

Vision System Development by Machine Learning:
Mashing Assessment in Brewing

Andy Song, Vic Ciesielski

Department of Computer Science, RMIT

GPO Box 2476V,

Melbourne Victoria, Australia

{asong,vc}@cs.rmit.edu.au

Peter Rogers

Carlton United Breweries Ltd

4-6 Southampton Crs, Abbotsford

VIC 3067, Australia

Peter.Rogers@cub.com.au

¹In Applied Intelligence, 15(8):777-795, September 2001.

Abstract

This paper describes the development of a machine vision application for automatic process assessment by image analysis and machine learning. The system is required to differentiate the various stages of a mashing process and determine the termination point. A large number of histogram, Haralick and Gabor features (835) were extracted from 275 training images. Three feature selection algorithms – wrapper, consistency filter and correlation filter – were then applied to the training data, resulting in feature sets of size 29, 15 and 11 respectively. A number of decision tree, rule induction and nearest neighbour classification algorithms were then applied to the reduced data set. For discriminating seven stages of the mashing process, the highest accuracy obtained was 71.6%. For the binary problem of differentiating the finish state from all of the other states the accuracy was 92.0%. This accuracy is good enough for deployment. The results indicate that using a large library of features and machine learning methods for removing redundant features can significantly reduce development times for vision systems by eliminating the time consuming manual search for the best discriminating features.

Keywords Machine vision, visual inspection, image classification, machine learning, feature extraction, feature selection, process assessment.

1 Introduction

Automated visual inspection systems are used in many industrial situations for a range of tasks such as detecting defective products(Lasher and Narayanan 1993) determining the state of a biochemical process and assessing the quality of raw materials (Uktu, Koksel, and Ozkara 1998; Winter, Sokhansanj, Wood, and Crerar 1996). The architecture of a typical automated visual inspection system is shown in figure 1. A camera captures the raw image at a key point in the process. The raw image is sent for preprocessing. In the preprocessing stage the raw image is cropped, normalised, smoothed, scaled and otherwise massaged to convert it into a standardized form. Features which are expected to be useful in discriminating the different states, for example, faulty and OK, are then extracted and represented in a feature vector. The feature vector is then passed to the classifier to determine the category of the new image, for example faulty or OK.

Developing an automated visual inspection system is usually a time consuming process. This is because there is no one set of preprocessing methods, set of features or classifiers that suits all visual inspection tasks. Most of the preprocessing programs need to be hand crafted for each new application. While there is a large literature on image features and there are several libraries of feature extraction programs, determining the best set of features for a new application is a time consuming process involving much trial and error. Similarly finding the best classifier for the task can involve much trial and error. In fact, the major problem in building an accurate system is to get the right combination of features and classifier.

To reduce the time spent in developing visual inspection systems we propose the approach

shown in figure 2. This has been made possible by recent progress in machine learning together with the availability of libraries of feature extraction programs and generalised machine learning software. In our approach a large number of raw images of the different process states are collected. Preprocessing converts these images into a standardized form and provides the training images. A large number of features which could be relevant are then extracted giving a very large feature vector for each image. At this stage there are two possibilities (1) Use the full feature vectors directly to generate the training set for a classifier and rely on the classifier to find the most salient features, or (2) Apply feature selection algorithms which remove irrelevant and redundant features from these large feature vectors and then generate the training set from the remaining features. Finally the classifier is induced from the training set. For the vision system developers, what needs to be done is to find a collection of available feature extraction programs, normalize their outputs to be suitable for learning and then generate classifiers based on the training data set. The “good” features will be selected during the learning process.

In this paper we describe the construction of a process assessment system in beer brewing using this methodology. The aim of the system is to determine whether the different stages of mashing can be differentiated by analysis of images of the surface of a mash tun, large vat of foamy material. In particular we are interested in whether the finishing state can be accurately determined. The images captured at various mashing times have quite different appearance. The appearance, which can be described by foam shape and colour distribution, reflects the underlying biochemical reaction. The mashing process should be terminated when a desired biochemical state has been reached.

Our hypotheses are:

1. There are a number of salient image features that change with increasing the mashing time.
2. These features are correlated with different states of the mashing process.
3. A classifier, developed with machine learning techniques, can be built to accurately classify the pictures taken at different states of the mashing processes.

2 Mashing

Mashing, one of the key processes in brewing, is a kind of enzymatic conversion process, in which soluble materials are extracted from malt into wort, the “un-fermented” beer. It is a most critical operation due to its influence on the physical properties of beer, such as foam, colour, clarity and most importantly, flavour. In practice, mashing is controlled by adjusting its duration, called “standing time”. Other operational parameters such as temperature gradient and agitation speed are fixed. The standing time, which depends on the malt quality and beer brand, varies from about 9 minutes to about 1 hour. An inappropriate standing time can cause difficulties in subsequent operations, such as fermentation, because of over extraction or inadequate extraction (Briggs and Hough 1982). In practice, the time is determined by brewers, half based on malt analytical data and half based on guesswork, which is based on experience.

The mashing process is carried out in huge stainless steel reaction containers, called mash tuns. The mash tuns are spherical and have viewport approximately $1m^2$ in size near the top. Our images (figure 5) were captured through this viewport. The temperature

of the process ranges from 30 degrees Celsius to 75 degrees Celsius and the tun becomes quite steamy.

There has been small amount of research in the use of vision in brewing industry in recent years. The research is mainly concerned with finding methods for automatic assessment, such as classifying barley (Uktu, Koksel, and Ozkara 1998), assessing foam stability of beer (Hegarty, R.Cope, Crompton, and Wallach 1991), monitoring cleaning processes (Wainwright 1999) and evaluating the status of yeast (Forrest and Grant 1993).

3 Machine vision

In general machine vision aims to recover useful information about a scene from its two dimensional projection, that is, an image captured by visual equipment such as cameras and visual probes (R. Hain 1995). Industrial machine vision systems are designed to eliminate human observation and error. As a result of research into finding better algorithms and more sensitive cameras, vision systems are being used in an ever growing range of applications (Lasher and Narayanan 1993). These applications fall into three categories: measurement, guidance and inspection. Measurement systems find the dimensions of an object through digitization and manipulation of the image of the object. Guidance systems cause a machine to perform certain actions based on what it “sees”. Inspection systems determine whether an object or a scene matches a preset description (Lasher and Narayanan 1993). The system described in this paper is an inspection system.

Machine vision techniques have been developed in many domains (Wallace 1988), including character recognition, human interaction, biomedical images, robotics, remote sens-

ing, automation with inspection, and quality control. In the area of quality assessment some vision systems have been successfully developed, such as grading of lentils (Winter, Sokhansanj, Wood, and Crerar 1996), grading of tobacco leaves (Zhang, Sokhansanj, Wu, Fang, Yang, and Winter 1998) and classification of barley (Uktu, Koxsel, and Ozkara 1998).

4 Feature extraction

In machine vision, there are two main approaches: pixel-based and feature-based. The systems following the former approach directly use the pixels of an image, and the systems following the latter approach work on the features extracted from the image. We adopt the feature-based approach in this project because this approach can dramatically reduce the dimensionality of the input data. Another benefit of the feature-based approach is that the key characteristics of images can be captured, and irrelevant information such as artifacts can be minimised.

The output of feature extraction is a feature vector, which can be viewed as a description, interpretation, or understanding of the scene provided by the feature extraction programs. A feature-based machine vision system is able to accomplish its pre-defined tasks, such as classification, by working on the relationship between the features and the underlying meanings.

There are a number of categories of feature extraction methods:

1. Statistical methods. These include gray level co-occurrence matrix (Haralick, Shanmugam, and Dinstein 1973), autocorrelation (Warner and Shank 1997), circular neighbourhood (Arof and Deravi 1997) and histogram features (Jain 1991).
2. Model-based methods. These include Markov Random Fields (Cross and Jain 1983).
3. Fractal-based methods. These include Wavelets (Mallat 1989) and Gabor filters (Chen and Chen 1996).

4.1 Feature Extraction for Mash Tun Images

The number of features that could be used in different image processing systems and vision systems is very large. A major problem in any computer vision application is to determine the best set of features to accomplish the required task.

The features that could be suitable for the mashing assessment problem need to have the following characteristics:

1. The features must be for the whole image, not for objects or image segments, because there are no obvious objects in our pictures, and it is hard to decide if one segment is more important than another.
2. The features must be rotation-invariant as the wort is under agitation and rotation during mashing.
3. The features must be reliable. This means that the feature vectors of images taken at the same states should be very close, while the feature vectors of the images

taken at different states should be far apart.

Based on the above criteria, we have used several feature extraction schemes. These generate histogram features, circular neighbourhood features, Haralick features and Gabor features. The last three kinds of features are used for extracting textural properties, which we believe are important for describing mashing images.

4.1.1 Histogram features

Image histograms are perhaps the simplest and most widely used image analysis tool. An image histogram gives the number of pixels which have a particular grey or red, blue green value:

$$p_u(x) = \frac{\text{Number of pixels with value } x}{\text{total number of pixels in the image}} \\ x = 0, \dots, L - 1 \quad (1)$$

L is the maximum possible value of x . For instance, L in a 8-bit gray level image is 256. $p_{red}(x)$ denotes the probability of the red component of a pixel having value x in an image. Figure 3 shows a histogram generated from a mashing image. The red, blue and green components are overlaid on the same axes.

Histogram features used in our experiments are the patterns of the histogram, which are the height and width of the histogram at certain positions. A pattern of the histogram can be measured in gray-level, in red space, in green space, in blue space, in hue space, or with the combinations. A total of 315 histogram features were generated. The moment and entropy of the histogram can also be used as features, but we have not done this.

4.1.2 Circular neighbourhood features

The basic idea of circular neighbourhood features is to cut an image into small non overlapping sub-images called lattices. There are $K \times L$ lattices in the image and each lattice consists of $N \times N$ pixels. The difference between neighbouring lattices (Arof and Deravi 1997) is to be compared. As illustrated in Figure 4, an image is marked from square A to square I.

In every lattice, the average intensity is calculated as $I(i, j)$, where i, j indicate the position of a lattice in the image. The difference of average intensity between a lattice with its all neighbours, $Diff(i, j)$, is defined in formula 2. The circular neighbourhood feature is the sum of $Diff(i, j)$ for all lattices and defined in formula 3:

$$Diff(i, j) = \sum_{i=-1}^1 \sum_{j=-1}^1 |I(i, j) - I(i + d_i, j + d_j)| \quad (2)$$

$$feature\ value = \sum_{i=0}^{K-1} \sum_{j=0}^{L-1} Diff(i, j) \quad (3)$$

In the our experiments, the sizes N of the lattice were set to 5, 10, 15, 20 and 25(pixels).

4.1.3 Haralick features

Haralick features were introduced in 1973. Haralick believed that texture information is contained in spatial relationships (Haralick, Shanmugam, and Dinstein 1973). These features have been widely used in machine vision areas such as remote-sensing (Rehrauer, Seidel, and Datcu 1999), document image understanding (Chetverikov, Liang, Komuves,

and Haralick 1996) and image database retrieval (Aksoy and Haralick 1998).

Haralick features are obtained from a gray-level co-occurrence matrix M , which keeps co-occurrence frequencies of pairs of gray intensity values. In such a matrix M with parameters d, θ , the value of element (i, j) in M is the frequency with which a pixel in gray tone i has a pixel in gray tone j at distance d , and the angle of two pixels is θ . It is defined by

$$\begin{aligned}
 M_{ij} = \# \{ & ((x, y), (x', y')) \in (L_x \times L_y) \times (L_x \times L_y) \mid \\
 & \| (x, y) - (x', y') \| = d, \operatorname{arctg} \left(\frac{x - x'}{y - y'} \right) = \theta \} \\
 & L_x \in \{1, 2, \dots, W\}, L_y \in \{1, 2, \dots, H\}
 \end{aligned} \tag{4}$$

where $\#$ denotes the number of elements in the set and W and H are width and height of the image.

Once the matrix is calculated, fourteen different categories of Haralick features such as angular second moment, contrast, correlation, variance, inverse difference moment, sum average, sum variance and sum entropy can be extracted at chosen θ and distance values (Haralick, Shanmugam, and Dinstein 1973). In our experiments, the max correlation coefficient has not been used. The angle values, θ , were set to 0° , 45° , 90° and 135° respectively, and distance values were arbitrarily set as 1, 4, 7, 10, 13, 16, and 19. The average of feature values for four angles is also used as an additional feature. Thus $13 \times 5 \times 7 = 455$ Haralick features are generated for each image.

4.1.4 Gabor features

Like Haralick features, Gabor features are also a kind of texture description. These features have been used in various domains in machine vision, such as textural feature extraction (Setchell and Campbell 1999), texture segmentation (Mittal 1999), face recognition (Lyons, Budynek, Plante, and Akamatsu 2000) and handwriting recognition (Lee and Liu 1999). Gabor features are obtained by Gabor filters, which have been shown to be optimal in the use of minimising the joint two-dimensional uncertainty in space and frequency. The Gabor filter is a harmonic oscillator, composed of a sinusoidal plane wave of a particular frequency and orientation with a Gaussian kernel (Chen and Chen 1996):

$$\Psi(x, y; \sigma, u, v) = \left\{ \exp \frac{-(u^2 + v^2)}{2\sigma^2} (x^2 + y^2) \right\} \exp i(ux + vy), \quad (5)$$

where (x, y) are the variables representing position in the spatial domain, (u, v) are the spatial frequencies, and σ is the width of the Gaussian. The Gabor transform, $G(x, y)$, of an image fragment, $I(x, y)$, is defined as the convolution of a Gabor kernel Ψ with I :

$$G(u, v) = \int \int \Psi(x, y; \sigma, u, v) I(x, y) dx dy \quad (6)$$

The orientation of the filter, which is defined as $\theta = \tan^{-1} v/u$, and the size of an image fragment are tunable. The Gabor filter in our experiments used six directions and ten sizes, giving 60 Gabor features in total.

5 Machine Learning and Classification

Machine learning, like knowledge representation and reasoning, plays a central role in artificial intelligence. Machine learning permeates all areas of AI: problem solving, speech

recognition, vision, robotics, planning and others.

Classification is one of the most prominent and basic machine learning tasks. The aim is to extract a decision rule or some other decision making procedure from sample data. The outcome of a classification system is a decision-making system, called a classifier. The classifier is expected to behave well on new data from the same population. Typical classifiers include linear discriminants, nearest neighbour classifiers, decision trees, rule inducers and neural nets (Carbonell 1990). Classifiers are created by inductive supervised learning. This means that the classifiers are generalized from known examples and all the examples have their class labels associated with the feature values.

5.1 Training and Testing

The process of classifier generation is called training. To evaluate whether the rules of feature space partitioning allow correct classification for new cases, those rules are applied on a set of unseen data. This process is called test. Classification performance is measured by the test error rate which is the percentage of the test instances that have been incorrectly classified. This is an estimate of the true error rate which is the error rate the classifier would have on the entire population of cases.

5.2 Cross-Validation

In situations where there are 250-300 cases in total a better estimate of the true error rate can be obtained by cross-validation. The idea is to randomly divide the data into N mutually exclusive partitions (folds), keep one fold for test and use the rest for training.

Then another fold will be selected for test and all the other folds for training. This process is repeated N times until all folds have been used once for test and $N-1$ times for training. The average error rate from the N repetitions is the cross-validation error rate. In our experiments the number of folds was set to 10. The results of experiments shown in the following section are in form of classification accuracy, which is $(1 - \text{error rate})$. The advantage of cross-validation is that all the instances are used in both training and test processes (Weiss and Kulikowski 1991).

5.3 Classifiers for the Mash Tun Images

There is large number of existing classifiers that could be used for the image feature data (Lim and Shih 2000). We have experimented with the ones provided in the WEKA system (Witten and Frank 2000) and provide the results for six of these: C4.5, PART, 1R, Naive Bayesian Classifiers, Decision Tables and an instance-based classifier (IB). These classifiers represent three categories of classification methods: statistical (IB and Naive Bayes Classifiers), rules(1R and Decision Table) and decision trees(C4.5 and PART). We have chosen these representative classifiers not only for comparing their performance on the mashing data, but also to obtain some insights into the data and the classification problem.

- C4.5 generates a decision tree by a top-down approach. It recursively partitions the instances into their classes by measuring information gain ratio (Quinlan 1993). This is the most widely used decision tree algorithm and is a common benchmark in machine learning.

- PART is another decision tree generator, which repeatedly generates partial decision trees in a separate-and-conquer manner (Frank and Witten 1998). It is a variation and refinement of the basic C4.5 approach and is expected to be better in some circumstances.
- Naive Bayes classifiers use a probabilistic induction approach. The attributes are assumed to be normally distributed and statistically independent. Classification is achieved by computing the posterior probability of each class and choosing the class with the highest posterior probability. Naive Bayesian classifiers work well when there are linear boundaries between the classes (John and Langley 1995).
- 1R is a very simple one-rule classifier. Rules containing a single attribute are generated. The rule which has the lowest error rate on the training data is selected as the classifier (Holte 1993). 1R can be used to determine whether there is a dominant attribute correlated to the classes.
- A decision table has two components: a schema and a body which consists of a set of features and labelled instances respectively. Given a new instance, a decision table classifier searches for exact matches using only the features in the schema (Kohavi 1995). It extends the basic 1R idea to several attributes.
- Instance-based learning employs the K -nearest neighbour rule. The class of a test instance is determined by a majority vote of its K nearest neighbours in the training data. Instance-based learning makes no assumptions about the probability distributions of the attributes and is suitable for data which do not have linear boundaries between the classes (Aha and Kibler 1991).

6 Feature Selection

Feature selection is another way of viewing the elimination of irrelevant or redundant features. The features not eliminated are selected. The feature vectors generated in feature extraction are potentially very large in size. In our case, there are 835 feature elements in each vector. To improve the quality of the data and remove as much of the irrelevant and redundant information as possible, the feature selection process is used. It is expected that only salient features will be kept. Fewer features mean less computation for feature extraction in the deployed system and less computation for classification. A fast feature extraction and classification process is highly desirable, especially for an on-line system.

In our the experiments we used three feature selection methods: wrappers, consistency filters and correlation based filters. These are all implemented in the WEKA system (Witten and Frank 2000).

- A wrapper uses exhaustive search to find an optimal feature subset. The algorithm partitions the features into different subsets and estimates the accuracy of each subset by applying a chosen learning algorithm on the subsets (Kohavi and John 1997).
- The consistency filter is a probabilistic approach for feature selection. It searches for the optimal subsets by applying the Las Vegas Search algorithm and calculating inconsistency of matching instances (Liu and Setiono 1996).
- Correlation-based feature selection uses a correlation based heuristic to evaluate

each feature. The heuristic is ‘good feature subsets contain features highly correlated with the class, yet uncorrelated with each other’ (Hall 1998)

7 Experiments and Results

7.1 Image Capture and Preprocessing

The images used in this project were taken systematically from the mash tun every five minutes. We tried to keep the conditions consistent, so that the angle, position and zoom setting of the camera were all fixed. The camera was set to work at its highest resolution which is 1280 by 960 pixels. After the images were transferred to a computer, they were consistently cropped to remove the irrelevant edge areas and scaled to 450 by 350 pixels. The images are saved in 24-bit colour JPEG format. The following experiments are based on these images. According to the time that the pictures were taken they are categorised as: “*start*” is taken at the very beginning of mashing, “*start+*” is taken 5 minutes later, “*start++*” is taken 10 minutes later, “*finish*” is taken just before the stopping point, “*finish-*” is taken 5 minutes before “*finish*”, “*finish--*” is taken 10 minutes before “*finish*” and others between “*start++*” and “*finish--*” are “*middle*”s. For each state, three pictures are taken, one after the other, as quickly as permitted by the camera operation. Generally all pictures are taken within 5 seconds.

A set of images from one mashing run is illustrated in Figure 5. In the following experiments, the state labels are used as classes for classification. There are 275 pictures in total from 15 different mashing processes.

7.2 Typical Features

After the preprocessing, all the images are passed to the feature extraction programs which generate Haralick features, Gabor features, circular neighbourhood features and histogram features for each image. All feature values are normalized to the range of -1 to 1.

One typical set of histogram feature values for a 35 minute mashing process is shown in the left graph of figure 6. For each time point, there are 3 data points which represent the three pictures been taken at that time point. The feature values at each time point oscillate inconsistently from the beginning to the end of the mashing process. This kind of pattern is hard to be learnt by classifiers.

The graph at the left hand side of figure 6 shows the circular neighbourhood feature values for a 25-minutes mashing process. For each state, there are 3 data points which represent the three pictures been taken at that time point (Note: only two are visible in black and white). The graph indicates that the circular neighbourhood feature values of the same state are quite similar and the difference between states are clearer, especially the last points which are at the “finish” stage. From this run would seem that the finish state can be easily separated from the others. However this was not the case for all runs. The lattice size for generating this feature graph was 5.

7.3 Comparison of Classifiers

The six classifiers mentioned above were applied to the generated features to compare their performance. The results are presented in Table 1. The first column lists the name

of the classifiers and the parameter settings. The middle column shows the training accuracy of these classifiers, and the right column is the test accuracy obtained by 10 fold cross-validation. In the subsequent discussion the term ‘accuracy’ refers to cross-validation accuracy.

The accuracy obtained by 1R is 37.1%, which indicates that there is no single dominant attribute which can be used to accurately classify all images. The accuracy obtained by Naive Bayes is 45.4%. This is quite low which suggests the attributes are not independent and that there are no distinguishable linear boundaries between the classes. For the decision table, the accuracy is 52%. It appears that these three simple classifiers are not very suitable for this image feature data.

When using one nearest neighbour ($K = 1$) in instance-based learning, the accuracy achieved was 66.5%. Somewhat surprisingly the performance for all other choices of K the performance was considerably worse.

The M parameter in C4.5 is for setting the minimum number of instances in each leaf of the decision tree, and the C parameter is for setting the confidence for reduced error pruning. The two parameters are also used in PART. With different combinations of M and C , the accuracy of C4.5 and PART varies from around 60% to 66%. There are significant differences between training and cross validation correctness which indicates significant over training. While the over training could be reduced by extreme choices for M and C this resulted in very low cross validation correctness.

7.4 Comparison of Feature Groups

In the experiments presented in the previous section classification was based on the whole feature set. In order to compare the contributions of different features groups, the classification has also been done based on each feature extraction scheme. The classifier used for this comparative experiment is C4.5 ($M=8$ and $C=0.2$). The results are presented in Table 2.

As can be seen from Table 2 the accuracy of the Haralick group of features alone is 62.9%. This compares to a best accuracy of 66.5% using all features. This suggests that the Haralick features are the most useful.

7.5 Feature Selection

Table 3 shows the feature subsets selected by the three feature selection methods: Correlation-based feature selection gives 29 features while consistency and wrapper select 15 and 11 features respectively. In the section of Haralick features in table 3, the numbers in brackets represent (1) Angular Second Moment,(2) Contrast,(3) Correlation,(4) Variance,(5) Inverse Difference Moment,(6) Sum Average,(7) Sum Variance,(8) Sum Entropy,(9) Entropy,(10) Difference Variance,(11) Difference Entropy,(12)Mean of Correlation-1 and (13)Mean of Correlation-2; θ is the angle value, ‘D’ is the distance measure, and ‘Average’ is the average of the feature values obtained at four angles: 0° , 45° , 90° and 135° . In the section of histogram features, ‘I’ indicates at which intensity value, the height of the histogram is measured as one feature and ‘P’ indicates at which height value (from 0 to 100) the width of the histogram is taken, (Grey),(r),(g),(b),(w)

mean that the histogram is generated in grey level, red space, green space, blue space or the combination of RGB space respectively.

The lines in boldface in 3 indicate features selected by at least two methods. No feature was selected by all three methods and only 4 features were selected by 2 methods.

Table 4 lists the classification accuracy based on the subsets generated in feature selection. The classifier used was C4.5. The second column lists the number of features selected from the 835 features after the feature selection process. The percentages are also listed. From the table, we can see that all the three methods reduce the dimensionality of the feature space dramatically. CFS keeps 29 features and still remains the classification accuracy at the same level as using all features (refer to Table 2). The results from Wrapper are particularly good. These use only 11 features and the accuracy has been improved to 71.6%. However, the exhaustive search in wrapper is very resource consuming. The run time was more than 60 hours on a Sun Enterprise 4000 server.

7.6 Binary Classification

To determine which states are easiest to distinguish, we performed a number of binary classification experiments. For example, to test the “start” state, all images that are taken in other states are labelled as “not-start”, so in the classification only two classes exist, “start” and “not-start”. Then the classification accuracy for this state can be obtained. These experiments also used C4.5 as the classifier.

The results of all states are listed in Table 5, which shows that the state “*start*” and state “*finish*” have higher classification accuracy. That indicates that these two states

can be differentiated more easily than the other states. This is very important for our application, especially identifying the state “*finish*”.

8 Conclusions

The goal of the work presented in this paper was to develop a mashing process inspection system by machine vision and machine learning. In this we have been successful. Using feature selection we have been able to reduce an original set of 835 features to a final set of 11 and achieve the classification accuracy required by a deployed system. With respect to our specific hypotheses:

1. There are a number of salient image features that change with increasing the mashing time.

There was some evidence for this hypothesis since some classifiers achieved high enough accuracy and the feature selection algorithms found small feature subsets which also gave high enough accuracy.

2. These features are correlated with different states of the mashing process.

That there is a correlation was established by achieving a high enough classification accuracy. However, the fact that the simpler classifiers, 1R, decision table and naive Bayes, performed so poorly suggests that this relationship is very complex and determining the exact analytical relationship would be very difficult.

3. A classifier, developed with machine learning techniques, can be built to accurately classify the pictures taken at different states of the mashing processes.

This hypothesis was established since a system delivering acceptable accuracy was developed. Furthermore, the development methodology which relied on generating very large feature vectors and using machine learning techniques to eliminate redundant attributes was also successful. Relying on the C4.5 classifier to remove redundant attributes was very successful and the wrapper feature selection algorithm even more so.

The methodology we propose in this paper involves using a library of feature extraction programs to build huge feature vectors which include many different kinds of image features and then selecting salient features by machine learning. Features are evaluated by their contribution, either alone or in combination with other features, towards final accuracy. For example, while histogram features overall are not much good (see Table 2), one of the histogram feature was in the best combination (see Table 3).

This approach combines machine vision and machine learning techniques and has been successfully applied in developing the mashing process inspection system. We believe it will also be useful for other kinds of vision applications.

References

- Aha, D. and D. Kibler (1991). Instance-based learning algorithms. *Machine Learning* 6, 37–66.
- Aksoy, S. and R. Haralick (1998). Textural features for image database retrieval. In *Content-Based Access of Image and Video Libraries, 1998. Proceedings. IEEE Workshop on*, pp. 45–49.
- Arof, H. and F. Deravi (1997). Circular neighbourhood features for texture classification. In *6th International Conference on Image Processing and Its Application*, Volume II, pp. 609–612.
- Briggs, D. E. and J. S. Hough (1982). *Malting and brewing science*. London ; New York : Chapman and Hall,.
- Carbonell, J. (Ed.) (1990). *Machine learning : paradigms and methods*. Cambridge, Mass : MIT Press.
- Chen, C.-C. and D. C. Chen (1996, September). Multi-resolutional Gabor filter in texture analysis. *Pattern Recognition Letters* 17(10), 1069–1076.
- Chetverikov, D., J. Liang, J. Komuves, and R. Haralick (1996). Zone classification using texture features. In *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, Volume 3, pp. 676–680.
- Cross, G. R. and A. K. Jain (1983, January). Markov random field texture models. *IEEE Trans. on Patt. Anal. and Machine Intell. PAMI-5*(1), 25–39.
- Forrest, I. and I. Grant (1993). Application of machine vision methods to in-line measurement of yeast (and other particulate matter) in beer. In *Proceedings of EBC, Oslo*, pp. 501–507.

- Frank, E. and I. H. Witten (1998). Generating accurate rule sets without global optimization. In *Machine Learning: Proceedings of the Fifteenth International Conference*, pp. 144–151. Morgan Kaufmann Publishers, San Francisco, CA.
- Hall, M. A. (1998). *Correlation-based Feature Subset Selection for Machine Learning*. Ph. D. thesis, the University of Waikato.
- Haralick, R. M., K. Shanmugam, and I. Dinstein (1973, November). Textural features for image classification. *IEEE Transactions On Systems, Man, and Cybernetics SMC-3*(6), 610–621.
- Hegarty, P., R. Cope, I. Crompton, and E. Wallach (1991). An image analysis system for measuring beer foam stability. In *Proceedings of EBC*, pp. 465–472.
- Holte, R. C. (1993). Very simple classification rules perform well on most commonly used datasets. *Machine Learning 11*, 63–91.
- Jain, A. K. (1991). *Fundamentals of digital Image processing*. Prentice Hall.
- John, G. H. and P. Langley (1995, August). Estimating continuous distributions in Bayesian classifiers. In Besnard, Philippe and S. Hanks (Eds.), *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence (UAI'95)*, San Francisco, CA, USA, pp. 338–345. Morgan Kaufmann Publishers.
- Kohavi, R. (1995, April). The power of decision tables. In N. Lavrač and S. Wrobel (Eds.), *Proceedings of the 8th European Conference on Machine Learning*, Volume 912 of *LNAI*, Berlin, pp. 174–189. Springer.
- Kohavi, R. and G. H. John (1997). Wrappers for feature subset selection. *Artificial Intelligence 97*(1–2), 273–323.
- Lasher, B. and M. Narayanan (1993). Vision systems-an overview. In *WESCON/'93*.

- Conference Record*, pp. 112–115.
- Lee, R. and J. Liu (1999). An oscillatory elastic graph matching model for recognition of offline handwritten chinese characters. In *Knowledge-Based Intelligent Information Engineering Systems, 1999. Third International Conference*, pp. 284 –287.
- Lim, T. and Y. Shih (2000, September). A comparison of prediction accuracy, complexity and training time of thirty-three old and new classification algorithms. *Machine Learning* 40(3), 203–228.
- Liu, H. and R. Setiono (1996). A probabilistic approach to feature selection – a filter solution. In *Proc. 13th International Conference on Machine Learning*, pp. 319–327. Morgan Kaufmann.
- Lyons, M., J. Budynek, A. Plante, and S. Akamatsu (2000). Classifying facial attributes using a 2-d gabor wavelet representation and discriminant analysis. In *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, pp. 202 –207.
- Mallat, S. G. (1989, July). A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Transaction on Pattern Analysis and Machine Intelligence* 11(7).
- Mittal, N.; Mital, D. K. L. C. (1999). Features for texture segmentation using gabor filters. In *Image Processing And Its Applications, 1999. Seventh International Conference on*, Volume 1, pp. 353 –357.
- Quinlan, R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA.
- R. Hain, R. Kasturi, B. S. (1995). *Machine Vision*. McGraw-Hill.

- Rehrauer, H., K. Seidel, and M. Datcu (1999). Characteristic scale detection in remote-sensing data. In *Geoscience and Remote Sensing Symposium, 1999. IGARSS'99 Proceedings. IEEE 1999*, Volume 1, pp. 116 –118.
- Setchell, C. and N. Campbell (1999). Using colour gabor texture features for scene understanding. In *Image Processing And Its Applications, 1999. Seventh International Conference on*, Volume 1, pp. 372 –376.
- Uktu, H., H. Koksel, and R. Ozkara (1998, Nov/Dec). Classification of barleys based on malting quality by image analysis. *Journal of Brewing Institute* 104(6), 351–151.
- Wainwright, T. (1999, April/May). Fermentation and cip control using a topscan camera. *Ferment*, 50–52.
- Wallace, A. (1988, May). Industrial applications of computer vision since 1982. In *Computers and Digital Techniques, IEE Proceedings*, Volume 135, pp. 117 –136.
- Warner, T. A. and M. C. Shank (1997, April). Spatial autocorrelation analysis of hyperspectral imagery for feature selection. *Remote Sensing of Environment* 60(1), 58–70.
- Weiss, S. M. and C. A. Kulikowski (1991). *Computer systems that learn : classification and prediction methods from statistics, neural nets, machine learning, and expert systems*. San Mateo, Calif. : M. Kaufmann Publishers.
- Winter, P., S. Sokhansanj, H. Wood, and W. Crerar (1996, July). Quality assessment and grading of lentils using machine vision. *Agricultural Institute of Canada Annual Conference*.
- Witten, I. H. and E. Frank (2000). *Data mining : practical machine learning tools and techniques with Java implementations*. San Francisco, Calif. : Morgan Kaufmann.

Zhang, J., S. Sokhansanj, S. Wu, R. Fang, W. Yang, and P. Winter (1998, January). A transformation technique from RGB signals to the munsell system for color analysis of tobacco leaves. *Computers and Electronics in Agriculture* 19(2), 155–166.

Classifier	Training	Cross-validation
	Correctness	Correctness
1R	53.1%	37.1%
Naive Bayes	48.0%	45.4%
Decision Table	81.1%	52.0%
IB (k=1)	100%	66.5%
IB (k=2)	84.0%	57.8%
IB (k=20)	54.9%	46.2%
IB (k=21)	54.2%	43.3%
IB (k=30)	47.6%	43.3%
C4.5 (M=2 C=0.2)	97.5%	66.5%
C4.5 (M=3 C=0.2)	93.8%	64.7%
C4.5 (M=5 C=0.3)	86.5%	60.0%
C4.5 (M=6 C=0.1)	81.8%	62.2%
C4.5 (M=8 C=0.2)	81.5%	64.7%
C4.5 (M=12 C=0.2)	79.2%	61.1%
PART (M=3 C=0.2)	95.0%	64.7%
PART (M=4 C=0.3)	93.0%	61.8%
PART (M=5 C=0.2)	90.5%	64.0%
PART (M=8 C=0.1)	86.5%	63.3%
PART (M=12 C=0.2)	77.8%	63.3%

Table 1: Comparison of Classifiers

Features	Training Correctness	Cross-validation Correctness
Haralick	80.4%	62.9%
Gabor	72.7%	46.9%
Histogram	68.0%	37.0%
Circular Neighbourhood	62.2%	45.8%

Table 2: Comparison of features

Features	CFS	Consistency	Wrapper
Haralick	No. 1:(1), $\theta=0$, D=1 No. 11:(3), $\theta=0$, D=1 No.12:(3), $\theta=45$, D=1 No.15:(3), (Average), D=1 No.21:(5), $\theta=0$, D=1 No.26:(6), $\theta=0$, D=1 No.46:(10), $\theta=0$, D=1 No.48:(10), $\theta=90$, D=1 No.49:(10), $\theta=135$, D=1 No.51:(11), $\theta = 0$, D=1 No.56:(12), $\theta = 0$, D=1 No.61:(13), $\theta = 0$, D=1 No.62:(13), $\theta = 45$, D=1 No.126:(13), $\theta = 0$, D=4	No. 2:(1), $\theta=45$, D=1 No.12:(3), $\theta=45$, D=1 No.13:(3), $\theta=90$, D=1 No.106:(8), $\theta=0$, D=4 No.130:(13), (Average), D=4 No.180:(10), (Average), D=7 No.188:(12), $\theta = 135$, D=7 No.245:(11), (Average), D= 10 No.284:(5), $\theta = 135$, D=13	No. 11:(3), $\theta=0$, D=1 No.50:(10), Average, D=1 No.80:(3), Average, D=4 No.116:(10), $\theta = 0$, D= 7 No.153:(4), $\theta = 90$, D = 7 No.173:(8), $\theta = 90$, D=7 No.182:(10), $\theta = 45$, D = 7 No.219:(4), $\theta = 135$, D = 10 No.238:(8), $\theta = 90$, D = 10 No.239:(8), $\theta = 135$, D=10
Gabor	No.21: Scale= 4, $\theta=120$ No.22: Scale= 4, $\theta=180$ No.23: Scale= 4, $\theta=240$ No.26: Scale= 5, $\theta = 120$ No.29: Scale= 5, $\theta = 240$	No.10: Scale=2, $\theta=180$ No.23: Scale=4, $\theta=240$	
Histogram	I= 196(w) I= 224(w) I= 196(b) I= 7 (g) I= 224(g) I= 49 (r) I= 126(r) I= 238(r) P= 65(a)	I= 224(w) I= 107(g) I= 126(r) P= 95(a)	I = 182 (Grey)
Neighbour	Lattice size= 15		

Table 3: Feature Subsets chosen by Feature Selection Methods

Features Selection	Selected Features	Training	Cross-validation
Methods	Numbers & Percentage	Correctness(C4.5)	Correctness(C4.5)
CFS	29 (3.4%)	88.7%	67.3%
Consistency	15 (1.8%)	88.7%	57.5%
Wrapper	11 (1.3%)	92.7%	71.6%

Table 4: Comparison of feature selection methods

States	Training Correctness	Cross-Validation Correctness
Start	99.6%	97.5%
Start+	98.5%	88.0%
Start++	98.9%	80.7%
Middle	98.2%	89.1%
Finish--	99.3%	83.3%
Finish-	99.6%	88.4%
Finish	99.3%	92.0%

Table 5: Binary Classification based on States

Figure 1: Architecture of a Typical Automated Visual Inspection System

Figure 2: A Methodology for Vision System Development Using Machine Learning

Figure 3: A Sample of an RGB Histogram

Figure 4: Circular Neighbourhood Features

Figure 5: A series of images from the beginning of mashing to finish

Figure 6: Histogram Features and Circular Neighbourhood Features for a Mash Run



Figure 1: Architecture of a Typical Automated Visual Inspection System

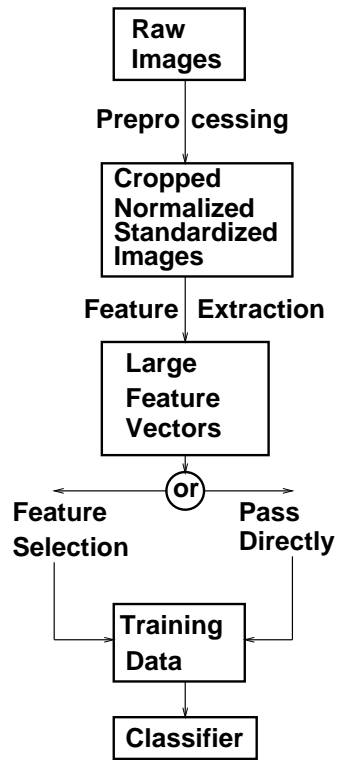


Figure 2: A Methodology for Vision System Development Using Machine Learning

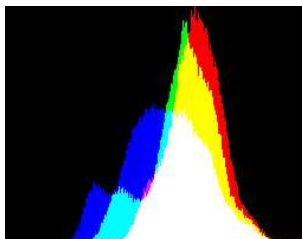


Figure 3: A Sample of an RGB Histogram

A	B	C
D	E	F
G	H	I

Figure 4: Circular Neighbourhood Features

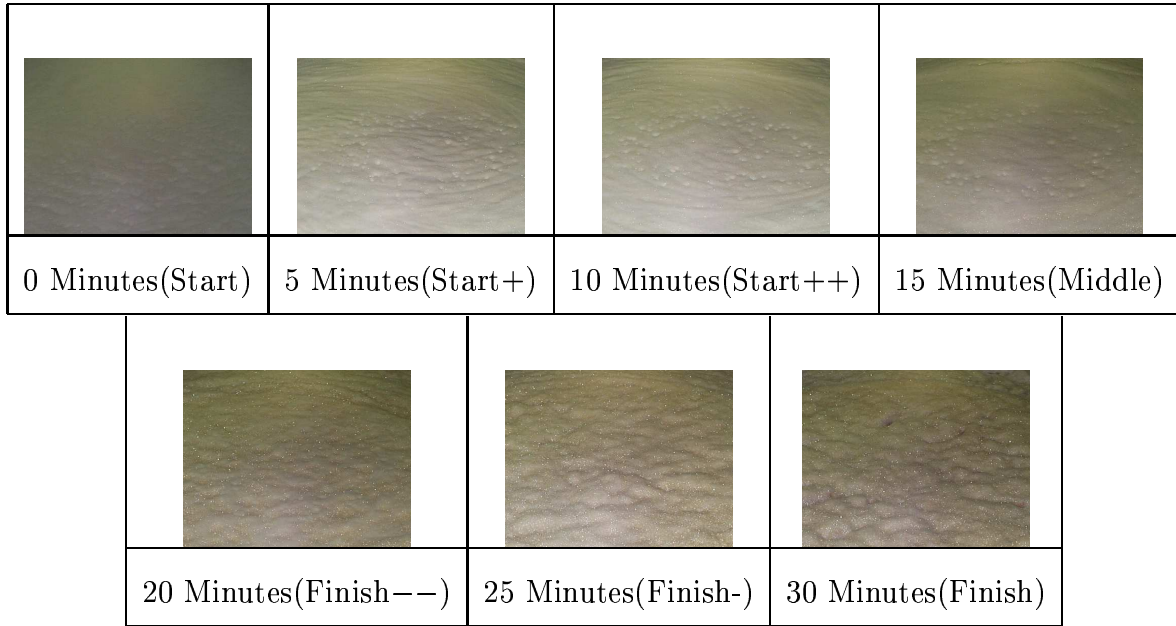


Figure 5: A series of images from the beginning of mashing to finish

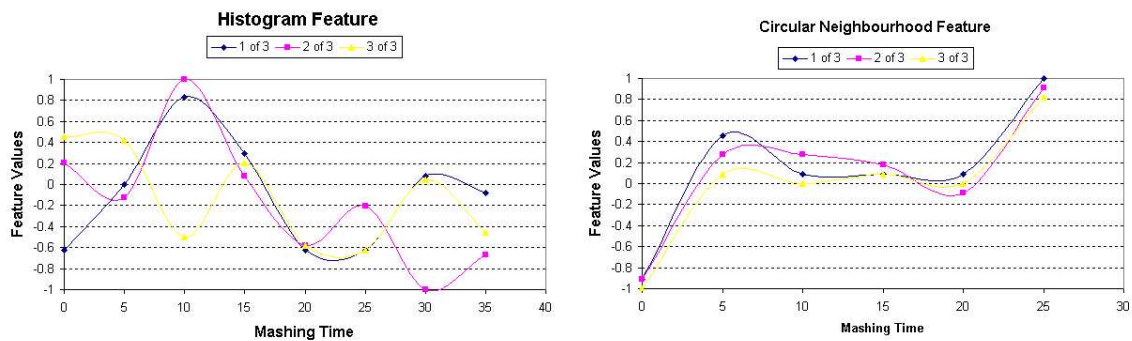


Figure 6: Histogram Features and Circular Neighbourhood Features for a Mash Run