Visual Learning by Set Covering Machine with Efficient Feature Selection

Hiroki Nomiya and **Kuniaki Uehara** [1]

Abstract. In this paper, we propose a new visual learning method for real-world object recognition task. Our method is based on the Set Covering Machine (SCM), to make the learning time shorter than the methods based on commonly used trial-and-error algorithms, such as genetic programming and reinforcement learning. Generally, the process of visual learning is quite time-consuming because image data consists of large amount of information. We attempt to reduce the learning time by introducing the effective feature selection method to find a small number of useful features in image data. Additionally, we introduced a criterion based on the Minimum Description Length (MDL) principle to refine the hypothesis. We perform some experiments to verify the effectiveness of our method.

1 Introduction

Visual learning [5] is widely studied to aquire information and knowledge from images for various object recognition tasks. However, there are some open problems for visual learning. Firstly, most of these methods are based on the trial-and-error learning algorithms, such as genetic programming [4] and reinforcement learning [9]. These methods require a large amount of time to search for the optimal solution. In addition, the learning process is nondeterministic and depends on randomness to some extent. Thus, the result of learning can be rather unstable. Secondly, specialized knowledge is often required to construct the learning algorithm. That is, it is difficult to construct a general-purpose algorithm for visual learning. Thirdly, visual learning learns based on the intensity of each pixel in an image. Thus, it is quite troublesome and time-consuming to find useful information from the data because the number of pixels in an image is very large even if the size of the image is not very large. Finally, some preprocessing is required for the input image data. It is very onerous to preprocess all the input image data.

In this paper, we propose a visual learning method using the Set Covering Machine (SCM) [7][8]. Since the SCM works without trial and error, the cost of searching the optimal solution is relatively small and the result of learning is stable. Additionally, the SCM is a general-purpose learning algorithm. It is applicable to a variety of visual learning tasks without domain-specific knowledge and image preprocessing.

To make our method efficient, we introduce a feature selection method to reduce the number of attributes (i.e. pixels). We define an estimation function to evaluate the usefulness of an attribute and select a small number of useful attributes by the estimation function. By doing this, the computational complexity can considerably be reduced. In addition, we propose the adaptive learning method based on the MDL principle to make the hypothesis concise. We attempt to reduce the computational complexity and make the hypothesis accurate by introducing the adaptive learning. We verify the usefulness of our method by some experiments.

[1] Graduate School of Science and Technology, Kobe University, Kobe, Japan email: nomiya@ai.cs.scitec.kobe-u.ac.jp, uehara@kobe-u.ac.jp

2 Set Covering Machine (SCM)

The SCM [7][8] is a general-purpose learning algorithm that is applicable for various learning tasks. The SCM takes a set of m examples $(\mathbf{x}_1, y_1),\ (\mathbf{x}_2, y_2),\ \cdots,(\mathbf{x}_m, y_m)$ as the training set, where each $\mathbf{x}_i$ is a n-dimensional vector and each y_i is a class label, which is an element of the label space $Y = \{0, 1\}$.

In the SCM, a Boolean-valued function h_i is defined for each example $\mathbf{x}_i$. h_i is called *feature*. A feature is a classifier to be defined as:

$$h_{i,\rho}(\mathbf{x}) \overset{\text{def}}{=} h_\rho(\mathbf{x}, \mathbf{x}_i) = \begin{cases} y_i & \text{if } d(\mathbf{x}, \mathbf{x}_i) \le \rho \\ \overline{y_i} & \text{otherwise} \end{cases} \tag{1}$$

where, $\overline{y_i}$ denotes the Boolean complement of y_i, and $d(\mathbf{x}, \mathbf{x}_i)$ denotes the distance between these two examples. Equation (1) means that if the distance is equal to or less than some real value ρ, the feature $h_{i,\rho}$ outputs y_i, otherwise $\overline{y_i}$. When there is no ambiguity about ρ, we abbreviate $h_{i,\rho}(\mathbf{x})$ to $h_i(\mathbf{x})$.

The hypothesis $f(\mathbf{x})$ of the SCM is expressed as a logical formula. It is represented by conjunctions or disjunctions of features as follows:

$$f(\mathbf{x}) = \begin{cases} \bigwedge_{i \in \mathcal{R}} h_i(\mathbf{x}) & \text{for a conjunction} \\ \bigvee_{i \in \mathcal{R}} h_i(\mathbf{x}) & \text{for a disjunction} \end{cases} \tag{2}$$

For the conjunction case, $f(\mathbf{x}) = 1$, if $h_i(\mathbf{x}) = 1$ for all i, otherwise $f(\mathbf{x}) = 0$. For the disjunction case, $f(\mathbf{x}) = 0$, if $h_i(\mathbf{x}) = 0$ for all i, otherwise $f(\mathbf{x}) = 1$.

To generate an accurate hypothesis, useful features must be selected. In the SCM, the *usefulness* of a feature is defined to determine useful features. The usefulness U_h of feature h is defined as follows:

$$U_h \overset{\text{def}}{=} |Q_h| - p \cdot |R_h| \tag{3}$$

where, in a conjunction (disjunction) case, $|Q_h|$ denotes the number of negative (positive) examples that are correctly classified by h, and $|R_h|$ denotes the number of positive (negative) examples that are misclassified by h. The penalty value p is defined for the tradeoff between the accuracy on the training data and the complexity of the hypothesis. A hypothesis consists of the features that are selected in descending order of their usefulness.

3 Visual Learning with Set Covering Machine

The learning process of our method mainly consists of three components (see Figure 1). First, image filtering to extract useful features of images to generate an accurate hypothesis. There are 17 filters used in our method, consisting of Intel Image Processing Libraries [1] and OpenCV Libraries [2]. Second, learning with the SCM to get a hypothesis. Third, evaluation of hypotheses generated by each filtering to find the best filter. The learning proceeds by repeating these three operations D times, then finally outputs a final hypothesis. D is the number of filters applied to the training images. The parameter is given by the user. The final hypothesis $f_{\hat{D}}$ is the hypothesis which has the highest value of the evaluation function among the hypotheses generated during the learning loop (step 2 to step 5 in Figure 1). Thus, the final hypothesis $f_{\hat{D}}$ is given by:

$$f_{\hat{D}} = \underset{f \in \{f_1, \cdots, f_D\}}{\operatorname{argmax}} E(f) \tag{4}$$

where, E is the evaluation function (defined later).

```
1. Input training images, and $d \leftarrow 1$.
2. For all filters, get a hypothesis with the SCM.
3. Evaluate each hypothesis with the evaluation function.
4. Retain the best hypothesis $f_d$ which has the highest
   value of the evaluation funciton.
5. If $d = D$, then goto step 6.
   Else Apply the best filter to all the training images,
   $d \leftarrow d + 1$, and goto step 2.
6. Output the final hypothesis $f_{\hat{D}}$ which has
   the highest estimation value.
```

Figure 1. The learning process of our method.

Our method uses raw images as input data. An image consists of $(width \times height)$ pixels. Provided that an image is a gray scale image, each pixel has the depth in bits. Its intensity is expressed by a scalar value. The intensity of a n-bit depth pixel ranges from 0 to $2^n - 1$. Thus, an example is represented as $(\mathbf{x}, y)$, where $\mathbf{x}$ is a vector of the intensity values and y is a class label.

We chose to construct a hypothesis with conjunctions of features for the following two reasons: First, it is reported in [7] and [8] that the difference between conjunction and disjunction case in the average classification accuracy is not significant. Second, the computational complexity is almost equal because the algorithms of conjunction and disjunction case are symmetrical. For our method, we define a feature $h_{i,j,\rho}$ by the distance between the intensity of the j-th pixel of two examples as follows:

$$h_{i,j,\rho}(\mathbf{x}) \overset{\text{def}}{=} h_\rho(\mathbf{x}_j, \mathbf{x}_{i,j}) = \begin{cases} y_i & \text{if } |\mathbf{x}_j - \mathbf{x}_{i,j}| \le \rho \\ \overline{y_i} & \text{otherwise} \end{cases} \tag{5}$$

where, $\mathbf{x}_j$ denotes the intensity of the j-th pixel. We abbreviate $h_{i,j,\rho}(\mathbf{x})$ to $h_i(\mathbf{x})$ when there is no ambiguity about j and ρ.

We fixed the penalty parameter p in equation (3) to ∞ because this parameter setting simplifies the procedure for constructing the hypothesis. By this setting, the features that misclassify one or more positive examples are never selected. Thus, a hypothesis consists only of the features that correctly classify all the positive examples. In our method, a hypothesis correctly classifies all the training examples if the two examples that have the same intensity values for all pixels and have opposite class label do not exist.

To make our method efficient, we address the following four points and discuss these points elaborately.

1. Find a small number of useful features to reduce the computational complexity.
2. Evaluate hypotheses to find the best filter to be applied.
3. Search the best sequence of filters efficiently.
4. Extend the algorithm to solve multi-class problems.

3.1 Feature Selection

In case of the first point, we present the method to select useful features by evaluating the usefulness of each attribute. The number of possible features is mp, where m is the number of training examples and p is the number of attributes. The number of attributes is very large even if an image is not very large. However, the number of useful features is small compared with that of useless features. Thus, the computational complexity to find useful features tends to be excessively high. Generally, the more the computational complexity increases, the learning time becomes longer. Hence, the computational complexity can be reduced considerably by selecting useful attributes to reduce the number of possible features.

Figure 2 (a) is an example of a useful attribute. In this case, positive and negative examples are separable. Thus, all the examples are correctly classified by only one feature. In contrast, in case of Figure 2 (b), it is impossible to correctly classify all the examples because there are two examples that have the same intensity values and opposite class labels. Moreover, it is obvious that even if these examples are eliminated, the remainder is not separable by one feature.

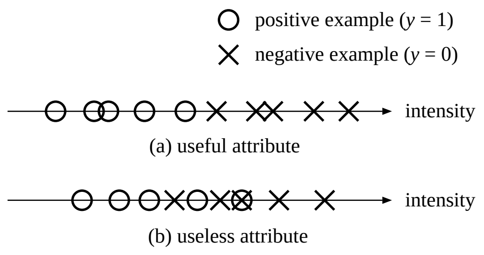

Figure 2. A useful attribute and a useless attribute.

We use the statistics of the training examples as the criterion of the usefulness of an attribute. If the variances of the positive and negative examples' intensity values are small, and the distance between the mean intensity values of the positive and negative examples are large, then the two classes will be separable. Thus, we define the usefulness $u(i)$ of the i-th attribute as

$$u(i) = \frac{s^2(i)}{s^2_{\mathcal{P}}(i) + s^2_{\mathcal{N}}(i)} \tag{6}$$

where, $s^2(i)$, $s^2_{\mathcal{P}}(i)$ and $s^2_{\mathcal{N}}(i)$ are the variance of mean intensity of each class of the i-th attribute, and variances of the i-th attribute's intensity of the positive and negative examples respectively. That is,

$$s^2(i) = (\mu(i) - \mu_{\mathcal{P}}(i))^2 + (\mu(i) - \mu_{\mathcal{N}}(i))^2 \tag{7}$$

$$s^2_{\mathcal{P}}(i) = \frac{\sum_{\mathbf{x} \in \mathcal{P}} (x_i - \mu_{\mathcal{P}}(i))^2}{N_{\mathcal{P}}} \tag{8}$$

$$s^2_{\mathcal{N}}(i) = \frac{\sum_{\mathbf{x} \in \mathcal{N}} (x_i - \mu_{\mathcal{N}}(i))^2}{N_{\mathcal{N}}} \tag{9}$$

where,

x_i : the i-th attribute's intensity of example $\mathbf{x}$
$\mathcal{P}$ $[\mathcal{N}]$: the set of positive [negative] examples
$N_\mathcal{P}$ $[N_\mathcal{N}]$: the number of positive [negative] examples
$\mu_\mathcal{P}(i),\ \mu_\mathcal{N}(i),\ \text{and}\ \mu(i)$:
the mean value of the i-th attribute's intensity of positive, negative, and all examples, respectively.

In our method, if the number of possible features is large, only a few features that have high usefulness are used to generate a hypothesis. The number of features M to be used is determined based on the number of the training images. The maximum number of features M_{max} included in the hypothesis of the SCM is given by

$$M_{max} = \max\{N_\mathcal{N},\ N_\mathcal{P}+1\}. \quad (10)$$

Hence, M_{max} features are considerd to be sufficient to generate a hypothesis and we define M as

$$M \overset{\text{def}}{=} \min\{M_{max},\ a\} \quad (11)$$

where, a is the number of attributes of the training images. The hypothesis $H(\mathbf{x})$ consists of M selected features $\eta_1(\mathbf{x}), \cdots, \eta_M(\mathbf{x}) \in \{h_1(\mathbf{x}), \cdots, h_{mp}(\mathbf{x})\}$, and is given by

$$H(\mathbf{x}) \quad = \quad \bigwedge_{i=1}^{M} \eta_i(\mathbf{x}). \quad (12)$$

3.2 Evaluation of Filters

For the second point, we define an evaluation function based on the information gain [10] to measure the usefulness of a filter. The information gain is a measurement to determine useful attributes to efficiently classify the training examples. It is used for decision tree learning algorithms such as C4.5 [11]. Though the information gain is usually computed for an attribute, in our method, the information gain is computed to evaluate all features included in a hypothesis. Since the information gain is the difference between the entropy of the $(i-1)$-th feature and the weighted mean of the entropy of the i-th feature, we define the information gain for the i-th feature $(i = 1, 2, \cdots, M)$ that

$$Gain(h_i) \overset{\text{def}}{=} Entropy(h_{i-1}) - \frac{N-\nu_i}{N-\nu_{i-1}} Entropy(h_i) \quad (13)$$

where, M denotes the number of generated features, and $Entropy(h_i)$ denotes the entropy of the set $(\mathcal{E} \setminus \mathcal{N}_i)$, where $\mathcal{E}$ is the training set and $\mathcal{N}_i$ is the set of negative examples that are correctly classified by features $h_1, \cdots, h_{i-1}$. The entropy $Entropy(h_i)$ for the i-th feature $(i = 0, 1, 2, \cdots, M)$ is given by:

$$\begin{aligned} Entropy(h_i) \quad = \quad & -\frac{N_\mathcal{P}}{N-\nu_i} \log_2 \frac{N_\mathcal{P}}{N-\nu_i} \\ & -\frac{N_\mathcal{N}-\nu_i}{N-\nu_i} \log_2 \frac{N_\mathcal{N}-\nu_i}{N-\nu_i} \end{aligned} \quad (14)$$

where,

N : the number of training examples
n_i : the number of negative examples correctly classified by the i-th feature h_i
ν_i : the sum of $n_1, \cdots, n_i$, that is $\sum_{j=1}^{i} n_j$, and ν_0 is defined to be 0.

We define the evaluation function E as the weighted sum of the information gain of all the features as follows:

$$E \overset{\text{def}}{=} \sum_{i=1}^{m} \frac{n_i}{N_\mathcal{N}} Gain(h_i). \quad (15)$$

3.3 Search for the Best Sequence of Filters

For the third point, we introduce a method to efficiently search for the best sequence of filters. The learning process of our method can be regarded as a search for the best sequence (or combination) of filters. There are n^D possible sequences of filters, and the search space is represented as a tree structure shown in Figure 3. In Figure 3, the nodes in i-th depth are the candidates for the i-th filter.

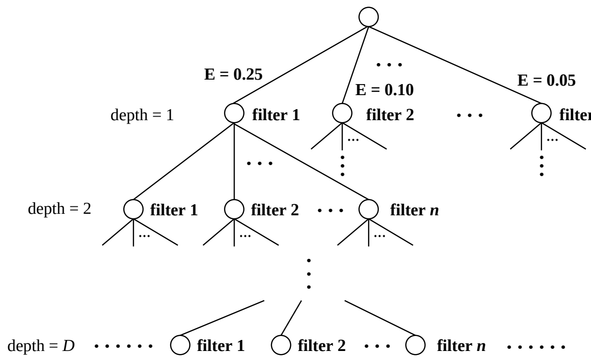

Figure 3. The tree structure of the search space.

The tree consists of $\sum_{i=0}^{D} n^i$ nodes including leafs, where n is the number of filters. Since the number of nodes in the tree increases exponentially as D increases, searching the overall nodes of the tree requires quite large amount of time.

Hence, we use *beam search* which is commonly used for efficient search to reduce the computational complexity by restricting the search space (i.e. the number of nodes). The maximum number of nodes retained by beam search is called *beam_width*. If the number of nodes to be retained is larger than the value of *beam_width*, then beam search does not open the nodes which has low estimation values. If a certain node has a low estimation value, then its child nodes also seem to have low estimation value. Thus, by applying beam search, the nodes that have low estimation values will not be opened. For example, in Figure 3, the node “filter 2” at depth 1, whose value of E is 0.10 and the node “filter n” at depth 1, whose value of E is 0.05 will not be opened.

3.4 Extension to Multi-class Problems

For the fourth point, we describe the method to solve multi-class problems. Since the SCM originally solves only binary (two-class) problems, when a classification problem contains more than two classes, we must extend the SCM to solve multi-class problems. To deal with multi-class problems, we used the *one-against-all* method. In the one-against-all method, k hypotheses are obtained by executing the learning algorithm k times, where k denotes the number of classes. The i-th hypothesis is generated by regarding all of the examples that belong to the i-th class as positive examples, and the remainder as negative examples.

4 Experiments

We performed some experiments using the synthetic aperture radar (SAR) data set in the MSTAR SAR database [3]. The learning task is to recognize the SAR images of the three different objects (i.e. classes): BMP-2 Armored Personnel Carrier, BTR-70 Armored Personnel Carrier, and T-72 Main Battle Tank. We used 585 images, and the set of images has been split into disjoint training set and test set. All images in the training set and the test set are regarded as a gray scale (8-bit depth) image, and resized to 48×48 pixels.

We carried out some experiments and the results are shown in Figure 4. For the confirmation of the effectiveness of feature selection and beam search, we show the results with four different settings: the results with feature selection and beam search (feature selection + search), with feature selection only (feature selection), with beam search only (search), and without feature selection and beam search (none). We fixed the value of *beam_width* to 10.

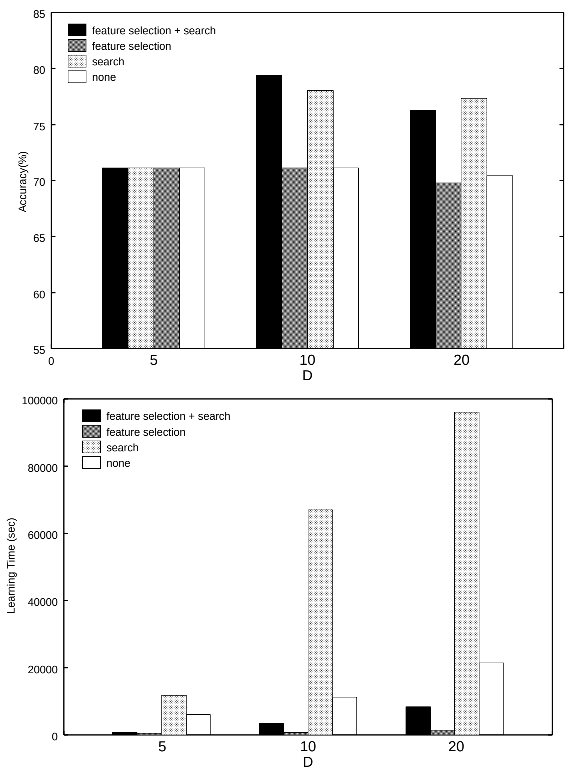

Figure 4. The results of experiments.

Compared with the results without feature selection, the learning time is considerably shortened by feature selection, whereas the classification accuracy is almost equal. By using feature selection, the number of features used to generate a hypothesis is reduced from 2304 to 98. It is obvious that useful features are selected by feature selection. With a small number of features, it is possible to generate a hypothesis which is as accurate as the hypothesis generated by using all the attributes. From the results, it can be said that beam search is effective to improve the classification accuracy, but the search requires relatively long learning time. Provided that beam search is carried out, the classification accuracy with $D = 10$ is fairly high compared with the classification accuracy with $D = 5$. However, the classification accuracy with $D = 20$ is almost equal to the classification accuracy with $D = 10$. This implies that to some extent the classification accuracy increases in proportion to the value of D. However, assigning too high a value to D will make the learning time quite long. Moreover, it will not further improve the classification accuracy of the hypothesis. Thus, we must take into consideration the tradeoff between the classification accuracy and the learning time. It is desirable to find some criterion to determine the optimal parameter setting.

5 Adaptive Learning Based on the MDL Principle

In our method, the SCM continues learning until there are no more negative training examples that can be correctly classified. A feature that correctly classifies many negative training examples generally makes a hypothesis accurate. In contrast, a feature that correctly classifies few negative training examples is generally useless to generate an accurate hypothesis.

Hence, we propose to introduce the adaptive learning into our method. To put it concretely, in the step 3 of Figure 1, the hypothesis is evaluated with a cost function. If the value of the cost function of the hypothesis is higher than that of the previous hypothesis, the SCM stops learning (i.e. go to step 6). By introducing the adaptive learning, the improvement of our method's performance is expected.

We defined the cost function C based on the stochastic complexity of the MDL principle [6]. The MDL principle provides a criterion for the tradeoff between the simplicity of the model and the model's fitness for the data. One is called *model description length* and the other is called *data description length*. On one hand, the simpler the model, the smaller is the model description length. On the other hand, the better the model's fitness for the data, the smaller is the data description length. In our case, the model description length increases in proportion to the number of features and training examples. The data description length is proportional to the rate of misclassification. The MDL principle asserts that the best model poses the minimum value of the sum of the model desciription length and the data description length. By introducing the MDL-based criterion into our method, useless features will be excluded from the hypothesis, then our method can be improved.

The cost function $C(h_i)$ of the i-th feature is defined as

$$C(h_i) \overset{\text{def}}{=} MD(h_i) + DD(h_i) \tag{16}$$

where, $MD(h_i)$ and $DD(h_i)$ are the model description length and the data description length of the i-th feature defined as

$$MD(h_i) \overset{\text{def}}{=} \frac{i}{2}\log_2 N \tag{17}$$

$$\begin{aligned} DD(h_i) \overset{\text{def}}{=} & \sum_{j=1}^{i} n_j \log_2 \frac{n_j}{N_{\mathcal{N}}} \\ & -(N_{\mathcal{N}} - \nu_i)\log_2 \frac{N_{\mathcal{N}} - \nu_i}{(\varepsilon - i)N_{\mathcal{N}}} \end{aligned} \tag{18}$$

where, ε is the estimation of the number of features to be included in the hypothesis. Since the value of $f(i) = (N_{\mathcal{N}} - \nu_i)$ tends to decrease exponentially as i increases, we estimate the value by the function $f(i) = ae^{bi} - 1$. The coefficients a and b are determined by the two conditions: $f(0) = N_{\mathcal{N}}, f(1) = N_{\mathcal{N}} - n_1$. Thus, we defined ε experimentally that

$$\varepsilon \overset{\text{def}}{=} \frac{\ln(N_{\mathcal{N}}+1)}{\ln(N_{\mathcal{N}}+1)-\ln(N_{\mathcal{N}}-n_1+1)} \tag{19}$$

here, $\ln x$ denotes the natural logarithm of x, and $f(\varepsilon)=0$.

We performed some experiments using the same data set and the same parameter settings as the experiments in section 4. To verify whether the adaptive learning improves the performance of our method, we show the results with four different settings: the results with adaptive learning and beam search (adaptive + search), with adaptive learning only (adaptive), with beam search only (search), and without adaptive learning and beam search (none). For all experiments, feature selection was carried out. The results are shown in Figure 5.

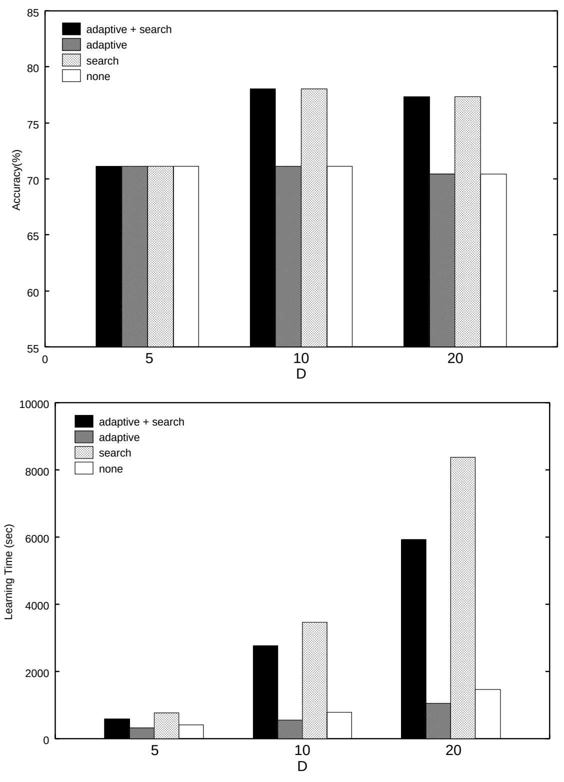

Figure 5. The results of experiments.

By introducing adaptive learning, the average number of features included in the hypothesis reduced from 27.7 to 13.3, and thus the learning time also reduced by 20 $\sim$ 30%. It is clear from the results that the reduction of the number of features is effective to reduce the computational complexity. However, the classification accuracy did not change for all experiments. This result implies that the features which correctly classify few negative training examples have small influence to overall classification ability of the hypothesis. The parameter ε may be connected with the performance of the adaptive learning. The number of features included in the hypothesis decreases in proportion to the reduction of ε, and the value of ε sometimes underestimated. More precise estimation of the value of ε may be effective to improve the adaptive learning method. From the experimental results, it is implied that the adaptive learning is effective in improving the computational complexity of our method.

6 Conclusions and Future Work

We have proposed the visual learning method by the SCM algorithm [7][8] and introduced the effective feature selection method. Additionaly, we have introduced the beam search to reduce the computational complexity. We have verified by some experiments that feature selection considerably shortens the learning time without the degradation of the classification accuracy. We confirmed from the result that feature selection enables to find only appropriate attributes which make the hypothesis accurate. Thus, our method is effective in learning time compared with commonly used trial-and-error-based algorithms. Since the search makes a hypothesis more accurate, our method is also effective in classification accuracy.

Though our method is highly efficient, we did not find the optimal parameter settings for the maximum number of filters D and *beam_width* for the tradeoff between the classification accuracy and the learning time. While it is very difficult to find the optimal parameter settings because the optimal settings generally depend on the given data set, we should define some criterion for the tradeoff.

Since the images in the SAR data set are rather noisy, we confirmed from the results of experiments that our method works well for noisy data set. However, it is not sufficient to prove that our method is robust to noisy data. We should carry out more experiments using noisy data set and make our method stabler.

In chapter 5, we have introduced the MDL-based cost function to improve the performance of our method by eliminating useless features. By using the cost function, the search for features is terminated at the proper point, then the hypothesis consists of only useful features. As a result, the learning time is shortened and the classification accuracy is not changed.

Finally, our method constructs a hypothesis using conjunctions of features because conjunctions and disjunctions are almost equal in average classification accuracy [8]. However, it is possible that disjunctions achieve higher performance on particular data sets. We should confirm whether the performance can be improved with disjunctions and find a criterion for making a selection from conjunctions and disjunctions.

REFERENCES

[1] Intel Corporation. Intel image processing library: Reference manual, 2000.

[2] Intel Corporation. Open source computer vision library: Reference manual, 2001.

[3] Center for Imaging Science. MSTAR SAR database, 1997.

[4] K. Krawiec, 'Pairwise comparison of hypotheses in evolutionary learning', *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*, 266–273, (2001).

[5] K. Krawiec and B. Bhanu, 'Learning by evolutionary feature synthesis', *Proceedings of the Twentieth International Conference on Machine Learning (ICML2003)*, 376–383, (2003).

[6] H. Li and N. Abe, 'Generalizing case frames using a thesaurus and the MDL principle', *Computational Linguistics*, **3**, 217–244, (1998).

[7] M. Marchand and J. Shawe-Taylor, 'Learning with the set covering machine', *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*, 345–352, (2001).

[8] M. Marchand and J. Shawe-Taylor, 'The set covering machine', *Journal of Machine Learning Research*, **3**, 723–746, (2002).

[9] J. Peng and B. Bhanu, 'Closed-loop object recognition using reinforcement learning', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**, 139–154, (1998).

[10] J. R. Quinlan, 'Induction of decision trees', *Machine Learning*, **1**, 81–106, (1986).

[11] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.