CS223B Project Final Report

Classification of Protein Crystallization Imagery

Samuel Cheng, Shaohua Sun and Xiaoqing Zhu
{samc, shaohuas, zhuxq}@stanford.edu

Abstract

We investigate the problem of automatic classification of protein crystallization images. A number of new methods are examined for their improvement of the classification result. For each image, the level-set method is used for boundary detection; texture features are extracted from the Gray Level Co-occurrence Matrices (GLCM); and finally, the support vector machine(SVM) is used for classification of the extracted feature vectors. A publicly available decision-tree classifier (C5.0) is also used for comparison of performance and for automatic feature selection.

Experimental results from around 600 images are presented for the binary classification problem: separating successful trials("Crystals") from failed attempts ("Clear" or "Precipitates"). We compare the effectiveness of geometric features versus texture features, and of the decision-tree based classifiers versus the support vector machine. The best false positive rate and false negative rates are at 14.6% and 9.6% respectively, achieved by feeding both sets of features to the C5.0 classifier with boosting.

1 Introduction

One of the most tantalizing problems in modern biology is to understand the three-dimensional structures of protein molecules. Among the many techniques in use, single-crystal X-ray crystallography, using diffraction, has proven to be the most effective and therefore the most popular. A major challenge associated with crystallography is the growth of protein crystals, since the outcome is very sensitive to experimental conditions such as chemical solution, temperature, and air pressure. In order to successfully produce protein crystals suitable for X-ray diffraction, hundreds of thousands of trials may be needed.

A recent solution to the time-consuming and expensive protein crystallization process has been the use of high-throughput (HT) crystallization robots, which can be set up to output over 100,000 digital crystallization photographs per day [1]. The robots are capable of dispensing small amounts of the reacting agents and protein into a well according to a specified formula, and of periodically recording the outcomes in the form of digital photographs. The automated experimental setup allows wider range of experimental conditions to be tested for crystal growth. However, the amount of data generated by such a system is enormous. Purely manual classification of the crystallization outcome images is thus no longer feasible; automatic classification is needed to aid in final human decisions.

Many research efforts have been devoted to the classification of crystallization imagery, which has turned out to be hard even for humans. Figure 1 illustrates four typical classes of the experiment outcomes:

Clear: Failed attempt, where the drop contains no solid particles.

Precipitate: Failed attempt, where precipitates instead of crystals are formed in the solution

Hit: Close-to-success trial, some microcrystalline solid is formed in the drop, maybe among the precipitates.

Crystal: Successful trial, the drop mainly contains protein crystals of non-trivial size, which may be manually dipped out for X-ray diffraction.

Note that the appearance of precipitate outcomes can vary significantly under different conditions. (see Fig. 2) The characteristics of protein crystals also cover a wide range since difference proteins crystallize into different shapes. (see Fig. 2) In addition, the boundary between "precipitate" and "hits" images is also somewhat blurred when the outcome contains both precipitates and tiny crystals. In this case one would prefer to label the image as "Hits" for further human inspection. Consequently, in classification, the false negative rate (mistaking "Hits" for "Precipitate") is more emphasized than the false positive rate (mistaking "Precipitate" for "Crystal").

For simplification, we will only consider the binary classification problem, where the "Clear" and "Precipitate" images are labeled as "Failure" and the latter two classes are labeled as "Success".

Current automatic classification algorithms typically achieve false negatives and false positives on the order of 20% [1][2][3]. Best results are reported in [4] with 12% false negatives and 14% false positives. To facilitate the more ambitious goal of automatic analysis of the conditions for successful crystallization, better performance is needed from the automatic classification process.

In this project, we build our system upon the framework described in [5], and experiment with several new methods in the hope of improving the classification performance. For drop boundary detection, the level-set method is used. For feature extraction, texture features are calculated from the gray level coexistence matrices (GLCMs) of the pixels within the boundary of each images. For classification, either the support vector machine or an automatic decision-tree classifier is used.

The remaining part of the report is organized as follows. In Section 2 we briefly review existing methods for the crystallization imagery classification problem. In Section 3, we explain the overall structure of the system. The major components of the system: drop boundary detection, feature extraction and the classifier are discussed in Sections 3.1, 3.2 and 4. We present experimental results for the level set drop boundary detection algorithm, for comparison of different feature sets and different classifiers in Section 5.

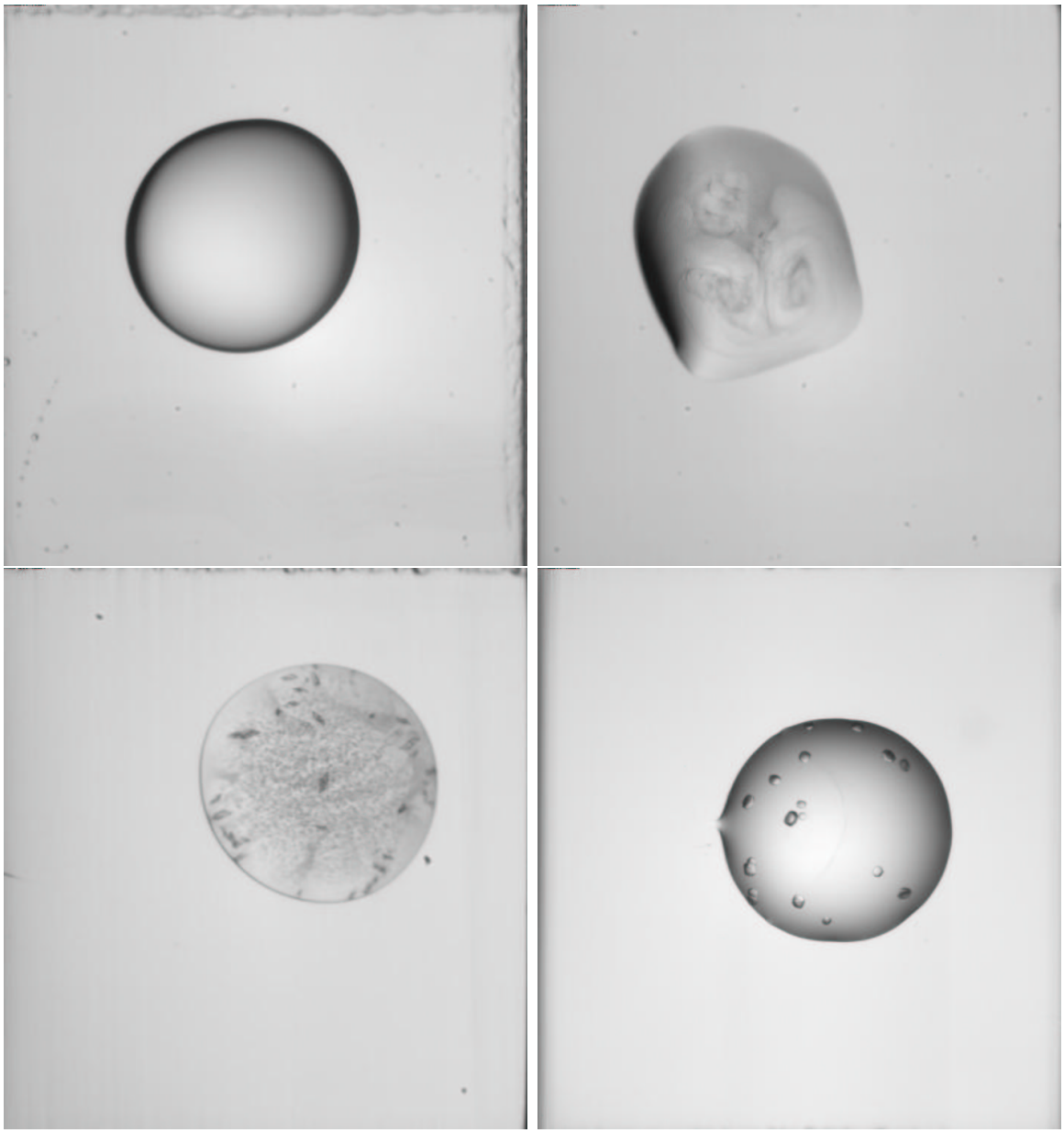

Figure 1: Sample images for four classes: Clear, Precipitate, Hit and Crystal

(a) (b) (c)

Figure 2: Sample images for different appearance of precipitate images

(a) (b) (c)

Figure 3: Sample images for different appearance of crystal images

2 Related Work

Modern robotic crystallization systems can carry out more than 10,000 trials per day, each with a unique set of chemical conditions and sequentially recorded outcomes via digital photography. Fast and automatic evaluation of the resultant data, however, is still a challenge [3]. The key issue is to distinguish successful crystallization results from unsuccessful ones, i.e., to classify the outcome images into two or more categories. This is a machine learning problem, usually with training images annotated by humans, and has been addressed by many researchers. In this section we give a brief survey of existing methods for automatic recognition and classification of crystallization imagery.

Early work by Zuk and Ward [2] uses the Hough transform to detect straight edges for crystals, but does not attempt to classify the images. A custom-built image acquisition and image processing system is reported in [1], where Jurisica et al. detected the drop boundary by fitting a conic curve and classified the images using spectral analysis. No specific error rates are reported. Instead, the correlation between extracted features and crystallization results is demonstrated by example.

In [3], Wilson proposes an algorithm to find drop boundaries using the Sobel edge detector and to detect objects with high circularity; he then classified the images into three categories based on features of edge pixels and reported accuracy rate at around 75%.

Spraggon et al. have tried the Canny edge detector and circle fitting for drop boundary detection and a self-organizing neural net for classification [6], using features related to straight lines and textures and reporting false negatives and false positives about 25%. One problem with this drop detection algorithm is that it may miss crystals outside the fitted circle.

More recently, a probabilistic graphical model is used to find the drop boundary and classification features based on correlation filters and the Radon transform (similar to Hough transform, with polar coordinate parameterization) [7]. A balanced error rate of 15% for the two-class classification: crystal-positive and crystal-negative is achieved.

The best result reported so far is in [4], where Bern et al. propose a line tracking algorithm for drop boundary detection and a decision-tree classifier with hand-crafted thresholds operating on the gradient- and geometry-related features of a selected drop. The drop boundary detection has a success ratio of 93%, whereas the classification achieves 12% false negative and 14% false positive. It is also mentioned that the current feature-detection algorithms fail to capture the difference between swirly precipitates due to convective currents and microcrystals, and that new features representing global characteristics of the images are needed.

3 System Overview

We note that current research in this classification problem is still in the trial-and-error stage, and that there is still much room for improvement in terms of performance. The choice of features are mainly based on heuristics, and in general the understanding of how well the feature vectors perform is lacking. Either an off-the-shelf classifier is used, or an ad

hoc solution is given with hand-tuning parameters. The classification results are still not good enough to support automatic processing of the data, and a vast area of algorithms for drop boundary detection, feature extraction and classification remain yet to be tested.

In our project, several new methods are implemented and evaluated, in the hope of improving the classification results. More importantly, we attempt to analyze the effectiveness of each feature, use an automatic pruning algorithm to decide on the more effective features.

The overall structure of the crystallization imagery classification system is illustrated in Fig. 4. Each image is preprocessed by the drop boundary detection algorithm. After that, texture features are extracted from the gray level coexistent matrices of the pixels within the drop boundary. These feature vectors are fed into the classifier, which has been trained off-line using human annotated training images. Additionally, the automatic decision-tree classifier C5.0 will be used to select the more effective features.

3.1 Boundary Detection

Drop segmentation is the first step in a classification system. It is important to separate the drop from the background well, so that the position of the droplet in the well is not important to the classifier. In addition, the segmenter must be conservative and not exclude significant parts of the droplet, as the rarity of crystallinity makes false negatives extremely undesirable. Finally, the segmenter must exclude the drop boundary, so that the edge of the drop does not confuse the feature detectors intended for the drop interior.

There are some problems with the currently-implemented techniques for boundary detection. Edge detection techniques must be modified in order to generate closed contours, and circle fitting fails on the 15% of droplets with irregular shape. [4] uses a parametric "snake" algorithm, which handles irregular shapes well, but sometimes follows false curves, segmenting some of the droplet into the background. In addition, sometimes it fails to find the droplet altogether. [8] demonstrates the effectiveness of level set algorithms in segmenting pollen grains from background in electron micrographs, and our experiments show that this method is robust and effective.

Level sets are particularly suited for this application, because some boundaries are oddly shaped or separated. The extra dimension of the level-set function allows it to remain well-behaved and continuous, despite discontinuities in the boundary.

The segmenter was implemented using the Multivac object-oriented framework for level set methods in two dimensions. This framework keeps track of the function ϕ, its initialization, and its propagation, allowing the user to concentrate on the functions used to initialize and evolve the boundary front.

Initialization of the level-set function ϕ can be accomplished in two ways. Initialization defines a front, such that all values of ϕ on that front are equal to zero. The first method initializes a front around the perimeter of the image, with the front propagating inwards toward the eventual solution. Unfortunately, this doesn't work in our application, as the droplet frequently intersects the side of the image, and sometimes the well walls are visible on the perimeter of the image. Thus the perimeter does not always satisfy the level-set constraint. The second method initializes the front to be a single point, propagating

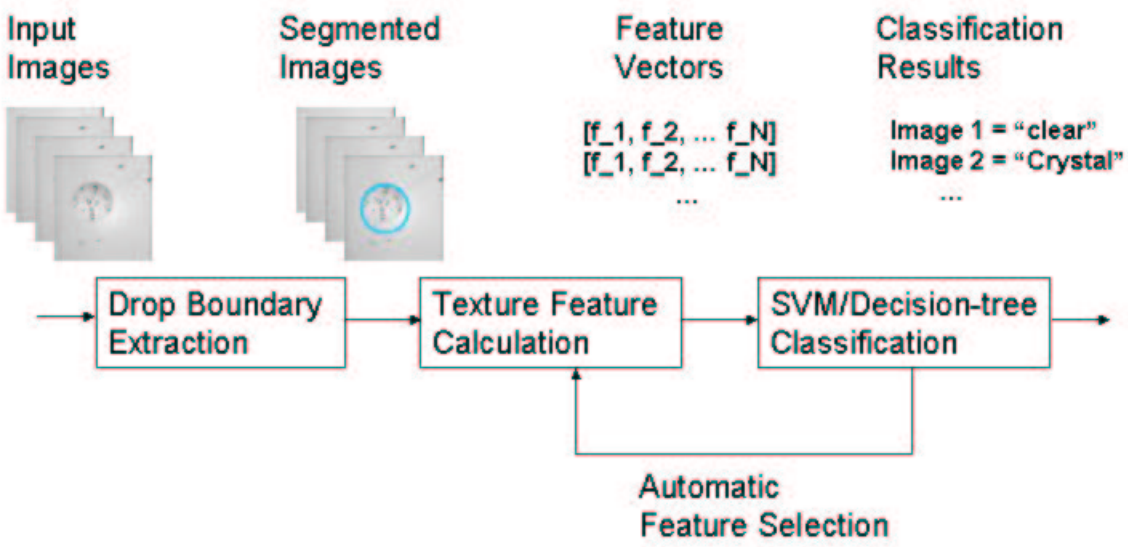

Figure 4: System diagram

outwards. This initialization method automatically meets the level-set constraint. The point is chosen to be in the corner of the image, inside any detected well walls. If the resulting front does not reach any other corner of the well, we choose another corner to initialize *phi*.

This front moves with a speed similar to that of a deformable contour, with terms for smoothness and edge attraction. The speed function is defined as

$$F = F_prop + F_curv + F_adv. \tag{1}$$

$F_prop = F_o$ is the propagation expansion speed of the front, a constant. $F_curv = -\epsilon\kappa$ is the dependence of the speed on the curvature, negative because convex curves should propagate less quickly, and concave curves (with negative curvature) should propagate outward. $F_adv = \vec{U}(x,y) \bullet \vec{n}$ is the advection speed, where $\vec{n}$ is the normal to the front, and $\vec{U}(x,y)$ is an underlying velocity field defined by the gradients in the image.

Some post-processing is needed, due to the sensitivity and generality of the level-set method. By discarding a margin of pixels near the drop boundary, small droplets are discarded, and only the interior of larger droplets are considered for feature extraction.

3.2 Feature Extraction

As discussed in Section 2, most current work focuses on geometry-based feature, e.g., the local gradient strength of the pixels or features from line detection algorithms. While this captures one important characteristic of the crystals, the local gradient strength may also be significant for certain types of precipitates, as shown in 2. One exception is the work in [1], in which the features are derived from frequency domain properties. In this case, still, both crystal images and precipitate images may contain high frequency components if distinctive edges are present, therefore these features cannot help to distinguish precipitates from hits and crystals either.

One observation from the images is that the images containing precipitates tend contain fluffy substance, whereas the crystal images tend to have more crispy textures, as demonstrated in Fig. 1 and 3. Therefore, it is expected that texture features might be helpful in distinguishing precipitates from crystals and hits.

We therefore investigate both the geometric and the texture features in greater details in Sections 3.3 and 3.4. Automatic selection of features are discussed in Section 3.5. See Section 5.2 for a comparison of their effectiveness.

3.3 Geometric Features

Features that have been tried in previous researches are mostly related to the geometric characteristic of a local region of the image (e.g., straight-line detection from Hough transform, conic curve fitting or line tracking).

As a representative case, the features in [4] are listed below and are used in the project for comparison with the proposed texture features:

F1 - Block direction gradient:

Average directional gradient of pixels within the best block.

F2 - Overall directional gradient:

Average directional gradient of all pixels inside the drop.

F3 - Hough transform:

Peak value in the Hough transform line detection within the best block.

F4 - Normalized Hough transform:

F3 normalized by the directional gradient of background pixels within the drop boundary.

F5 - Curve score

Evaluates the straightness and smoothness of the curves formed by the selected pixels.

F6 - Curve ratio

F5 normalized by F1.

F7 - Gray standard deviation

Standard deviation of the grey levels in the best block.

The notion of the directional gradient refers to the component along gradient value at each pixel discounting the effect of drop boundaries. The notion of the "best block" refers to a 25×25 block containing the most image pixels with high gradients. Interested readers are referred to the original paper for further details.

3.4 Texture Features

We base our texture feature extraction on the gray level co-occurrence matrix (GLCM) defined in [9]. More specifically, a GLCM gives a joint histogram of the quantized gray-level value pairs of two image pixels having a certain spatial relationship. The most generally used GLCMs are the ones for two neighboring pixels, aligned horizontally, vertically, along the diagonal or the counter-diagonal directions. A collection of different properties such as contrast, correlation and entropy are calculated for each GLCM. To make the features orientation-invariant, the average and variance of these features, over the GLCMs of different orientations, are used as the final texture features. For each image, 4 GLCMs are computed from the pixels within the drop boundary, corresponding to neighboring pixels aligned horizontally, vertically, diagonally and counter-diagonally.

For each GLCM, we compute 12 out of the 14 features as suggested in the original paper. The average and variance of these features over the 4 GLCMs are then calculated. Therefore, we obtain a 24-dimensional feature vector for each image. The texture features are listed below: F1 - Angular Second Moment

Measures the homogeneity of the image

F2 - Contrast

Measures the amount of local variations present in the images

F3 - Correlation

Measures the gray-tone linear-dependencies in the image

F4 - Inverse Difference Moment

moment calculated from the difference in gray-tone values of the adjacent pixels

F5 - Sum Average

Expectation calculated from the distribution of the sum of the gray-level values for the two pixels.

F6 - Sum Variance

Variance calculated from the distribution of the sum of the gray-level values for the two pixels.

F7 - Sum Entropy

Entropy calculated from the distribution of the sum of the gray-level values for the two pixels.

F8 - Entropy

Entropy calculated from GLCM; textures with more irregular patterns receive higher scores.

F9 - Difference Variance

Variance calculated from the distribution of the difference of the gray-level values for the two pixels.

F10 - Difference Entropy

Entropy calculated from the distribution of the difference of the gray-level values for the two pixels.

F11 - Information Measure of Correlation (A)

Measures similar property as in F3, based on entropy calculations

F12 - Information Measure of Correlation (B)

An alternative formula for F11, also based on entropy calculations

Interested readers are referred to [9] for the exact formulas. Note that although each of the features are calculated to capture some statistical property of the GLCM, hence the original image, the meanings of some of them are rather vague. They are, nevertheless, included in the initial classification to ensure that we have kept plenty of information from the original image data. Their effectiveness can then be automatically examined during the classification stage.

3.5 Automatic Feature Selection

As mentioned before, the actual meaning of many of the feature vectors may be quite vague, and their effectiveness for classification need to be further evaluated A straightforward approach is to perform a principle component analysis (PCA) on the collection of feature vectors from the training images. The features or linear combination of features related to the largest eigenvalues of the can be considered as more important. The drawbacks of such a scheme, however, are that it is subject to different scalings of each features, and that it assumes linear separable clustering of the dataset. We therefore pursue an alternative approach, namely to select features based on the classification output.

For the automatic decision-tree based classifier C5.0 in Section 4.1, the software provides a winnowing option which pre-selects the feature vectors used for the decision tree. Alternatively, one can examine the output of the decision tree, calculated from all features, and derive the relative importance of the features. The features used at the earlier decision stages are more important than the features used at the later stages.

For the support vector machine(SVM) in Section 4.2, the classifier also outputs the support vectors closest to the decision boundary. These correspond to images that are most easily confused during the classification. The feature vectors of these images can then be compared, and the features that distinguish these images tend to be more important than the others.

4 The Classifier

4.1 Automatic Decision Tree

The C5.0 classifier is a publicly available open-source free software for data mining [10]. It automatically extracts classification rules in the form of a decision tree from given training data. It also supports adaptive boosting, based on the idea from Freund and Schapire [11], where more than one classifiers are generated, and the classification result is voted by all the classifiers. In addition, C5.0 also provide an option called "winnowing", which pre-selects the more differentiating features in the design of the decision tress. This option is used in our feature selection process as well.

4.2 Support Vector Machine

The main idea of a support vector machine (SVM) is to construct a hyperplane as the decision surface in such a way that the margin of separation between positive and negative examples is maximized. Figure 5 illustrates the geometric construction of an optimal hyperplane for a two-dimensional input space [12].

The SVM method hinges to two mathematical operations: nonlinear mapping of an input vector into a high-dimensional feature space that is hidden from both the input and output (see Fig. 6); and construction of an optimal hyperplane for separating the features discovered in the previous step [12]. The first operation is in accordance with Cover's theorem on the separability of patterns, which states that such a multidimensional space may be transformed into a new feature space where the patterns are linearly separable with high probability, provided that the transformation is nonlinear and that the dimensionality of the feature space is sufficiently high. In the second operation, an optimal separating hyperplane is built in the high-dimensional feature space.

In the SVM method, the inner-product kernel plays the key role of mapping input patterns into the high-dimensional feature space. Inner-product kernel is defined in the form:

$$K(x, x_i) = \sum_{j=0}^{m_l} \phi_j(x)\phi_j(x_i), \tag{2}$$

where $K(x, x_i)$ denotes the inner-product kernel, $\phi_j(\dot{})$ denotes the jth non-linear function

mapping the input pattern into the jth-dimension feature space, x_i denotes the ith input pattern, and m_l is the dimensionality of feature space.

Most commonly used kernels are linear kernel and radial basis function (RBF) kernel. Linear kernel has the form $K(x, x_i) = x^T x_i$, and RBF kernel has the form $K(x, x_i) = \exp(-\frac{1}{2\sigma^2}||x - x_i||^2)$, where σ^2 is a design parameter specified by the user.

The overall architecture of the SVM is illustrated in Fig. 7.

For our problem, the goal is to find a separation hyperplane between "failure" and " success" trials from the given training dataset. The hyperplane should also work well on the new unknown test data. To construct such a hyperplane, the SVM minimizes the structural risk, given as the probability of misclassifying previously unseen data. Thus, theoretically, the SVM is expected to work well on new test data.

During the training process, all the information in the training set is gradually packed into a small number of support vectors. Only these vectors are used to classify new data. In this way, we can know what vectors are the most effective in distinguishing between the two classes [13][14][15]. The SVM exploits the information in the training set optimally. Compared with hand-crafted classification, it eliminates the process of defining appropriate discrimination criteria.

The selection of kernel directly affects the performance of the SVM. In our experiments, we tried both the linear kernel and RBF kernel. Experimental results show that, generally, the linear kernel performs better for our problem. Therefore only results using the linear kernels are presented in Section 5. One possible reason for this is that the feature vectors used in this project still have relatively low dimensions compared to other scenarios where the SVM is used.

5 Experimental Results

5.1 Level Set Drop Detection Results

Armed with the correct speed function, the algorithm successfully partitions the images, without the need for circle fitting. Thus, the level set algorithm is more robust than previous droplet detection methods. The number of images dropped due to failure in boundary detection were reduced from 15% to 6%, as irregular droplets no longer posed any difficulty to the segmenter. Most of these remaining failures were due to a failure to correctly detect the walls of the well.

5.2 Classification Results

The image dataset consisting 520 human annotated images are obtained from the website of Joint Center for Structure Genomics [16], and courtesy of Dr. Bern at the Palo Alto Research Center (PARC). The images consist of 130 samples in each of the four categories: Clear, Precipitate, Hit, and Crystals. For the binary classification problem we are concerned with, the first two categories are regarded as "Failure" and the last two categories are

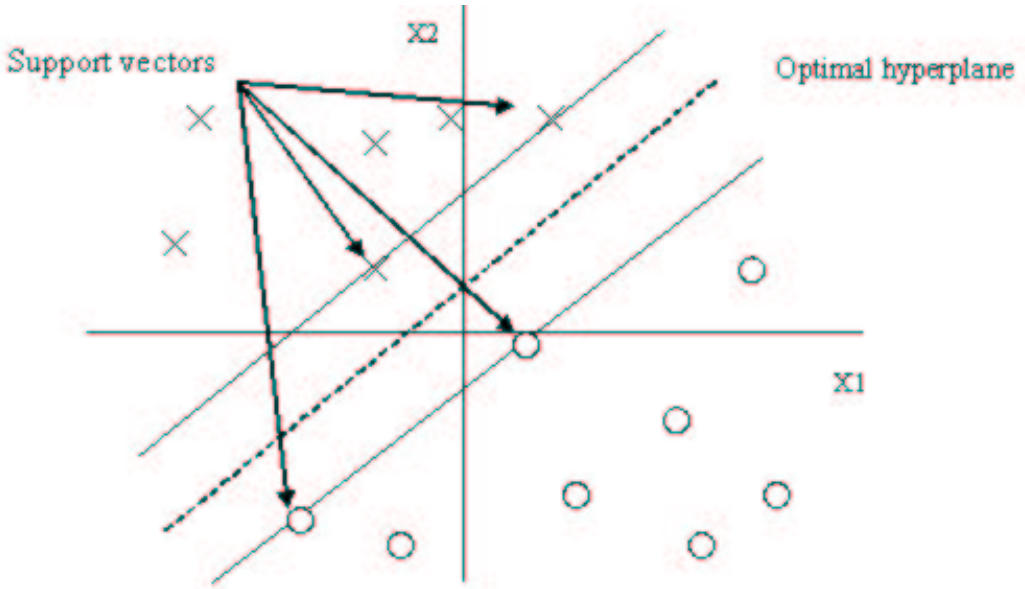

Figure 5: Illustration of an optimal hyperplane for linearly separable patters

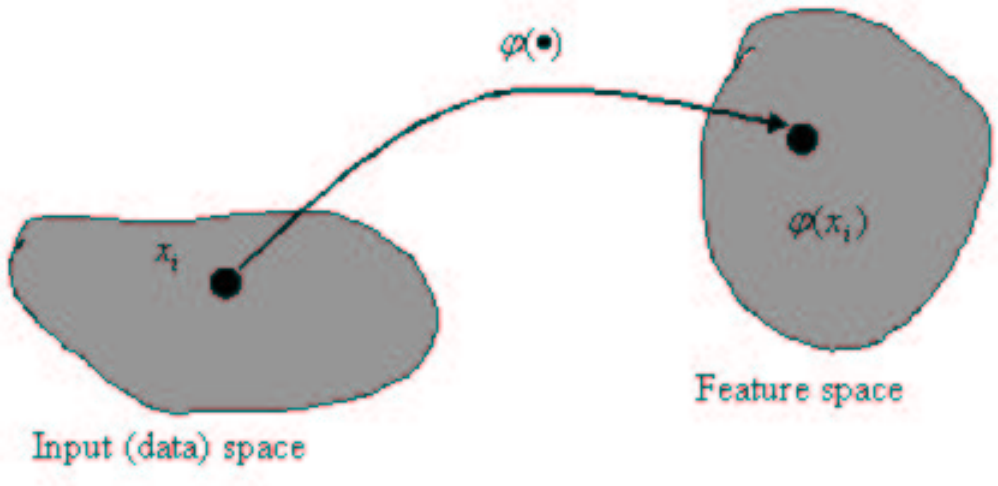

Figure 6: Non-linear function mapping from the input space to the feature space

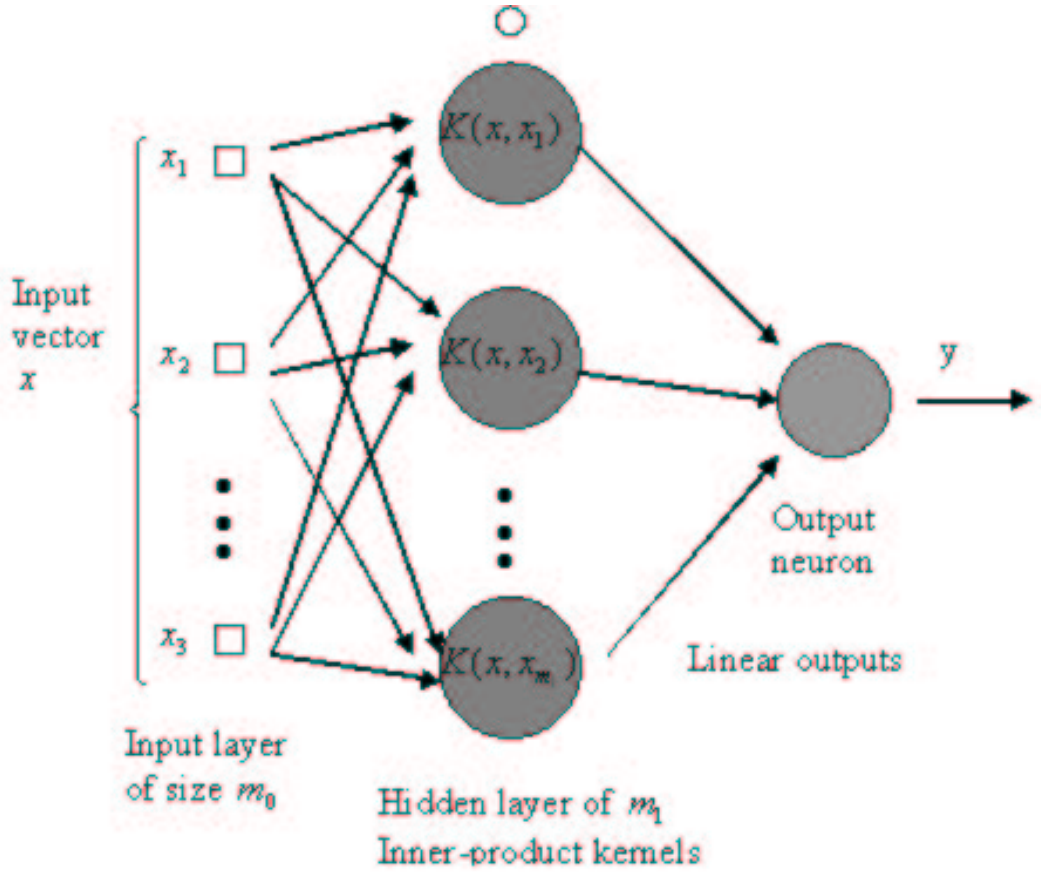

Figure 7: Architecture of support vector machine

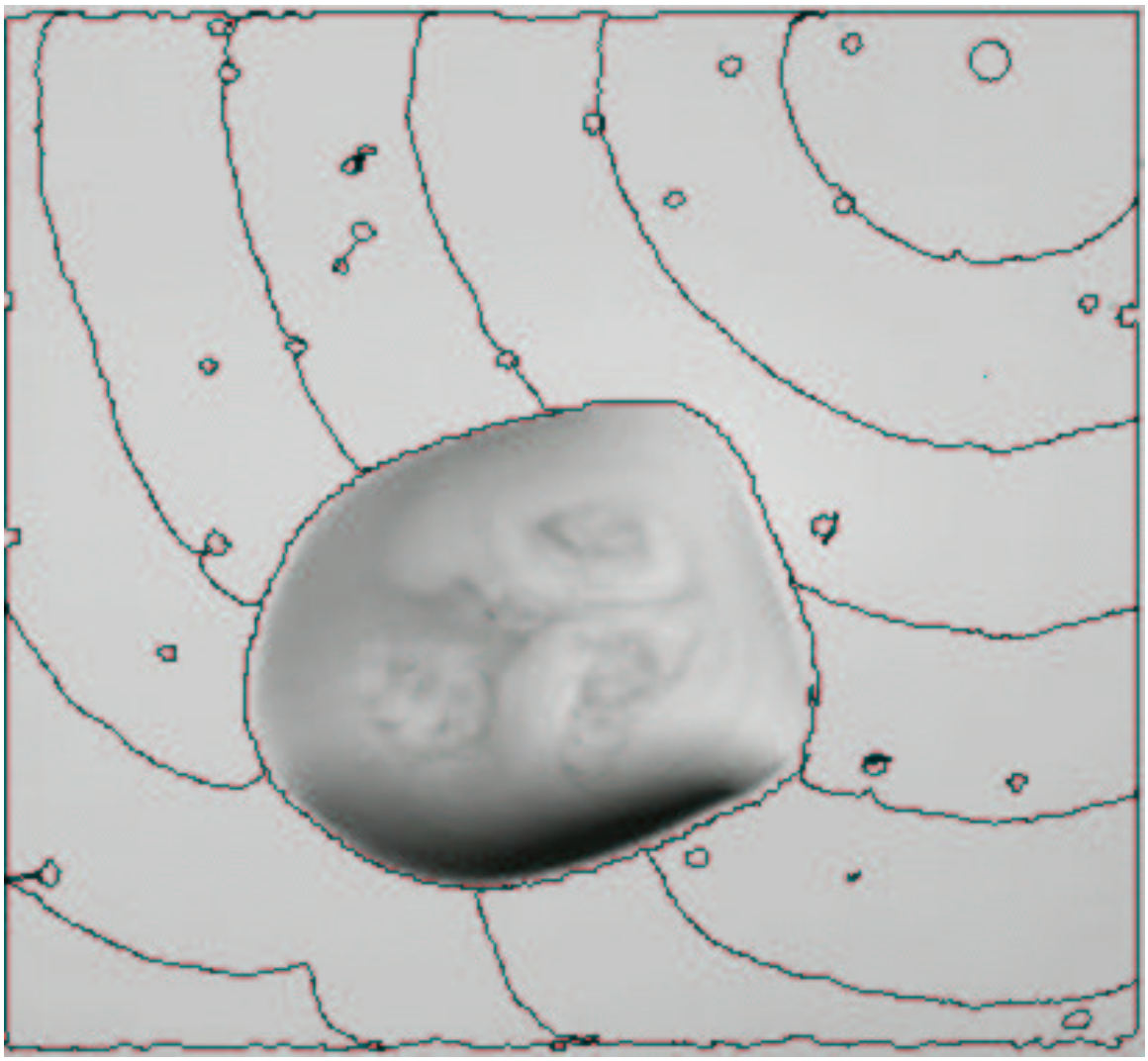

Figure 8: Result of Level-Set Boundary Front Propagating Across Image, Originating in the Upper Right

considered as "Success"

For feature extraction, we tested the geometric features, the texture feature and the combination of both. The dimensionality of the feature vectors are 11, 24, and 35, respectively.

Four different classifier configurations are evaluated: the SVM [1], the automatic decision-tree C5.0, C5.0 with boosting, and the handcraft decision tree classifier reported in **??**. The C5.0 classifier and its boosted version are applied to the given dataset of 520 images For the SVM classifier, however, due to the limitation of the code we are using, it can not be applied to large scale training set. Therefore we picked 50 images from each category and formed a subset of 200 images altogether and tested SVM classifier on this sub-dataset. It should be noted that the performance of the SVM may be affected by this constraint on the training data size.

To carry out the training and testing for each classifier, a 10-fold cross validation method was adopted. The entire dataset is randomly divided into 10 disjoint sets and 10 folds of experiments are performed. Each time, one of the ten sets is used as test set and the others as training set. Comparing the test results with the human annotations, we can calculate all rates of the true positives, false positives, true negatives and false negatives.

The classification results are listed in Tables 1-4. In general, the texture feature yields better results than geometric features, and the C5.0 classifier with boosting gives best results among the classifiers. The three automatic classifiers all outperform the handcrafted classifier, possibly because the parameters in the handcrafted classifier are no longer well-tuned for the new data set.

Using the texture features, SVM, C5.0, and C5.0 with boosting achieve accuracies of 72%, 80.77%, 82.3% respectively, and false positive rates at 20%, 14.23%, 12.69% correspondingly. With the geometric features, SVM, C5.0, and C5.0 with boosting achieve accuracies of 58%, 68.85%, 71.54% respectively, and false positive rates at 31%, 12.31%, 10.38% correspondingly. For combined features, SVM, C5.0, and C5.0 with boosting achieve accuracies of 58%, 84.62%, 85.39% respectively, and false positive rates at 31%, 14.61%, 9.6% correspondingly.

We also analyzed the trade off between false positive and false negative rates when the zero-crossing thresholding procedure in SVM classifier is replaced by level crossing. As the level decreases, more true crystals can be detected, at the cost of higher false positive rates. In classification, this trade off is usually characterized by the free-response operating characteristic(FROC) curve, which is given in Fig. 9 for the case of texture features.

Table 1: Experimental results for the SVM classifier

	Texture Features	Geometric Features	Combined Features
False Negative Rate	28%	42%	42%
False Positive Rate	20%	31%	31%

[1]The MATLAB code of the support vector machine is provided by Dr. Göktürk.

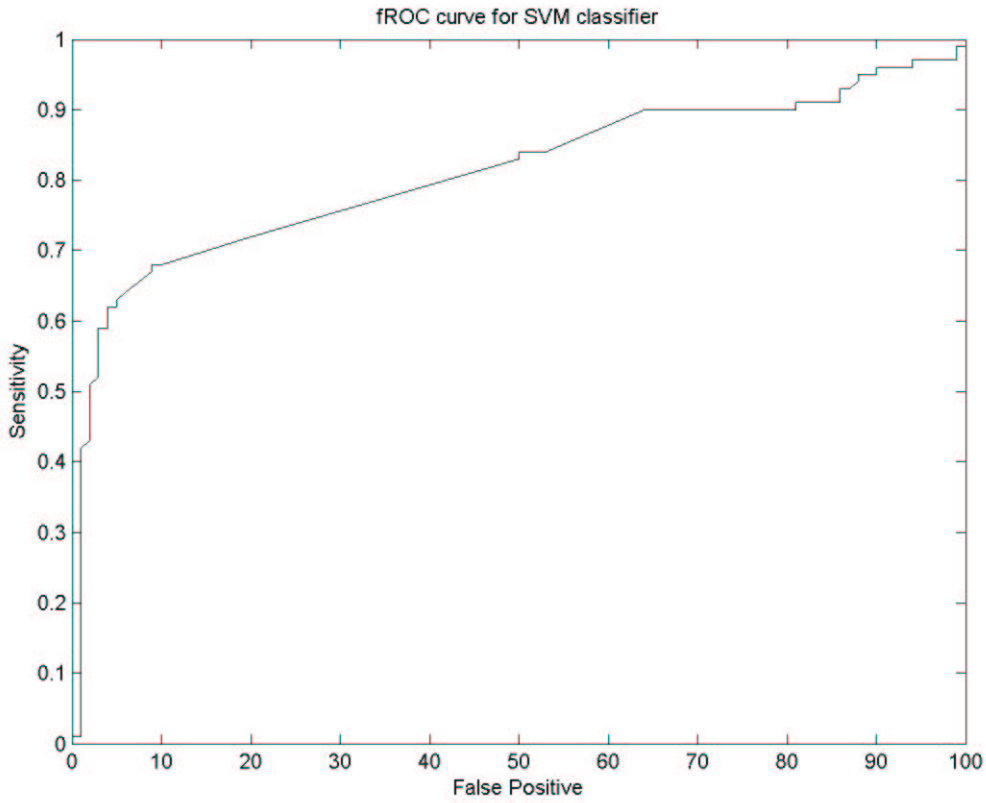

Figure 9: FROC curve

Table 2: Experimental results for the C5.0 classifier

	Texture Features	Geometric Features	Combined Features
False Negative Rate	19.2%	31.2%	15.4%
False Positive Rate	14.2%	12.3%	14.6%

Table 3: Experimental results for the C5.0 classifier with boosting

	Texture Features	Geometric Features	Combined Features
False Negative Rate	17.7%	28.5%	14.6%
False Positive Rate	12.7%	10.4%	9.6%

Table 4: Experimental results for the handcraft decision-tree classifier

	Geometric Features
False Negative Rate	47.3%
False Positive Rate	18.8%

5.3 Selected Features

By invoking the winnowing option in the C5.0 classifier, we can get some initial idea of how important each feature. Among the geometric features, the ones selected by the classifiers are: F2 - Overall directional gradient
F4 - Normalized Hough Transform
F5 - Curve ratio
F7 - Gray standard deviation

Among the texture features, the most important ones are: F2 - Contrast
F3 - Correlation
F5 - Sum Average
F9 - Entropy

6 Conclusion

We have implemented several new component algorithms, including the level-set method for drop boundary detection, GLCM-based texture feature extraction and the SVM classifier, for the classification of protein crystallization imagery.

Using a data set of 520 human annotated images, we have evaluated the performance of the proposed algorithms with respect to existing methods reported in literature. Without fine-tuning of the design parameters, the proposed feature extraction and classifier achieves

comparable or slightly superior results. For the binary classification between failed and successful protein crystallization trails, the best results are false positive rate at 9.6% and false negative rate at 14.6%, achieved by the C5.0 automatic decision tree classifier using both geometric and texture features.

7 Acknowledgments

The authors would like to thank Dr. Marshall Bern at Palo Alto Research Center (PARC) for providing the initial code-base of the program and additional image data, Daniel Russakoff for facilitating meetings and dispensing advice, and Drs. Sabastian Thrun and Gary Bradsky for initial direction.

References

[1] I. Jurisica et al., "Intelligent decision support for protein crystal growth," *IBM Systems Journal*, 2001.

[2] W. M. Zuk and K. B. Ward, "Methods of analysis of protein crystal images," *Journal of Crystal Growth*, vol. 110, 1991.

[3] J. Wilson, "Towards the automated evaluation of crystallization trials," *Acta Crystallographica D*, vol. 58, 2002.

[4] Marshall Bern et al., "Automatic classification of protein crystallization images using a line tracking algorithm," *Acta Crystallographica D*, 2003.

[5] Marshall Bern, "Grant Proposal R21 and R33," 2003.

[6] G. Spraggon, A. Kreusch S. A. Lesley, and J. P. Priestle, "Computational analysis of crystallization trials," *Acta Crystallographica D*, vol. 58, November 2002.

[7] G. Spraggon, A. Kreusch S. A. Lesley, and J. P. Priestle, "Automatic classification of sub-microlitre protein-crystallization trials in 1536-well plate," November 2002.

[8] Weeratunga S. and C. Kamath, "An investigation of implicit active contours for scientific image segmentation," *Video Communications and Image Processing, SPIE Electronic Imaging*, January 2004.

[9] R. Haralick, K. Shanmugam, and I. Dinstei, "Textural Features for Image Classification," *IEEE Transactions on Systems Man Cybernetics(SMC-3)*, 1973.

[10] "Data Mining Tools See5 and C5.0," *http://www.rulequest.com/see5-info.html*.

[11] Y.Freund and R.E. Schapire, "A short introduction to boosting," *Journal of Japanese Society for Artificial Intelligence*, vol. 14, 1999.

[12] Simon Haykin, "Neural networks: a comprehensive foundation," *Prentice Hall*, 1999.

[13] Salih Burak Göktürk and Carlo Tomasi, "A New 3-D Pattern Recognition Technique With Application to Computer Aided Colonoscopy," *Computer Vision and Pattern Recognition (CVPR'01)*, vol. 1, December 2001.

[14] B. Schölkopf, "Support vector learning," *R. Oldenbourg Verlag, Munich*, 1997.

[15] V.N. Vapnik, "The nature of statistical learning theory," *Springer, New York*, 1995.

[16] "Joint Center for Structural Genomics," *http://www.jcsg.org*.