4th Asian Conference on Computer Vision (ACCV 2000), Taipei, Taiwan, January 8-11, 2000

Integrating Multiple Classifiers In Visual Object Detectors Learned From User Input

Alejandro Jaimes and Shih-Fu Chang

{ajaimes, sfchang}@ee.columbia.edu

Image and Advanced TV Laboratory, Department of Electrical Engineering
Columbia University, 1312 S.W. Mudd Building Mail Code 4712 Box F8
New York, NY 10027, USA

ABSTRACT

There have been many recent efforts in content-based retrieval to perform automatic classification of images/visual objects. Most approaches, however, have focused on using individual classifiers. In this paper, we study the way in which, in a *dynamic* framework, multiple classifiers can be combined when applying *Visual Object Detectors*. We propose a hybrid classifier combination approach, in which decisions of individual classifiers are combined in the following three ways: (1) *classifier fusion*, (2) *classifier cooperation*, and (3) *hierarchical combination*. In earlier work, we presented the *Visual Apprentice* framework, in which a user defines visual object models via a multiple-level *object-definition hierarchy* (*region, perceptual-area, object part, and object*). As the user provides examples from images or videos, visual features are extracted and multiple classifiers are learned for each node of the *hierarchy*. In this paper, we discuss the benefits of hybrid classifier combination in the *Visual Apprentice* framework, and show some experimental results in classifier fusion. These results suggest possible improvements in classification accuracy, particularly of detectors reported earlier for Baseball video, images with skies, and images with handshakes.

Keywords: machine learning, combination of classifiers, content-based retrieval, detection of visual objects.

1. INTRODUCTION

In recent years, many new techniques have been developed to automatically index visual content. Some of these techniques are based on similarity or query-by sketch approaches (e.g., an image that looks like another one; an image that resembles a drawing). Examples of systems that use similarity search or query-by-sketch techniques include QBIC, and VisualSEEk [4].

Other recent work has focused on the automatic extraction of higher level descriptions of the visual content, via classification. In that scenario, the image or video is automatically placed into a semantic category. Examples of this approach include the classification of images according to scenes (e.g., indoor vs. outdoor [12], city, landscape [13]), and objects (e.g., naked people and horses [5]).

As pointed out in [7], one of the problems with those approaches is that they are *static*: many of their components are built by hand and cannot be changed. In order to successfully build general detection systems (for objects or scenes), it is imperative to have *dynamic* frameworks in which different components adapt to the task at hand, and where such components interact to exploit the structure of elements to be detected. One step in that direction is to use machine learning techniques to build classifiers that can automatically label content.

The next step is to exploit the structure present in the objects to be detected, and to allow the interaction of different hypotheses during the detection process. Typically, hypotheses are formed by means of classification: in general, an instance *x* (i.e., an image or video) can be represented by a feature vector $v=f_1,\ldots,f_n$. The feature vector serves as input to a classifier function *f(v)*, which outputs a label *l* that determines *x*'s classification. Classification is often performed using an individual classifier: the decision regarding the class of *x*, is made by a single expert (i.e., classification function).

In some cases, however, it may be beneficial to decide on the class of *x*, based on a combination of distinct classification functions. This can lead to better accuracy and more robustness. The final decision, then, depends on the outputs of the individual classifiers and on the strategy used to combine those outputs. The simplest classifier combination scenario is depicted in figure 1.

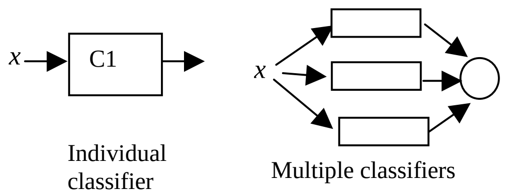

Figure 1. Individual classifier function, and classifier combination. Note that each classification function could use a distinct feature vector to represent instance *x*.

In building detectors for objects or scenes though, there are many other possible ways to combine classification functions- some of these strategies have their origin in the machine learning community, in statistics, and even in studies of how the human visual system works.

In this paper, we study the ways in which multiple classifiers can be combined when building *Visual Object Detectors*. We propose a hybrid approach that combines several distinct classifiers using the framework of the *Visual Apprentice* [6]: a user defines visual object models according to his interests via a multiple-level *object-definition hierarchy* (*region, perceptual-area, object part,* and *object*). As the user provides examples from images or video, visual features are extracted and multiple classifiers are learned for each node of the *hierarchy*. In our novel hybrid combination approach to detect visual objects, classifiers interact in different ways at different levels. Classifiers are combined in the following three ways (figure 2):

- *Classifier fusion:* individual classifiers are applied to the same input, to achieve a form of consensus.
- *Classifier cooperation:* individual classifiers are influenced by other individual classifiers, which take different inputs.
- *Hierarchical classification:* classifiers decide the inputs to other classifiers.

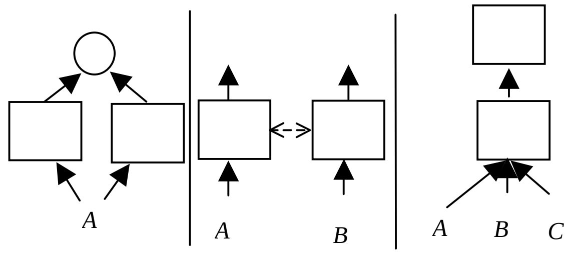

Figure 2. Classifier combination strategies.

In the first case (*fusion*), the decisions of two classifiers are combined. In the second one (*cooperation*), independent classifiers influence each other. In the third scenario (*hierarchical*), the decision of a classifier affects the input to a second classifier.

Our approach differs from previous work in which automatic classification is performed using an individual classifier[1]. The body plans approach [5], for example, performs classification based on a multiple stage process in which each decision is made by a single classifier function. In the indoor vs. outdoor approach presented in [12], individual blocks are classified by different classifiers, whose outputs are combined via stacking. In our previous work [6][7], only hierarchical classification takes place. In the work presented in [7], the best single classifiers and feature are selected.

This paper is organized as follows. In Section 2 we give a brief overview of the *Visual Apprentice*. In section 3 we discuss different classifier combination techniques and our proposed hybrid combination approach. Finally we present some experimental results and summarize our approach.

2. LEARNING FROM USER INPUT

2.1 Object Definition Hierarchy

In the *Visual Apprentice* [6], users can build automatic *Visual Object Detectors* according to their interests. Learning is based on an *object-definition hierarchy* consisting of the following levels (fig. 4): (1) *region*; (2) *perceptual*; (3) *object-part*; (4) *object*, and (5) *scene*. The user has the flexibility of defining the object definition hierarchy so that it represents the class he is interested in.

2.2 Training Phase

To construct a *Visual Object Detector*, the user begins by selecting the training images/videos and building an *object-definition hierarchy* according to his interests. As described in [6], our recognition strategy is based on automatic segmentation of training and new images (fig. 3)- during training, each image/video is automatically segmented and example regions are manually labeled by the user according to the *hierarchy* he defined. The labeled regions from all of the training examples are then used to compute the training set: features (color, texture, shape, spatial relationships, etc.) at each node of the *hierarchy* are automatically extracted and stored in a database. This data is then used for training by learning algorithms that yield sets of classifiers at each node of the hierarchy (e.g., grass-regions, sand, etc.).

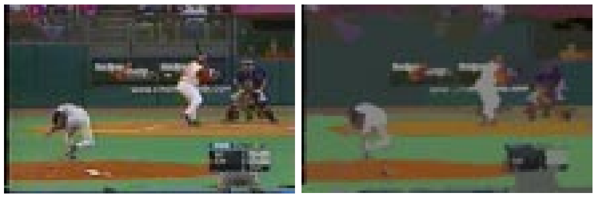

Figure 3. Images and video sequences are automatically segmented during training and classification.

2.3 Classification Phase

The *Visual Object Detector* performs automatic classification by first applying automatic segmentation, and then combining classifiers and

[1] A classifier can be seen as a “black box” which takes as input a set of features and returns a value that determines whether the features correspond to an object in the class in question. The function itself may be a complex one, but the decision is made by a single “expert”.

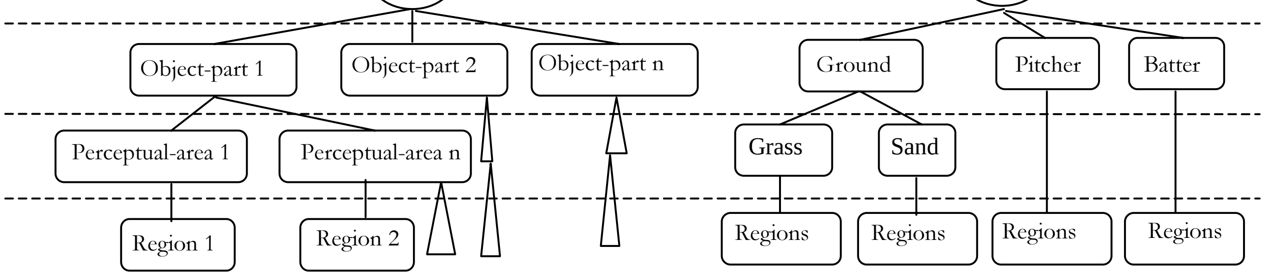

Figure 4. General *Object Definition Hierarchy* and *hierarchy* for the image in figure 3. Every node in the tree has a conceptual meaning (e.g., pitcher), but also corresponds to a set of connected pixels in the image. This example shows how a scene can be modeled using the *hierarchy*.

grouping strategies at the levels of figure 4: *regions* are classified first and combined to obtain *perceptual-areas* which are used by *object-part* classifiers. *Object-parts*, in turn, are combined and passed to *object* classifiers. In the work presented in [6], classifiers interacted only across levels of the hierarchy (e.g., only regions that pass the region level classifier can be seen by the object-level classifier), and the best features/classifiers were automatically chosen in [7]. Figure 6 at the end shows an example of a handshake classifier in which the need for interaction between several classifiers is apparent. In the next section, we describe the hybrid classifier combination scheme proposed in this paper.

3. COMBINING CLASSIFIERS

Several classifier combination methods have been proposed. As mentioned in the introduction (see figure 2), in our framework, there are three basic ways in which classifiers can be combined. In this section, we outline the existing techniques in each modality and present our hybrid combination approach.

When deciding on a classifier combination strategy, several factors should be considered. One of those factors, is each classifier's representation of the input pattern [9]:

- All classifiers use the same representation of the input pattern (i.e., the same feature vector).
- Each classifier uses its own representation of the input pattern. The measurements extracted from the pattern are unique to each classifier.

When classifiers use different input patterns, their combination is often referred to as sensor fusion (e.g., audio and video classifiers). In our scenario, classifier fusion refers to the combination of classifiers that receive the same input pattern. Note, however, that different representations of the pattern may be used. For example, region level classifiers try to determine the class of the regions in the image- a pair of pitcher-region and batter-region classifiers receive as input a set of feature vectors for each region. The features used by each classifier, however, are selected automatically and may be different [7]. The input to both classifiers, nevertheless, is the same pattern.

3.1 Classifier Fusion

Classifier fusion occurs when several experts make a decision regarding the class of the same input pattern. This was shown in figure 1. The outputs of several classifiers, or "experts" are combined according to a decision strategy. Some of the most common classifier combination strategies for classifier fusion are based on rules (for details see [9]):

- Maximum rule
- Minimum rule
- Median rule
- Majority vote rule

Other strategies include the use of decision theoretic approaches (i.e., using Bayes rule [2]), among others.

In the context of our framework, classifier fusion occurs at each node of the object-definition hierarchy. For each class (e.g., sand region in figure 4), several classifiers are combined during classification. For example, the decision on whether a particular region is a sand region or not, depends on the output of several sand-region classifiers. We use the bagging technique and tools implemented in [10]. In that approach, several classifiers vote on the class of the input pattern and the final decision is made depending on the majority vote.

One important question to be raised is whether having multiple binary classifiers is better than having a single multiple-class classifier. All elements at a given level of the hierarchy of figure 4 (e.g., the region level) could be classified using a single multiple-class classifier (e.g., handling the classes grass, sand, pitcher, and batter region). As pointed out in [8], however, constructing and combining small classifiers into a larger network can improve performance and be less time and space demanding than having a single large classifier (see also [1]).

3.2 Classifier Cooperation

In many real-world scenarios, individual classifiers influence each other. An image that contains a tree, for example, is likely to also contain a patch of sky. In that respect, a tree classifier would be "helped" by a sky classifier, and vice-versa. We refer to this interaction as *classifier cooperation*.

In general, classifier cooperation occurs with different types of classifiers (i.e., their concepts are non-overlapping). This differs from the classifier fusion of the previous section, in which different classifiers are meant to cover the same concept (e.g., the decisions of two sky classifiers are combined). In our particular framework, then, classifier cooperation is beneficial between nodes that are not directly linked in the definition hierarchy (e.g., in fig. 4, grass and sand; pitcher and batter).

A classifier trying to detect the pitcher in the scene of figure 3, might "give up" if it knows that the batter classifier did not detect a batter. In a similar way, the batter classifier could be influenced by the decision made by the pitcher classifier. Since classification is performed according to the levels of the hierarchy (first regions are classified, then perceptual-areas, etc.), classifiers can obtain information from other classifiers at different levels.

In figure 5, we observe how this type of cooperation takes place. Since region-level classification is performed first, the classifiers at higher levels can make use of that information (even if they are not directly linked in the hierarchy). The pitcher object-part classifier receives as input information from the batter region-level classifier. The batter object-part classifier also receives information from the pitcher region-level classifier. This information is important to the batter object-level classifier sine for the visual object detector being built the presence of a pitcher implies presence of a batter (for this particular model). If pitcher regions are not found by the pitcher region-level classifier, then the pitcher object-part classifier will not succeed.

Information from lower to higher levels (as depicted by the dashed arrows of figure 5) can be included in training and classification in several ways. The feature vector of each higher level node, for example, can be augmented by indicator features of the lower level nodes. For example, the batter object-part feature vector would include the pertinent features plus some indication features of the pitcher region node- such as how many pitcher regions exist, their area, etc.

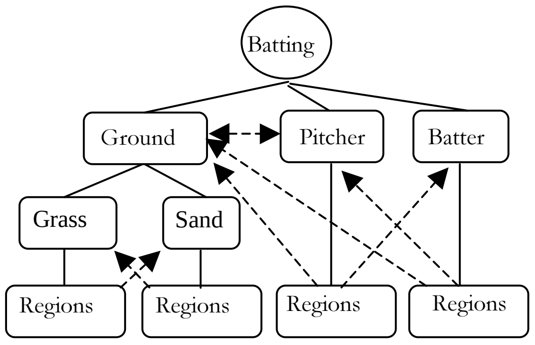

Figure 5. Some examples of classifier cooperation in our framework, as indicated by dotted lines.

3.3 Hierarchical Combination

In the framework of the *Visual Apprentice* presented earlier, hierarchical combination of classifiers occurs across levels of the object-definition hierarchy. The input to a perceptual-area classifier, for instance depends on the classification decisions of the corresponding region-level classifier (e.g., pitcher regions that pass the region level classifier are used by the pitcher object-part classifier).

The interchange of the previous section is different. Again, observing the example of fig. 5, the pitcher object-part classifier during *classifier cooperation* does not directly use the regions found by the grass region-level classifier. It only uses the knowledge that they were found, when deciding on new regions that are likely to correspond to a pitcher. This is an example of meta-classification since the input to a classifier is the output of another classifier.

4. EXPERIMENTAL RESULTS

In this section we present some experimental results on classifier combination in the *Visual Apprentice*.

We use the same training set used in [7]. In that work, *Visual Object Detectors* were built for the batting scene of figure 3, for handshakes, and skies. The *object-definition hierarchy* for the batting scene is similar to the one shown in figure 4, with the difference that four perceptual areas were used for the field object-part: mound, top grass (near the batter), and bottom grass (near the pitcher). We show some detailed results for the batting scene, and general results for the skies and handshakes.

In these experiments, we compare the performance of using single classifiers and using multiple classifiers at each region-level node (*classifier fusion*) of the hierarchy of figure 4. In particular, we used the bagging methodology in [10], where the results of several classifiers are combined through majority voting.

A total of 376 Baseball video shots were obtained from professional TV broadcast (from 6 different games). Out of the 376 shots, 125 were of the batting scene. A set of 60 (out of the 125) batting shots constituted the *training set.* Regions in the key frames for each shot were given the labels in the first column of table 1 (pitcher, top grass, bottom grass, mound, and batter)- all other regions were automatically labeled *unknown*.

That training set originally contained 3,395 regions, including 494 that correspond to each of the classes in the hierarchy (there were approximately 100 examples for each of the labels). Since each binary classifier is trained independently, in that training set, only 100 unknown regions where included (i.e., regions that do not belong to any of the classes of table 1). In other words, the training set consisted of 494 regions with each of the labels of table 1, and 100 regions with the label unknown, for a total of 594 regions in the training set. It is important to note that the original distribution of the data plays an important role in the application of some learning algorithms. In this case, however, each classifier is concerned only with examples in two classes: unknown and the label that it seeks to classify (e.g., the pitcher classifier receives as input positive pitcher regions and unknown regions, which may correspond to regions with other labels). The *training set* was used to build classifiers and 10-fold cross-validation was done to determine the their performance (see [10] for cross-validation accuracy).

The first line of each row in table 1 (in non-italic font) represents the average error rate of the cross-validation process, when bagging was used. The classifiers built using that training data were also tested on the *independent test set,* which was not used at all by the learning algorithms during training. The error over the test set is shown in the second line of each row.

In order to compare the benefits of using the bagging algorithm, the analysis was performed twice. The *italic* fonts represent the results obtained when bagging was *not* used.

Even though in some cases the performance differences were small, in most cases the bagging algorithm did have an improvement over the individual classifier.

We did not include the region-level bagging classifiers when performing the shot level classification we describe next (also reported in [7]). Improvements shown using bagging, however, suggest that overall accuracy improvements can occur.

In each of the experiments performed, the test sets used were independent of the training sets. For the baseball batting scene classifier, 316 shots were used for testing. The test set included 65 batting scene shots, and, as above, an independent training set of 60 shots was used. We obtained 92% accuracy (64% recall, and 100% precision). This is not surprising, since detection of a batting scene implies detection of each of the components- all of the components of the object hierarchy must be present (if they were always provided during training)- it is unlikely to encounter false positives for all of them within a single shot. It is also important to note that a change in the hierarchy (e.g., remove the mound- higher accuracy?) implies changes in performance.

	IB1	**IB3**	**IB5**	**ID3**	**NB**
Pitcher	4.6% 23.1%	4.3% 21.3%	4.32% 20.6%	1.8% 20%	7.9% 29%
	4.60% 23.28%	*4.75% 22.25%*	*4.32% 21.8%*	*2.6% 21.2%*	*8.20% 29.9%*
Top Grass	3.44% 19.88%	4.02% 19.06%	4.01% 18.3%	1.43% 16.9%	4.31% 13.1%
	4.01% 20.12%	*3.88% 19.45%*	*4.45% 19.1%*	*1.73% 17.2%*	*3.88% 13.1%*
Bottom Grass	2.58% 13.71%	2.44% 13.38%	2.72% 13.3%	1.72% 19.0%	4.30% 13.2%
	2.58% 13.75%	*2.72% 13.30%*	*2.87% 13.3%*	*2.87% 21.2%*	*4.88% 13.2%*
Mound	1.74% 16.31%	1.74% 16.31%	2.46% 15.3%	0.43% 18.8%	2.17% 15.4%
	1.45% 16.37%	*2.02% 15.94%*	*2.46% 15.5%*	*0.43% 18.5%*	*2.32% 15.3%*
Batter	1.87% 20.70%	2.44% 20.90%	2.30% 21.3%	1.72% 24.7%	4.31% 20.0%
	1.87% 20.7%	*2.30% 21.23%*	*2.30% 21.95%*	*2.01% 23.40%*	*4.17% 20.16%*

Table 1. Performance of different classifiers for the region-level nodes of figure 4. The first line of each row shows the average error rate for 10-fold cross-validation over the training set. The second line of each row shows the error rate over the independent test set. Rows in *italics* correspond to the performance of the respective classifiers *without* using bagging.

For the handshake tests (figure 6 at the end), we used 80 training images, and an independent test set of 733 news images. Out of the 733 images, 85 were handshakes. We obtained 94% accuracy (74% recall, 70% precision) in a set of 89 images automatically placed in the handshake class. Sky detection was performed on a set of 1,300 images that contained 128 skies. We obtained 94% accuracy (50% recall, 87% precision), in a set of 134 images retrieved.

Note that in these scene-level experiments we only included the hierarchical combination scheme- classifier fusion and cooperation where not tested at the scene level. It is likely, however, that improvements will occur if we use the full combination framework proposed in this paper.

5. SUMMARY

In this paper, we have presented ways in which multiple classifiers can be combined when building *Visual Object Detectors*. We proposed a hybrid classifier combination approach, in which decisions of individual classifiers are combined in the following three ways: (1) *classifier fusion*, (2) *classifier cooperation*, and (3) *hierarchical classification*. We used our earlier *Visual Apprentice* framework, in which visual object models are built with components of a multiple-level *object-definition* *hierarchy* (*region, perceptual-area, object part, and object*).

Our future work includes further testing of each combination component independently and at the scene level, and the use of adaptive techniques for the combination and interaction of classifiers. Evaluation of the cost/benefit associated with each modality is also in an important issue we will address in the future- it is of particular importance to obtain a better understanding of where the errors occur.

REFERENCES

[1] K. Ali, and M. Pazzani, "HYDRA-MM: Learning Multiple Descriptions to Improve Classification Accuracy", *International Journal on Artificial Intelligence Tools*, No. 4, 1995.

[2] K. Ali, and M. Pazzani, "Classification using Bayes Averaging of Multiple, Relational Rule-based Models", *5th International Workshop on AI & Statistics*, Ft. Lauderdale, FL, 1995.

[3] E. Bauer, and R. Kohavi, "An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants", *Machine Learning* Vol. 36, Nos. 1/2, pp. 105-139, July/August, 1999.

[4] S.-F. Chang, J.R. Smith, M. Beigi and A. Benitez, "Visual Information Retrieval from Large Distributed On-line Repositories", *Communications of the ACM*, December, 1997. ACM.

[5] D.A. Forsyth and M. Fleck, "Body Plans", *Proceedings Computer Vision and Pattern Recognition - CVPR* 97, San Juan, Puerto Rico, June 1997.

[6] A. Jaimes and S.-F. Chang, "Model-based classification of visual information for content-based retrieval", *Storage and Retrieval for Image and Video Databases VII*, IS&T/SPIE, San Jose, CA, January 1999.

[7] A. Jaimes and S.-F. Chang, "Automatic Selection of Visual Features and Classifiers", *Storage and Retrieval for Image and Video Databases VIII*, IS&T/SPIE, San Jose, CA, January 2000.

[8] C. Ji and S. Ma, "Combinations of Weak Classifiers", *IEEE Transactions on Neural Networks* Vol. 8, No. 1, Jan., 1997.

[9] J. Kittler, M. Hatef, R. P.W. Dui, and J. Matas, "On Combining Classifiers", *IEEE PAMI*, Vol. 20, No. 3, March, 1998.

[10] R. Kohavi, G. John, R. Long, D. Manley, and K. Pfleger, "MLC++: A Machine Learning Library in C++", *Tools with Artificial Intelligence 94,* 1994.

[11] T. Minka, and R. Picard, "An Image Database Browser that Learns From User Interaction", Technical Report 365, MIT Media Laboratory and Modeling Group, 1996.

[12] M. Szummer and R.W. Picard, "Indoor-Outdoor Image Classification", *IEEE International Workshop on Content-based Access of Image and Video Databases*, in conjunction with ICCV'98. Bombay, India.

[13] A. Vailaya, A. Jain and H.J. Zhang, "On Image Classification: City vs. Landscape", *IEEE Workshop on Content-Based Access of Image and Video Libraries*, June 21, 1998, Santa Barbara, California.

[14] K. Woods, W.P. Kegelmeyer Jr., and K. Bowyer, "Combination of Multiple Classifiers Using Local Accuracy Estimates", *IEEE PAMI*, Vol. 19, No. 4, April 1997.

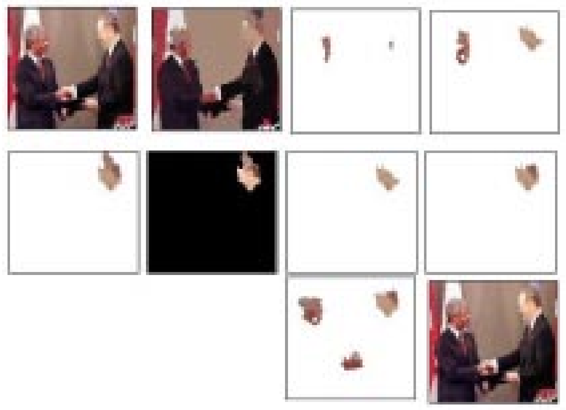

Figure 6. An example handshake classification process.