On Adaptive Confidences for Critic-Driven Classifier Combining

Matti Aksela and Jorma Laaksonen

Neural Networks Research Centre
Laboratory of Computer and Information Science
P.O.Box 5400, Fin-02015 HUT, Finland
matti.aksela@hut.fi, jorma.laaksonen@hut.fi

Abstract. When combining classifiers in order to improve the classification accuracy, precise estimation of the reliability of each member classifier can be very beneficial. One approach for estimating how confident we can be in the member classifiers' results being correct is to use specialized critics to evaluate the classifiers' performances. We introduce an adaptive, critic-based confidence evaluation scheme, where each critic can not only learn from the behavior of its respective classifier, but also strives to be robust with respect to changes in its classifier. This is accomplished via creating distribution models constructed from the classifier's stored output decisions, and weighting them in a manner that attempts to bring robustness toward changes in the classifier's behavior. Experiments with handwritten character classification showing promising results are presented to support the proposed approach.

1 Introduction

In an attempt to improve pattern recognition performance, several approaches can be taken. One approach often found beneficial is to combine the results of several classifiers in the hope that the combination will outperform its members. The basic operation of a committee classifier is to take the outputs of a set of member classifiers and attempt to combine them in a way that improves accuracy. As a committee merely combines the results produced by its members, the member classifiers have a significant effect on the performance. The member classifiers' two most important features that affect the committee's performance are (i) their individual error rates, and (ii) the correlatedness of the errors between the members. The more different the mistakes made by the classifiers are, the more beneficial the combination of the classifiers can be [1].

In order to combine the results as effectively as possible, one approach is to obtain some measure of how confident we can be that a classifier is making a correct prediction. Therefore, for combining classifiers in an intelligent way, it would obviously be beneficial to be able to estimate the reliability of each classifier in an accurate fashion. One way of accomplishing this is to use a separate classification unit to decide whether the classifier is correct or not. Schemes with separate experts for evaluating the classifiers' reliability are often called *critic-driven models*, as each classifier can be assigned a critic that provides a measure of confidence in the classifier's decisions. Critics themselves are specialized classifiers providing an estimate of how likely it is expected that the classifier in question is correct. This prediction can be made based either only on the output of the classifier or also on its inputs.

In pattern recognition the problem is basically to estimate the posterior probabilities of the classes for a given input sample and to choose the class most probably correct. However, estimating the true posterior probabilities is far from simple in many cases, for example when using prototype-based classifiers. The critic-driven approach is one method of attempting to produce accurate estimates on how likely the classifiers are to be correct and thus which label should be chosen. Also, it may be necessary to combine different types of classifiers, and thus it is advantageous to pose as few requirements for the classifiers as possible.

Critic-driven approaches to classifier combining have been investigated previously, e.g. in a situation where the critic makes its decision based on the same input data as the classifier [2] and in a case where scaling schemes and activation functions for critics were examined [3]. Most critic-driven schemes are static in the sense that they have no memory of previous input samples. In general, however, a classifier's performance tends to be similar in similar situations, for example for samples belonging to the same class. It could thus be beneficial to take advantage of this by incorporating also information on the classifier's prior performance into the critic.

Conversely, also the classifier's performance may change in time – for example in the case of handwriting recognition the writer's style may change due to a different situation or we may encounter an entirely new writer. Therefore the critic scheme should be capable of robustness also under changing conditions, a trait somewhat suppressed by the desire to use all collected information for the predictions. Thus it should be advantageous to find a balance between the impact of the older and the more recent samples. In this paper, we propose an approach where information on the classifier's prior performance is used for the critic's decisions while incorporating a weighting scheme to focus on the most recent samples in order to improve the robustness.

2 Adaptive Class-Wise Confidences

In most critic-driven schemes the critic bases its decisions on only the current sample and the member classifiers' outputs. However, the approach presented in this paper is based on our Class-Confidence Critic Combining (CCCC) scheme [4], a model that attempts to learn continuously from the behavior of the critic's associated classifier. Each critic gives its estimate on the classifier's accuracy based not only on the classifier's output to the sample at hand, but also the classifier's prior performance in similar situations. This enables the critic to adapt to the performance of the particular classifier it has been assigned to.

Learning the classifier's performance will inevitably also make the critic less robust to changes in the classifier's behavior, as its evaluations are now by definition based also on the collected knowledge on the classifier's prior performance. To counter this, we in this paper expand the CCCC model further by introducing weighting schemes to make the critics emphasize the most recent inputs. The original model is also modified to produce confidence estimates for all classes from each classifier for each input sample. The output of the combiner is then selected from this array of confidences using one of the standard methods to be presented in Section 2.5.

2.1 The Proposed Approach

In this paper we use prototype-based classifiers that calculate the input's distance from the nearest prototype of each class. Each member classifier produces a vector of distances, with one distance for each class, where a smaller distance indicates a better match. These distances are stored in the critics which attempt to model the distances' distributions. Based on the current and modeled prior distances for the same class, a confidence value is calculated by the critic. These values are then used in deciding the committee's final output. If the classifier is not based on distances, any measure that decreases as similarity increases can be used, or a native confidence-measure may be transformed into a distance for the distribution estimates. For example, if we have a confidence measure $t \in [0, 1]$, we may simply use $1 - t$ as the distance.

Each time a new input sample is processed, it is classified by all member classifiers who calculate the smallest distance to each class. Each classifier's critic then produces an estimate on that classifier's correctness based on the classifier's prior behavior for the same class and that particular input's normalized distances. The final output of the committee is decided from the classifier outputs and confidences obtained from the critics by using one of the combination rules discussed below. After the correctness of the classification has been established, the input is incorporated into the respective critic's distribution model to refine the estimate of the input data distribution. Additionally, weights are used for the distance values stored in the distribution models and adjusted in order to strive for more robust behavior with respect to changes in the classifier's performance.

2.2 Normalizing Distance Values Obtained from Classifiers

As such the distances produced by the member classifiers can be over a wide range of numerical values and have no confidence interpretation on their own. The distances computed in the classifiers may be normalized as follows.

Let there be K classifiers each calculating some type of distances and deciding its output based on minimizing that distance measure. Let there be C pattern classes. Each classifier may have $p \geq C$ prototypes to which the distance is calculated, but at least one prototype for each class. Now let x be the input sample, k the classifier index, $k \in [1, \ldots, K]$, and c the class index, $c \in [1, \ldots, C]$.

From each classifier for each input we may find the shortest distance to the nearest prototype of each class, $d_c^k(x) \in [0, \infty]$.

In practice the classifiers should not produce an infinite distance if matching to any prototype of that class is possible. But for example in the stroke-based handwritten character classifiers used in the experiments presented in this paper, the distance is defined to be infinite if the number of strokes is different in the input and the prototype. Thus also the case of infinite distance should be taken into account. We can now define normalized distances to be used in the critics' distributions as

$$q_c^k(x) = \begin{cases} \frac{d_c^k(x)}{\sum_{i=1}^{C} \hat{d}_i^k(x)} & \text{, if } d_c^k(x) \text{ is finite} \\ 1 & \text{, otherwise} \end{cases} \quad . \tag{1}$$

$\hat{d}_c^k(x)$ used in the summation equals $d_c^k(x)$ if $d_c^k(x)$ is finite and is otherwise zero. However, if the distance to only one class is finite, the normalized distance for that class is defined to be zero. If the distances are close to another, $q_c^k(x)$ becomes relatively large, but if the distance to the nearest prototype is much smaller than the others, the normalized value is notably smaller for that class. If the prototype matches the input sample exactly, also the normalized distance equals zero.

2.3 Distribution Types

In order to obtain confidences for decisions on previously unseen x, the values $q_c^k(x)$ must be modeled somehow. The approach of gathering previous values into distribution models from which the value for the confidence can be obtained as a function of $q_c^k(x)$ has been chosen for this task.

One key point in the effectiveness of a scheme based on confidence values calculated from distribution models is in the ease of creating and modifying the models. The amount of data that is obtained from each distribution is quite limited, and in a real situation may vary greatly between distributions due to the fact that some classes occur more frequently than others. The methods should therefore be capable of producing reliable estimates even with small amounts of data. Kernel-based distribution estimates fulfill these requirements and two such schemes have been experimented with and are explained below.

Let us first shorten the notation by using $z \equiv q_c^k(x)$. The confidence obtained from the distribution i then stands as $p^i(q_c^k(x)) = p_c^i(z)$. The distribution model i contains N_i previously collected values $z_j^i, j = 1, \ldots, N_i$. The weight assigned to each sample of the critic is denoted with $w_i(z_j^i)$. The distribution index i runs over the distributions for each class c in each member classifier k.

Triangular kernel distribution model estimate: This distribution estimate uses a triangular kernel function,

$$p_{\text{Tri}}^i(z) = \frac{1}{\sum_{j=1}^{N_i} w_i(z_j^i)} \sum_{j=1}^{N_i} w_i(z_j^i) \max\{0, (b - |z - z_j^i|)\} \tag{2}$$

defined by the kernel bandwidth b, which is given as a parameter.

Gaussian kernel distribution model estimate: The distribution is estimated with a Gaussian function as the kernel. The kernel bandwidth b is used as the variance for the Gaussian.

$$p^i_{\text{Gauss}}(z) = \frac{1}{\sum_{j=1}^{N_i} w_i(z^i_j)} \sum_{j=1}^{N_i} w_i(z^i_j) e^{-\frac{(z-z^i_j)^2}{2b}}. \tag{3}$$

Both distribution models are initialized so that while no data points have been collected, the confidence is evaluated as if one value of zero distance had been obtained. The kernel bandwidth b is optimized from the training data.

2.4 Adapting the Critics

It is assumed that information on the correctness of earlier decisions can be obtained and is available for adapting the critics. The first phase of adaptation is the modification of the distribution models. The $q^k_c(x)$ values received from the member classifiers are incorporated into the corresponding critic's distribution model if the classifier was correct. In practice this is done by appending the new $q^k_c(x)$ value to the list of values for that distribution model.

Additionally, a weight is assigned with each distance value stored in the critic's distribution model to facilitate emphasizing newer inputs. For the second phase of adaptation, the weights will be modified to obtain more robust behavior. Three approaches to adjusting the weights of the sample points have been experimented with, and a constant weighting scheme is used for reference.

Constant weights: The weight for each sample is constant, $w_i(z^i_j) = 1$ for all z^i_j. Weighting the samples equally is used as a reference to examine the benefits of the proposed true weighting schemes.

Class-independent weights: The weights are initially set in an increasing order by using an increasing counter (sample index) $n(z^i_j)$ scaled with a suitable constant. If known beforehand, the total number of test samples N can be used for the scaling factor to obtain the weights

$$w_i(z^i_j) = \frac{n(z^i_j)}{N}. \tag{4}$$

These weights do not depend on the distribution model the sample is inserted into, so within each distribution there can be large differences in the weights.

Class-dependent weights: For each distribution, the weights are scaled linearly every time a new sample is inserted. As a result, each sample has weight equal to the ratio of its index $n_i(z^j_i)$ in that particular distribution model and the total number of samples in that model, N_i,

$$w_i(z^i_j) = \frac{n_i(z^i_j)}{N_i}. \tag{5}$$

This results in the first sample having the smallest weight of $1/N_i$ and the most recent sample having the weight $N_i/N_i = 1$.

Decaying weights: When a new sample is inserted to the distribution model, the weights are recalculated to decrease in accordance with a decay constant λ so that

$$w_i(z_i^j) = \max\{0, 1 - \lambda(N_i - n_i(z_j^i))\}. \tag{6}$$

Effectively the inverse of the decay constant λ states how many previous samples the distribution "remembers" at any given time point, with the newest samples being given the most weight.

2.5 Final Confidence and Decision Mechanisms

As the committee now has label information from the member classifiers and the corresponding confidence values from the critics to work with, a scheme is needed for combining them into the final result. The output of the committee is the label of the class that it decides the input most likely belongs to. This label should be deduced from the available information in an optimal manner.

The overall confidence $u_c^k(x)$ given by the critic k for the input x belonging to class c is obtained from the corresponding distribution model by weighting the confidence estimate with a running evaluation of the classifiers' overall correctness rate. This rate p(classifier k correct) is obtained by tracking how many times classifier k has been correct so far and dividing that by the total number of samples classified. Hence the overall confidence is

$$u_c^k(x) = p_c^k(q_c^k(x)) \cdot p(\text{classifier } k \text{ correct}). \tag{7}$$

For the input sample x the decision schemes take the confidences in each label $u_c^k(x)$ from the critics and attempt to form the best possible decision. As the decision mechanisms, especially in the beginning, do not have very much information to work on, a default rule is needed. The default rule here is to use the result of the classifier ranked to be the best on the member validation database. This default rule is applied if no critic suggests a result or several results have exactly the same confidence value.

The effectiveness of the decision scheme is naturally a very important factor in the overall performance of a classifier combination method. It has been often found that in a setting where confidences for all labels can be obtained and the most likely one should be chosen, four basic ways of combining confidences, the *sum*, *product*, *min* and *max* rules, can be very effective in spite of their simplicity [5]. Also in this work these decision schemes are used.

Product rule: For each label, the confidences of the critics are multiplied together, and then the label with the greatest total confidence is chosen,

$$c(x) = \arg\max_{j=1}^{C} \prod_{k=1}^{K} u_j^k(x). \tag{8}$$

Sum rule: For each label, the confidences of the critics are summed together, and then the label with the largest resulting confidence is selected,

$$c(x) = \arg\max_{j=1}^{C} \sum_{k=1}^{K} u_j^k(x). \tag{9}$$

Min rule: For each label, the smallest confidence from a critic is discovered, and then the label with the largest minimum confidence is chosen,

$$c(x) = \arg \max_{j=1}^{C} \min_{k=1}^{K} u_j^k(x). \tag{10}$$

Max rule: For each label, the largest confidence from a critic is discovered, and then the label with the largest maximum confidence is selected,

$$c(x) = \arg \max_{j=1}^{C} \max_{k=1}^{K} u_j^k(x). \tag{11}$$

2.6 An Example of the Committee's Operation

In order to illustrate the operation of the combination scheme, let us review an example. In this example we shall use the triangular kernel distribution model estimate and the *sum* rule for determining the final output, and modify the weights according to the *decaying* weights scheme. Now let there be K classifiers and C classes.

For the input sample x each classifier k outputs C values $d_c^k(x)$, with each value corresponding to the distance to the nearest prototype of class c in classifier k. Then this batch of distances is normalized as in equation (1) to obtain again C values $q_c^k(x)$ for each classifier $k = 1, \ldots, K$.

Now the normalized distances are examined by the respective critics, who calculate their confidence values from their distribution models of existing data points as in equation (2). This results in a set of C confidence values $p_c^k(q_c^k(x))$ for every classifier. These confidence values are further adjusted in the critic by weighting them with the respective classifier's correct classification rate in accordance with equation (7). The final output is then selected using the *sum* rule of equation (9) from the final confidences $u_c^k(x)$ obtained from the critics.

For updating the distributions it is assumed that information on the correctness of the result can be obtained after the classification. Now for each of the K classifiers and their respective critics, if that particular classifier was correct, the normalized distance for the correct class is stored into that classifier's critic's distribution model. Furthermore, the weight corresponding to each collected normalized distance value in the distribution model is updated in accordance to equation (6). After the distributions of all the classifiers that were correct have been updated, the next input sample can be processed.

3 Experiments

The committee experiments were performed using a total of six different classifiers. The used data was online handwritten characters written one-by-one. The collection and preprocessing is covered in detail in [6]. All letters, upper and lower case, and digits were used in the experiments.

The member classifiers were based on stroke-by-stroke distances between the given character and prototypes. Dynamic Time Warping (DTW) was used to

Table 1. Recognition error rates of the member classifiers

Classifier	Distance measure	Accuracy
1	DTW-PP-MC	79.98%
2	DTW-PL-MC	79.22%
3	DTW-PP-BBC	78.82%
4	DTW-PL-BBC	77.72%
5	DTW-NPP-MC	79.17%
6	DTW-NPP-BBC	77.68%

compute one of three distances, the point-to-point (PP), the normalized point-to-point (NPP), or point-to-line (PL) [6]. The PP distance simply uses the squared Euclidean distance between two data points as the cost function. In the NPP distance the distances are normalized by the number of matchings performed. In the PL distance the points of a stroke are matched to lines interpolated between the successive points of the opposite stroke.

All character samples were scaled so that the length of the longer side of their bounding box was normalized and the aspect ratio kept unchanged. The centers of the characters were moved to the origin. For this we used two different approaches: the center of a character was defined either by its 'Mass Center' (MC) or by its 'Bounding Box Center' (BBC) [6].

The data formed three independent databases consisting of different writers. Database 1 consists of 9961 characters from 22 different writers, which were written without any visual feedback. The pressure level thresholding of the measured data into pen up and pen down movements was set afterwards individually for each writer. The *a priori* probabilities of the classes were somewhat similar to that of the Finnish language. Databases 2 and 3 were collected with a program that showed the pen trace on the screen and recognized the characters online. They both contain data from eight different writers and a total of 8077 and 8047 characters, respectively. The minimum writing pressure for detecting and displaying pen down movements was the same for all writers. The distribution of the character classes was approximately even.

Database 1 was used for forming the initial user-independent prototype set for the DTW-based member classifiers. The prototype set for the DTW-based classifiers consisted of seven prototypes per class. Database 2 was used for evaluating the values for the necessary numeric parameters for the committee and determining the performance rankings of the classifiers. Database 3 was used as a test set. The configurations and corresponding error rates of the member classifiers are shown in Table 1. For experiments with the triangular kernel, the decay parameter was set to $\lambda = 0.26$ for the *product*, *sum* and *max* rules and $\lambda = 0.08$ for the *min* rule. Similarly, for the Gaussian kernel the decay parameter was set to $\lambda = 0.27$ and $\lambda = 0.10$, respectively. The kernel bandwidth for both kernel types was in all cases set to $b = 0.4$.

The data is ordered so that all samples from one writer are processed before moving on to the next writer. The adaptive critics are not reinitialized in between writers, so the committee works in a writer-independent fashion.

4 Results

This preliminary set of experiments examines the two applied distribution types used in conjunction with the four weighting schemes presented. The results obtained with all the four combination rules are shown in Table 2. As can be seen, the triangular kernel performs slightly better than the Gaussian kernel for all the combinations. The results using the constant weighting scheme of both the triangular and Gaussian kernel functions seem somewhat disappointing, as they are outperformed by the respective member classifiers. This was however to be expected, as the distribution models strive to model all input data, and with several different subjects providing the data, the distribution estimate collected from the previous writers may well be suboptimal for the new writer.

Robustness with respect to the changing environment is clearly much better obtained by including the use of the weighting schemes. With the *product* and *sum* rules, the accuracies obtained with the most effective *decaying* scheme clearly outperform those of the member classifiers. The *class-independent* and *class-dependent* schemes perform on roughly the same level. In all cases they provide improvement over the situation where no weighting scheme is used, but less than the *decaying* weights scheme. The *decaying* approach is clearly the most effective one, suggesting that using only a subset consisting of $1/\lambda$ most recent samples is more effective in modeling the classifier's performance than using all available data for the distribution estimates. In these experiments $1/\lambda$ corresponded to between five and ten most recent samples.

It can also be noted that the two most effective combination methods are the *product* and *sum* rules. This may be due to the fact that the confidences, while not always satisfying the properties of being a valid probability, share many of the characteristics of probability values and are modeled in a similar fashion. Although the classifiers are hardly independent, the *product* rule seems to be effective. Furthermore the *product* and *sum* rules take the results from all the classifiers and their confidences from the critics into account when making the decision. The *min* rule can also be seen as trusting the most doubtful critic, which, although risk minimization in a way, clearly is not the optimal scheme. Also the *max* rule trusts a single critic, the most confident one, for the decision making process. In its greediness this appears to be the least beneficial strategy.

5 Conclusions

This paper has presented a scheme for calculating adaptive confidence values from member classifiers' distance values. A distribution model that estimates the member classifiers' performance based not only on their performance for the sample being processed, but also learning from prior samples was suggested. In

Table 2. Experiment results with different weighting schemes and decision methods

Distribution model	Weight scheme	Product rule	Sum rule	Min rule	Max rule
Triangular kernel	Constant	75.07%	74.36%	73.89%	72.28%
Triangular kernel	Class-independent	76.07%	75.22%	74.70%	72.40%
Triangular kernel	Class-dependent	76.06%	75.85%	74.65%	72.98%
Triangular kernel	Decay	81.48%	80.78%	78.65%	73.48%
Gaussian kernel	Constant	70.61%	69.91%	71.88%	71.69%
Gaussian kernel	Class-independent	71.13%	70.21%	72.01%	71.34%
Gaussian kernel	Class-dependent	71.50%	70.86%	71.90%	72.06%
Gaussian kernel	Decay	80.60%	79.92%	78.38%	73.31%

the presented approach a weighting scheme forcing the distributions to emphasize most recent samples is used to enhance robustness. This two-staged adaptive confidence evaluation scheme should provide an effective balance between learning from prior samples and being robust with respect to changes in the member classifiers' performances.

Some preliminary results were presented using two types of kernel functions for constructing the distance distribution models and three weighting schemes to provide robustness. These were applied in an experiment where handwritten character data was recognized. The results clearly showed that the applied scheme can improve upon the results of the member classifiers and that the weighting greatly enhances performance. Especially the decaying weights scheme, where the weight of the samples in the distribution model decays linearly to zero, was found to be effective. This suggests that estimating the confidence of a classifier based on a subset consisting of the newest prior results may be a very effective strategy.

References

1. Kuncheva, L., Whittaker, C., Shipp, C., Duin, R.: Is independence good for combining classifiers. In: Proceedings of the 15th ICPR. Volume 2. (2000) 168–171
2. Miller, D., Yan, L.: Critic-driven ensemble classification. IEEE Transactions on Signal Processing **47** (1999) 2833–2844
3. Hongwei Hao, Cheng-Lin Liu, H.S.: Confidence 'evaluation for combining classifiers. In: Proceedings of International Conference on Document Analysis and Recognition. (2003) 755–759
4. Aksela, M., Girdziušas, R., Laaksonen, J., Oja, E., Kangas, J.: Methods for adaptive combination of classifiers with application to recognition of handwritten characters. International Journal of Document Analysis and Recognition **6** (2003) 23–41
5. Kittler, J., Hatef, M., Duin, R., Matas, J.: On combining classifiers. IEEE Transactions on Pattern Analysis and Machine Intelligence **20** (1998) 226–239
6. Vuori, V., Laaksonen, J., Oja, E., Kangas, J.: Experiments with adaptation strategies for a prototype-based recognition system of isolated handwritten characters. International Journal of Document Analysis and Recognition **3** (2001) 150–159