_f)
   c) Average image intensity in the foreground region (μ_f)
   d) Standard deviation of intensity in the foreground region (σ_f)
   e) Average image intensity in the background region (μ_b)
   f) Standard deviation of intensity in the background region (σ_b)
2) Region (Blob) features
   a) No of blobs (η)
   b) Area of the largest blob (a_1)
   c) The largest blob fullness (f_1)
   d) The largest blob boundary pixel count(N_b^1)
   e) The largest blob boundary uniformity measure (u_1^1)
   f) The largest blob uniformity measure (u_2^1)
   g) The largest blob measure of symmetry (ζ_1)
   h) Average area of the top 5 largest blobs excluding largest blob (a_{avg})
   i) Average fullness of the top 5 largest blobs excluding largest blob (f_{avg})

V. Experimental Results

Our experimental dataset consists of 2250 expert labeled images with 67% non-crystals, 18% likely leads and 15% crystals. Most crystallization images belong to non-crystal category. Hence, we included more crystal images in our dataset to reduce the class imbalance in the training and to include all kinds of crystals. In this study, we consider two classification problems (2-class and 3-class) for the protein crystallization image classification. For the 2-class problem, images in likely leads and crystals categories are grouped together to form a single class called *likely crystals*. The two classes, non-crystals and likely crystals are represented as 67% and 33% in the dataset. 3-class classification is performed using the original image categories.

We evaluate the classification performances of two generic semi-supervised algorithms - self-training and Yet Another Two Stage Idea (YATSI) using different base classifiers. Our experiments assume limited labeled data availability. We evaluate the performance of selected classifiers for 5 different training sizes (1%, 2%, 5%, 10%, and 20%) of the labeled data. In each of these cases, remaining portions of the data (99%, 98%, 95%, 90%, and 80%) are used for testing (i.e., considered as unlabeled data). For the supervised learning algorithms, we use classifiers from WEKA project (www.cs.waikato.ac.nz/ ml/weka), which are implemented in Java [15]. For the YATSI implementation, we use collective classification package available from MARSDEN project (http://www.cs.waikato.ac.nz/ fracpete/projects/collective-classification/). Programs are written and tested in Java programming language in Eclipse environment.

A. Performance comparison with self-training

Self-training is an iterative method where a training model is retrained using the high confidence prediction from the previous iteration to find the actual predictions for unlabeled data. This is also a wrapper based semi-supervised approach. Besides predicting the label for an instance, the classifiers should output a value for the confidence on that prediction. Hence, not all supervised classifiers can be used as the base classifier with this approach. In our experiments, we use naïve Bayesian (NB) and sequential minimal optimization (SMO) as the base classifiers for self-training. Since this is an iterative method, we can proceed the self-training many times. We only perform a single iteration. One parameter that can be adjusted to limit the pre-labeled data into re-training is the threshold for the minimum confidence (c) for prediction. We evaluate our experiments for 3 different values of c (0.8, 0.9 and 0.95) for minimum confidence.

TABLE I. 2-Class classification performance with self-training for naïve Bayes and SMO classifiers

Classifier	Training size				
	1%	2%	5%	10%	20%
NB	84.82	**87.38**	87.84	87.79	87.88
Self-NB (c=0.8)	85.48	87.27	87.96	87.84	**87.89**
Self-NB (c=0.9)	85.57	87.28	88.02	87.81	87.86
Self-NB (c=0.95)	**85.63**	87.28	**88.02**	**87.86**	87.85
SMO	82.56	88.39	88.53	88.59	89.02
Self-SMO (c=0.8)	83.01	89.13	89.11	89.27	89.20
Self-SMO (c=0.9)	83.14	89.28	89.18	89.37	89.32
Self-SMO (c=0.95)	**83.22**	**89.53**	**89.43**	**89.58**	**89.48**

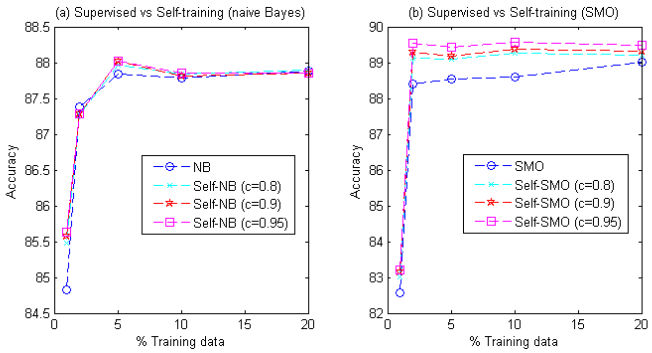

Fig. 2. Supervised vs Self-training performance comparison for a) Naïve Bayesian b) Sequential minimum optimization (SMO)

Table I shows the experimental results with self-training for 2 classifiers with different values of c. Self-NB and Self-SMO correspond to the performances with self-training for NB and SMO classifiers respectively. The value for c in the parentheses refer to the minimum confidence used to select the pre-labeled instances for re-training. Fig. 2(a) and Fig. 2(b) provide the performance comparison plot for the two classifiers. Our results indicate that both NB and SMO classifiers using self-training improve accuracies with respect to supervised learning. For NB, the performances with self-training is improved very slightly. For SMO, the accuracies

TABLE II. 2-CLASS CLASSIFICATION PERFORMANCE FOR DIFFERENT CLASSIFIERS

Classifier	Training size				
	1%	2%	5%	10%	20%
NB	84.82	87.38	87.84	87.79	87.88
Y-NB (K=10)	86.01	88.59	89.64	89.90	**91.19**
Y-NB (K=20)	86.74	88.65	90.15	**89.91**	90.62
Y-NB (K=30)	**86.78**	**88.76**	**90.19**	89.62	90.28
MLP	**82.86**	88.26	**90.95**	**93.60**	**95.13**
Y-MLP (K=10)	82.26	88.06	89.99	92.09	92.94
Y-MLP (K=20)	82.48	88.22	89.88	92.58	92.86
Y-MLP (K=30)	82.57	**88.31**	90.07	92.46	92.87
SMO	82.56	**88.39**	88.53	88.59	89.02
Y-SMO (K=10)	83.57	88.10	90.41	**92.09**	**92.98**
Y-SMO (K=20)	83.83	87.94	90.34	91.90	92.97
Y-SMO (K=30)	**83.93**	88.10	**90.60**	91.66	92.81
J48	**89.06**	**88.62**	**91.71**	92.72	**94.49**
Y-J48 (K=10)	88.40	88.50	90.76	91.92	93.02
Y-J48 (K=20)	88.18	88.44	90.52	**92.54**	93.08
Y-J48 (K=30)	87.87	88.52	90.97	92.39	92.96
RF	84.41	**88.77**	**92.20**	**94.21**	**95.86**
Y-RF (K=10)	84.31	88.64	90.92	92.08	92.99
Y-RF (K=20)	**84.67**	88.31	90.62	92.48	92.93
Y-RF (K=30)	84.54	88.34	90.96	92.36	93.01

with self-training is improved by around 1% over the accuracy with SMO alone. For both the classifiers, the accuracy is usually improved for higher value of c. Although the accuracies are improved by using self-training, the time complexity of the method is significantly high.

B. Performance comparison with YATSI

YATSI is a two stage semi-supervised learning algorithm. Firstly, the labeled data is used to form the prediction model using a supervised classifier. This model is used to get pre-labels for the test instances. Secondly, K-neighborhood algorithm is applied on the combined labeled and pre-labeled instances to predict actual labels for the test (pre-labeled) instances. In this study, we consider the following five supervised classification techniques - naïve Bayesian (NB), sequential minimum optimization (SMO), J48, multilayer perceptron (MLP) and random forest (RF) and their five YATSI semi-supervised learning counterparts - YATSI with naïve Bayesian (Y-NB), YATSI with SMO (Y-SMO), YATSI with J48 (Y-J48), YATSI with MLP (Y-MLP) and YATSI with random forest (Y-RF).

For all the supervised classifiers, we apply the default settings provided in Weka [15]. For YATSI classifiers, we test K-nearest neighbors (K_{nn}) with 10, 20 and 30 neighbors. For the YATSI classifiers, the weighting factor for pre-labeled data (F) is set to 1.

Table II provides the classification results for the 2-class problem for 5 supervised classifiers and corresponding YATSI classifiers. In the classifier column, for YATSI classifiers, the value for K_{nn} is given in parenthesis. In each column, the largest value is highlighted to indicate the best classifier for the given training size. Fig. 3 shows the performance comparison graphs for each classification method for 2-class problem.

Performance of Classifiers: Our initial observation is that naïve Bayesian and SMO classifiers benefit from YATSI. The performance of these classifiers improved with YATSI. Naïve Bayesian classifier with YATSI improved its accuracy by 1.96 % for 1% training size and by 2.4% for 20% training size using 30 neighbors. For naïve Bayesian, performance improved with semi-supervised approach whatever the portion of training data. This can be visualized in Fig. 3(a). Similarly, SMO with YATSI approach improved its accuracy by 1.37% for 1% training size and 3.79% for 20% training size. Fig. 3(c) shows that the YATSI-SMO approach provides significant improvement over SMO for all training sizes.

Our results indicate that MLP, J48, and random forest classifiers do not benefit from YATSI method. The performance of random forest with YATSI is almost 2.85% down the supervised one for 20% training whereas it is almost similar for 1% training set.

In general, the performances of the YATSI classifiers improved with higher values for K_{nn} up to certain value. However, for higher values, the variation in performance was not consistent. A good choice for K_{nn} is critical for the performance of YATSI classifiers. For real deployment of the classifiers, the value for K_{nn} can be determined by optimizing the performance on a validation set.

As the size of training data increases, the performance is improved for all classifiers. This is usually true for semi-supervised approach as well. This improvement comes at the cost of extra labeled data. Hence, this should be analyzed separately.

In Fig. 3(f), we plot the graphs combining the best conditions for each of the five classifiers considered. This allows us to compare the performances of all classifiers in a single figure. From the figure, we can observe that supervised learning using random forest provided the best performance on our dataset.

Performance over 3-class classification: We also investigated the supervised versus YATSI approach for 3-class problem. Table III provides the classification results for the 3-class problem and Fig. 4 shows the corresponding performance graphs for each classification method. Our results show that the results for 2-class and 3-class problem are almost consistent. Similar to the results for 2-class problem, the performances of naïve Bayesian and SMO classifiers are improved by the YATSI approach. Naïve Bayesian classifier with YATSI improved its accuracy by 1.95 % for 1% training size and by 2.45% for 20% training size using 30 neighbors. Similarly, SMO-YATSI improved by 0.66% for 1% training size and by 4.46% for 20% training size. Overall improvement by the YATSI approach for the two classifiers over supervised approach can be visualized in Fig. 4 (a) and Fig. 4 (c).

As in the results for 2-class problem, classifiers J48, MLP and random forest did not benefit from the semi-supervised approach. The combined plot with the best classifiers for 3-class classification is drawn in Fig. 4(f) which shows that supervised learning using random forest gives the best performance over all other classifiers.

C. Summary and Discussion

The pre-labeled data may have incorrect labels. Self-learning classifier used the pre-labeled having high confidence. In YATSI, the incorrect labels are expected to be corrected by K-nearest neighborhood classifier. Therefore, YATSI performs better than self-learning for naïve Bayesian and SMO

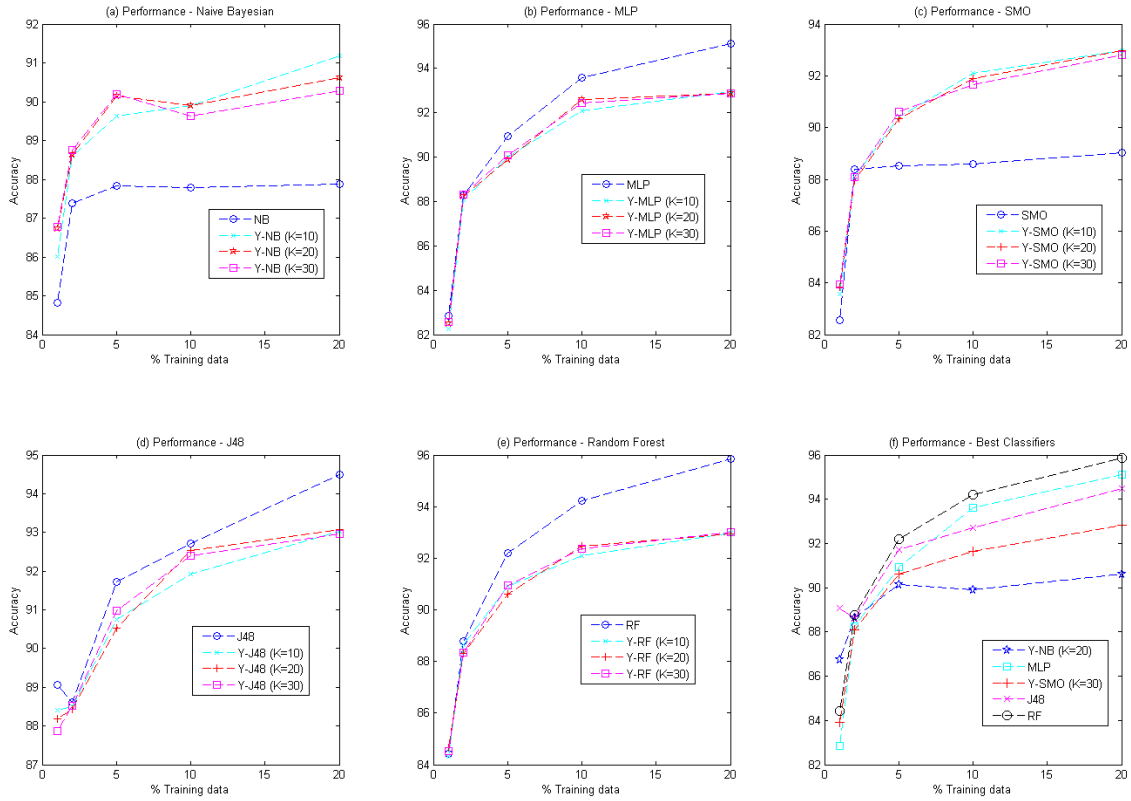

Fig. 3. Supervised vs YATSI semi-supervised performance comparison for 2-class classification a) Naïve Bayesian b) Multilayer perceptron (MLP) c) Sequential minimal optimization (SMO) d) J48 e) Random forest f) Best classifier for each of the five classifiers

TABLE III. 3-CLASS CLASSIFICATION PERFORMANCE FOR DIFFERENT CLASSIFIERS

Classifier	Training size				
	1%	2%	5%	10%	20%
NB	73.92	78.39	80.19	81.41	81.26
Y-NB (K=10)	75.11	79.88	82.38	**83.83**	**85.16**
Y-NB (K=20)	75.61	**79.95**	82.01	83.40	84.16
Y-NB (K=30)	**75.87**	79.72	**82.43**	83.22	83.71
MLP	76.57	**80.61**	**84.90**	**87.21**	**90.45**
Y-MLP (K=10)	76.96	80.27	84.13	85.58	87.13
Y-MLP (K=20)	77.16	79.96	83.08	86.00	86.84
Y-MLP (K=30)	**77.23**	79.68	83.06	86.08	86.80
SMO	74.29	77.27	79.05	80.74	82.53
Y-SMO (K=10)	**77.16**	**80.42**	**83.52**	86.09	**87.42**
Y-SMO (K=20)	76.98	80.30	82.70	**86.22**	87.22
Y-SMO (K=30)	77.04	79.94	83.10	85.93	86.99
J48	75.61	77.78	83.62	**86.08**	**89.13**
Y-J48 (K=10)	75.50	78.59	**83.93**	85.58	87.07
Y-J48 (K=20)	75.56	**78.61**	83.28	85.87	86.64
Y-J48 (K=30)	**75.76**	78.55	83.55	85.87	86.49
RF	76.92	80.77	**85.19**	**88.18**	**91.07**
Y-RF (K=10)	77.10	**80.82**	84.07	85.70	87.33
Y-RF (K=20)	**77.24**	80.58	83.15	86.17	86.83
Y-RF (K=30)	77.22	80.20	83.45	86.24	86.74

classifiers, since these classifiers are benefiting from these corrections. However, for other classifiers, RF, J48, and MLP, these pre-labeled data are just noise to the system. In other words, the addition of pre-labeled data misguides the inference for these classifiers. These classifiers would rather prefer to work on accurately labeled data. We should note that random forest with supervised learning outperforms others.

This may lead to the following discussion. If the base classifier with supervised learning works comparatively well for naïve Bayesian and SMO classifiers, they may be chosen as the base classifiers and semi-supervised learning might be beneficial. On the other hand, if a classifier, such as RF, performs well as a base classifier, there is no need to try semi-supervised learning since the pre-labeled data is not beneficial for RF.

VI. CONCLUSION AND FUTURE WORK

In this paper, we investigated the performance of two wrapper methods for semi-supervised learning algorithms for classification of protein crystallization images. Our motivation behind this work was to apply semi-supervised approach and see if we get reasonable performance with limited labeled data. We compared the performances of semi-supervised classification techniques using self-training and YATSI approach. Our results show that naïve Bayesian (NB) and sequential minimal optimization (SMO) classifiers benefit from both the self-training and YATSI semi-supervised approach on our dataset. However, classifiers J48, multilayer perceptron (MLP) and random forest (RF) did not show improvement by applying

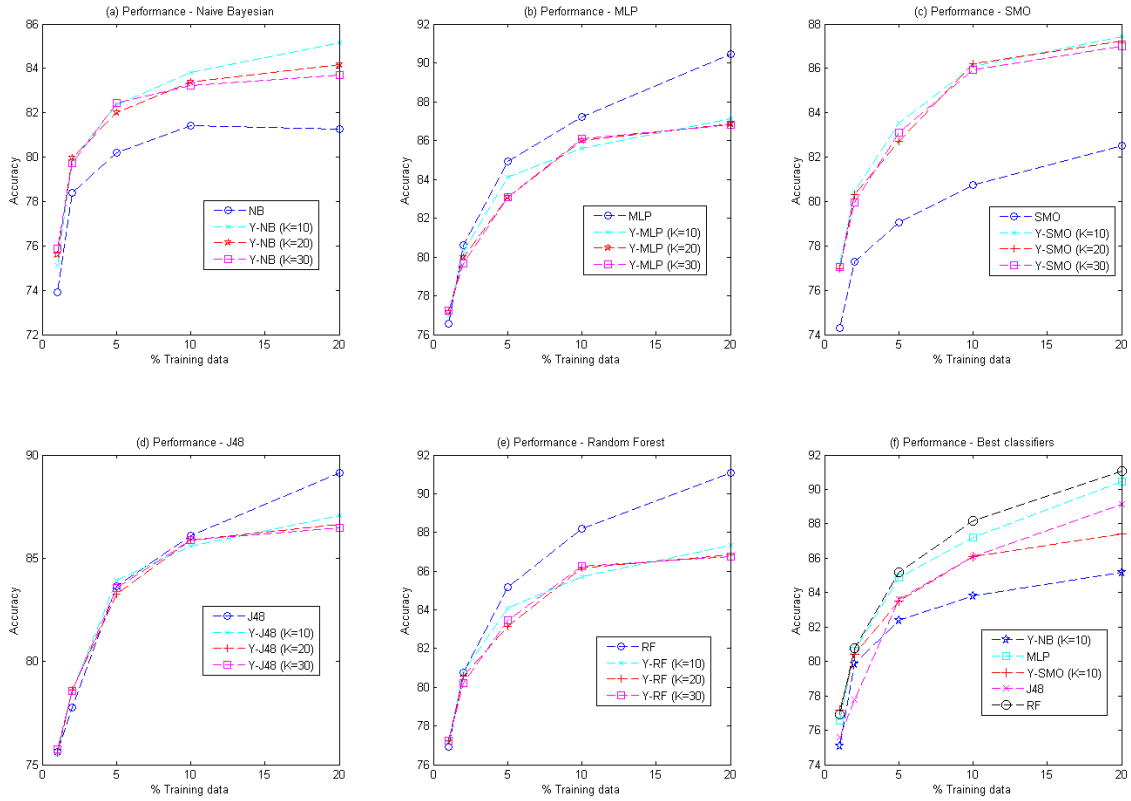

Fig. 4. Supervised vs YATSI semi-supervised performance comparison for 3-class classification a) Naïve Bayesian b) Multilayer perceptron (MLP) c) Sequential minimal optimization (SMO) d) J48 e) Random forest f) Best classifier for each of the five classifiers

semi-supervised approach. In overall, random forest provided the best performance on our dataset.

As further work, we would like to investigate active learning in combination with semi-supervised learning to improve the classification performance.

VII. Acknowledgement

This research was supported by National Institutes of Health (GM090453) grant.

References

[1] M. L. Pusey, Z.-J. Liu, W. Tempel, J. Praissman, D. Lin, B.-C. Wang, J. A. Gavira, and J. D. Ng, "Life in the fast lane for protein crystallization and x-ray crystallography," *Progress in Biophysics and Molecular Biology*, vol. 88, no. 3, pp. 359 – 386, 2005.

[2] M. Sigdel, M. L. Pusey, and R. S. Aygun, "Real-time protein crystallization image acquisition and classification system," *Crystal Growth Design*, vol. 13, no. 7, pp. 2728–2736, 2013.

[3] K. Saitoh, K. Kawabata, and H. Asama, "Design of classifier to automate the evaluation of protein crystallization states," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*. IEEE, 2006, pp. 1800–1805.

[4] C. A. Cumbaa and I. Jurisica, "Protein crystallization analysis on the world community grid," *J Struct Funct Genomics*, vol. 11, no. 1, pp. 61–9.

[5] M. Po and A. Laine, "Leveraging genetic algorithm and neural network in automated protein crystal recognition." in *IEEE Eng Med Biol Soc.*, 2008.

[6] K. Driessens, P. Reutemann, B. Pfahringer, and C. Leschi, "Using weighted nearest neighbor to benefit from unlabeled data," in *PAKDD*, mar 2006, pp. 60–69.

[7] Y. Wang and S. Chen, "Safety-aware semi-supervised classification," 2013.

[8] C. Catal and B. Diri, "Unlabelled extra data do not always mean extra performance for semi-supervised fault prediction," *Expert Systems*, vol. 26, no. 5.

[9] T. Joachims, "Transductive inference for text classification using support vector machines," in *ICML*, vol. 99, 1999, pp. 200–209.

[10] B. Pfahringer, "A semi-supervised spam mail detector," 2006.

[11] J. Levatić, S. Džeroski, F. Supek, and T. Šmuc, "Semi-supervised learning for quantitative structure-activity modeling." *Informatica (03505596)*, vol. 37, no. 2, 2013.

[12] X. Zhu, "Semi-supervised learning literature survey," 2006.

[13] X. Zhu, J. Lafferty, and Z. Ghahramani, "Combining active learning and semi-supervised learning using gaussian fields and harmonic functions," in *ICML 2003 workshop*.

[14] M. Seeger, "Learning with labeled and unlabeled data," technical report, University of Edinburgh, Tech. Rep., 2001.

[15] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *ACM SIGKDD Explorations Newsletter*, vol. 11, no. 1, pp. 10–18, 2009.