속성화된 관계 그래프 표현을 이용한 무제약의 손으로 그린 기호 인식을 위한 모델 학습 기법

이 성환, 김 진형

한국과학기술원 전산학과 인공지능 연구실

A Model Learning Scheme for Unconstrained Handprinted Symbol Recognition Using Attributed Relational Graph Representation

Seongwhan Lee and Jin H Kim

Artificial Intelligence Lab , Dept of Computer Science, KAIST

요약

모델 기반 물체 인식 시스템(Model-Based Object Recognition System)에서 자동적인 모델 학습에 관한 연구는, 보다 융통성있는 물체 인식 시스템을 개발하기 위한 필수 불가결한 연구이다 본 논문에서는 속성화된 관계 그래프(Attributed Relational Graph) 표현을 이용한 손으로 그린 기호 인식을 위한 모델 학습 기법이 소개된다 제안된 모델 학습 기법은 다양한 공학 도면 또는 문서에 나타날 수 있는 여러 종류의 기호들이 임의의 크기를 갖거나, 임의의 각도로 기울어져 있어도, 이들을 인식할 수 있는 실험적 인식 시스템에 결합되었으며, 실험을 통하여 제안된 모델 학습 기법이 손으로 그린 기호의 인식과 같은 부류의, 속성화된 관계 그래프에 의해서 표현되는 기하학적 모델의 학습에 매우 유용함이 밝혀졌다

1. Introduction

In this paper, we describe a model learning scheme for handprinted symbol recognition system capable of reading a wide variety of isolated and unconstrained symbols appearing on various engineering drawings and documents. This system has been successfully applied to the problem of recognizing handprinted symbols in engineering drawings (Lee and Kim, 1988a), and that of verifying Chinese seal imprints automatically (Lee and Kim, 1988b). In this system, a symbol is represented by a set of vertices and segments A vertex is an end or a joining point, and a segment is a straight line or a curved line segment between vertices in a single-pixel-width line-representation of symbol images. To both the vertices and the segments, various attributes may be attached. This representation is called attributed relational graph (ARG) (Bunke and Allermann, 1983, Wong and You, 1986). Symbol recognition is equivalent to ARG matching when the observation as well as models are represented in ARG. Therefore, the problem of recognising an observed symbol can be formulated as the problem of matching an observed ARG (ARG_O) against the model ARG's (ARG_M's) and selecting ARG_M with the maximum similarity. The similarity measure is based on the degree of the match of the individual segment and vertex correspondences.

Our approach begins with an assumption that a particular set of vertices, called control vertices, are found reliably in the observation The use of control vertices yields two advantages First, the amount of data to be processed is reduced by considering only control vertices Therefore, the problem becomes tractable. Second, undesirable errors which may result from dealing with unstable vertices are minimized. In our current implementation, control vertices are selected manually from the ARG_M's during the model learning stage. In doing so, a few considerations are made· First, control vertices must be stable This means they must have a high probability of occurrence and, therefore, they are almost always present in ARG_O's Second, control vertices must contain reliable information about the pose of the ARG_O. In other words, they should be in their general position in 2-D space, i e., they should not be in collinear position nor too close to each other

2. Definition of ARG Representation

Attributes of the ARG are typically numerical measurements of the local structure formed by the primitives (vertices and segments) in a single-pixel-width line-representation of symbol images The position, the type (end, corner or junction), the list of strokes attached to this

vertex, and the list of angles of strokes attached to this vertex are used as attributes for the vertex The length, the curvature, and two vertices it connects are used as attributes for the stroke

Quantitative information is attached to these attributes and used to improve the efficiency of matching by constraining possible structural correspondences, as well as to improve the matching accuracy and reliability. These attributes also make it possible to calculate the degree of the match. So they give a probabilistic basis to how good a vertex or a segment matches The ARG_M is constructed during the learning stage and contains statistical attributes of the vertices and segments For the segment, these are the number of occurrences, mean length, the variance in the length, mean curvature, and variance in curvature. For the vertex, these are the number of occurrences, the mean position, the variance in position, and the histogram of the number of segments.

3. Attributed Relational Graph Matching

In the ARG matching procedure, an unknown ARG_O is matched to the different ARG_M's and the ARG_O is classified as the ARG_M with the minimum distance The normalized positions of the vertices in the ARG_O are defined by fast minimum square error transform (Lee and Kim, 1988c) That transform is described by a parameter vector, (r_0, s_0, θ, c) where r_0, s_0 is the translation, θ is the rotation, c is the scale, such that the image (x_i, y_i) of an arbitrary point (r_i, s_i) of the ARG_M is given by the following set of equations

$$x_i = c \cos\theta (r_i - r_0) + c \sin\theta (s_i - s_0)$$
$$y_i = -c \sin\theta (r_i - r_0) + c \cos\theta (s_i - s_0)$$

Given an ARG_O, a hypothesis (i e , a prediction of the position of the ARG_O with respect to the ARG_M) is generated by matching a set of control vertices in the ARG_M to a set of compatible vertices in the ARG_O

The process of ARG matching proceeds as follows (Lee and Kim, 1988a) First, an ARG_O is constructed from a single-pixel-width line-representation of an observed symbol Second, the pose of the ARG_O is estimated in terms of translation, rotation and scale with respect to the ARG_M's The pose estimation is based on the minimum square error minimization For fast computation, we use a new computational method for the minimum square error transform (Lee and KIm, 1988c) The search space is effectively pruned by introducing the concept of control vertex and applying geometrical constraints in an early stage In this step, a small number of candidate ARG_M's are selected Third, a correspondence between the primitives in the normalized observed ARG (ARG_O^N) and those of the ARG_M is found for the given pose The ARG_O^N is the geometrically transformed ARG according to the minimum square error transform parameters with respect to the ARG_M Fourth, distance measures between the ARG_O^N and the ARG_M's are calculated, based upon the correspondences Finally, the ARG_O^N is classified as the ARG_M with the minimum distance

4. A Distance Measure between ARG's

After finding the correspondence between the primitives in the ARG_O^N and those of the ARG_M, the distance measure is calculated for all the segments and vertices of the ARG_O The distance measure is computed by the contributions of each segment and vertex of the ARG_O^N The distance of a segment match consists of three parts based upon the chance of occurrence of the length of the segment ($p_l(S_i)$), the chance of occurrence of the curvature of the segment ($p_c(S_i)$), and the stability of the segment ($p_s(S_i)$) When a segment is not matched, a minimum likelihood is assigned The distance of a vertex match consists of three parts based upon the chance of occurrence of the distance between model and observed vertex ($p_d(V_j)$), the chance of occurrence of the number of segments attached to this vertex ($p_n(V_j)$), and the stability of the vertex ($p_s(V_j)$)

$p_l(S_i)$, $p_c(S_i)$ and $p_d(V_j)$ are computed in terms of a set of likelihoods indicating the conditional probability that a primitive (vertex or segment) will have some particular value for some attribute in the ARG_O, given that its corresponding attribute in the ARG_M has some particular probability distribution In other words, the likelihood of a given mapping function from the attributed primitive descriptions and statistical knowledge of the distortion processes was assessed in terms of transition probabilities (or densities) acting on the attribute values. These conditional probabilities were approximated by modeling the probability of some attributes of a primitive by a Gaussian distribution $p_s(S_i)$ and $p_s(V_j)$ are computed as the number of occurrences of each S_i and V_j divided by the total number of learning samples, and $p_n(V_j)$ is computed as the number of segments attached to V_j divided by histogram of same number of segments in ARG_M

The final result of matching is a correspondence list and a numerical measure of the distance of the model and the observation The distance of the ARG_O^N and the ARG_M is measured as follows

$$DIST(ARG_O^N, ARG_M) = -\sum_i(\log p_l(S_i)+\log p_c(S_i)+\log p_s(S_i)) - \sum_j(\log p_d(V_j)+\log p_n(V_j)+\log p_s(V_j))$$

5. Learning Model ARG's

The ARG_M's are constructed by a model learning stage. From an ensemble of observations, which is called the "learning set", and, in our case, is a collection of images of symbols, the structural components and their attributes are extracted to construct model Learning is performed on the learning set, that is, the images contain only an example of the pattern class to be learned. This helps to assure that only the "significant" set of control vertices are found in the ARG_M

5.1 Learning algorithm

The learning algorithm begins with an empty ARG_M. For the first symbol image in the learning set, control vertices are selected by the human supervisor and repeats the following procedures for each sample symbol image in the learning set:

(1) A pose (translation, rotation, scale) is obtained which places the ARG_M at the position, orientation, and the scale of the pattern in the sample image This pose is hypothesized by the system and confirmed by the human supervisor.

(2) For each control vertex in the ARG_M, the correspondence to the most likely control vertex in the ARG_O is determined and displayed

(3) As each correspondence is found, the mean and standard deviation of the probability distributions for the attributes are incrementally calculated, and the probability of occurrence for the control vertices is updated.

(4) The most likely correspondences are then found for the rest of ARG_O, and their probability distributions and probability of appearances are also updated

(5) If a part of the ARG_O being learned is not present in the ARG_M, the ARG_M may be updated and learned by adding the unmatched part to the ARG_M This addition starts with the matched vertices of the unmatched segments.

5.2 Learning program

Learning is currently done with an interactive program that uses a graphic monitor Our philosophy has been to begin with a program in which the user must verify each step, and to incrementally automate the learning process as confidence and experience are gained with each stage

The screen of the monitor is divided into four windows In the upper left is a "model" window, in which a copy of the first learning sample is shown The image of the current learning sample is shown in the upper right window At the bottom of the screen a set of text fields are maintained that provide information about learning

Most of learning involves specifying control vertices that match. Control vertices in both the ARG_O and the ARG_M are indicated by drawing cursors (crosses) in the overlay plane, over the model window or the learning images window. Examples of the screen of the interactive learning program which employs advanced interfacing features including menu systems and multiple window capabilities are shown in Figure 1.

5.3 Correspondence matching in learning

As with matching, the basic problem in learning is to find a correspondence between vertices in an ARG_O and vertices in the ARG_M In some ways, the problem is easier than the general matching problem because it is known *a priori* that the learning sample is an instance of the model, and because restrictions can be made about the "cleanness" of the images used for learning In other ways, the problem is harder, however, because during learning the model is only partially constructed and may not be useful in finding the model is only partially constructed and may not be useful in finding the best correspondence.

Our initial implementation of the learning stage is based on the assumption that the learning images are "clean", that is, the example of the pattern to be learned is the only thing in the image. In this way, the control vertices and the rest of ARG to be learned can be easily found The user is asked to approve this correspondence The correspondence for the control vertices determines the pose of the ARG to be learned

In the learning procedure, the statistical attributes are updated after the matching and classification If a part of the ARG_O being learned is not present in the ARG_M, the ARG_M may be updated by adding the unmatched part to the ARG_M. This addition starts with the matched vertices of the unmatched segments

6. Concluding Remarks

In this paper, we described a model learning scheme for handprinted symbol recognition using ARG representation. The proposed model learning scheme has been incorporated into the experimental recognition system which is capable of reading a wide variety of isolated and unconstrained symbols appearing on various engineering drawings and documents Experimental results revealed that the proposed scheme is very useful for learning geometrical models represented by ARG's, as it is the case of handprinted symbol recognition

References

H. Bunke and G. Allermann, "Inexact graph matching for structural pattern recognition," *International Journal of Pattern Recognition Letters* **1** (1983) 245-253

S. Lee and J H Kim, "Recognising Hand-Drawn Symbols in Engineering Drawings", *Artificial Intelligence in Engineering: Robotics and Processes*, Computational Mechanics Publications, Southampton (1988a) 179-200

S. Lee and J.H. Kim, "Automatic Verification of Seal Imprints Using Attributed Stroke Graph Matching," *International Journal of Pattern Recognition Letters*, 1988b. (Accepted for Publication)

S Lee and J H Kim, "A Fast Computational Method for Minimum Square Error Transform," *International Journal of Pattern Recognition Letters*, 1988c. (Accepted for Publication)

A.K C. Wong and M L You, "Entropy and Distance of Random Graphs with Application to Structural Pattern Recognition," *IEEE Trans Pattern Analysis and Machine Intelligence* **7** (1986) 599-609.

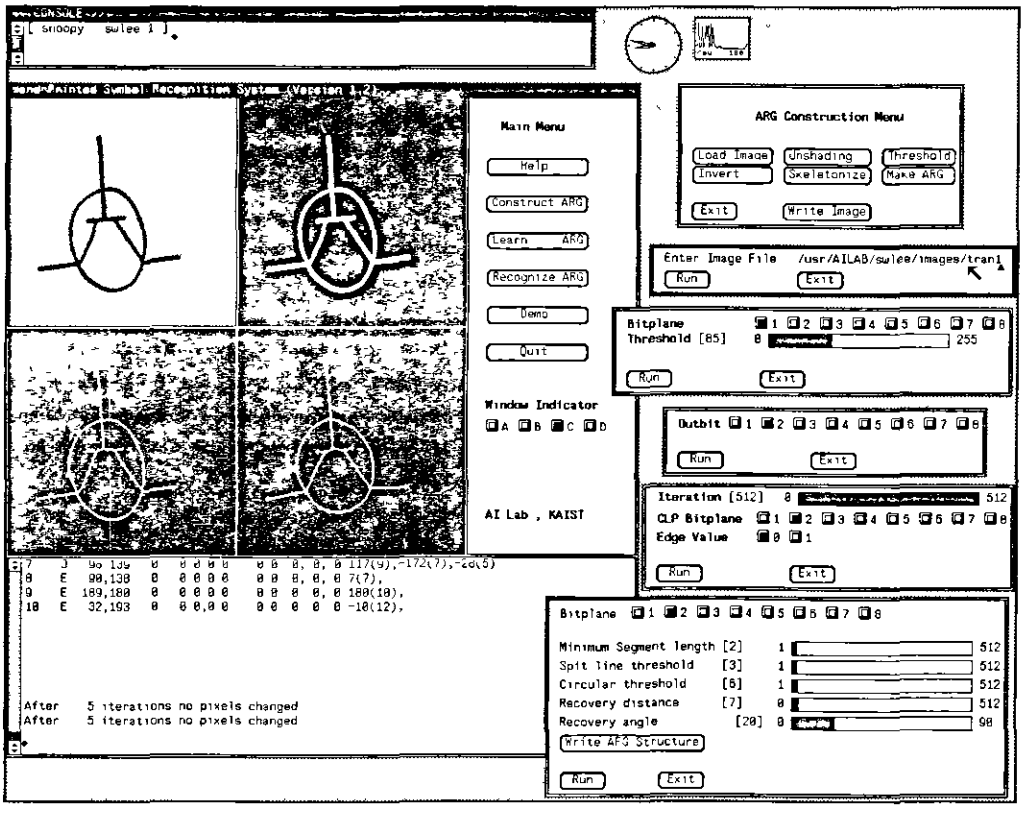

Figure 1 A typical example of the screen of the interactive model learning program which employs advanced interfacing features including menu systems and multiple window capabilities