A Meta-Classifier Architecture for Real-Time Pattern Classification

Mihai R. Jalobeanu
Computer Science Department*
Technical University of Cluj Napoca
Baritiu 26, Cluj-Napoca, 400 020
Romania
mihaijal@hotmail.com

Abstract **– We consider the problem of using classifier ensembles for real-time pattern recognition and propose a speed-optimized meta-classifier architecture that adapts to time constraints by trading some classification accuracy over speed when required. We show that, even when making such a trade-off, the proposed meta-classifier can perform significantly better than any of the classifiers in the ensemble. A set of promising experimental results is also presented.**

I. INTRODUCTION

The use of multiple classifiers to improve classification performance has been a fairly common research topic in the pattern recognition field for some time now [1] and a large number of combination methods have been proposed. Most of the previous work has been focused on static, trainable or data-dependent meta-classifiers (combiners) that perform an a posteriori fusion of the classification results produced by the individual classifiers [2, 3]. Some methods for a priori dynamic classifier selection have also been proposed. These are usually based on estimating the classifier performance in the neighborhood of the feature vector being classified [3, 4, 5].

Both approaches mentioned above can involve a significant computational effort in any non-trivial pattern classification scenario. In the combiner case, each classifier has to be executed first before the meta-classifier can decide on the final answer. In the dynamic selection case, the relevant neighborhood of the test vector must be computed and the most competent classifier in that vicinity must be determined. This means that neither approach is particularly well suited to be used in time-critical pattern recognition scenarios.

In this paper we propose a speed-optimized meta-classifier that can scale down its computing requirements under time pressure while still producing good classification results. The proposed architecture would be particularly useful in real-time pattern recognition systems like the ones employed in autonomous robots, where taking a reasonable decision fast is, in some cases, more desirable than spending a lot of time attempting to find the best solution.

II. THE PIPELINE MODEL

Most pattern recognition systems serialize the execution of the classification stage behind the feature extraction stage. In other words, classification is a distinct step that starts only after all the features have been extracted. Moreover, when classifier ensembles are used, the meta-classifier is usually serialized behind all classifiers in the ensemble. This means that the time T it takes to classify a pattern – the *reaction time* - is: b

$$max(T_{Fi}) + max(T_{Cj}) \le T \le \sum_i T_{Fi} + \sum_j T_{Cj}, \quad (1)$$

where T_{Fi} is the execution time of the feature extractor F_i and T_{Cj} is the execution time of the classifier C_j. How close T is to one boundary or the other depends on the level of parallelism of the system.

We propose an alternative architecture (Fig. 1) called *classification pipeline*, which attempts to improve the reaction time of the system while maintaining good classification accuracy.

In this model, a set of feature extraction modules FEM, potentially running in parallel, produce a partial feature vector $\boldsymbol{X}$, initially empty. At any moment t, the position i in $\boldsymbol{X}$ is occupied by the value x_i produced by module F_i if the module finished already, or is empty if the module is still executing. Every time a new value x_i becomes available, a subset $\boldsymbol{S}$ of classifiers is dynamically chosen based on the content of $\boldsymbol{X}$ and the current recognition context R. The classifiers in $\boldsymbol{S}$ are then executed in the order of their known performance, until either all of them have been executed or a predefined reaction time limit is reached. The context R identifies an arbitrary, domain-specific partitioning of the feature space.

The meta-classifier combines the set of partial classification results as soon as any results become available and produces a partial result ω_p - essentially an early best guess – together with a confidence level $p(\omega_p)$.

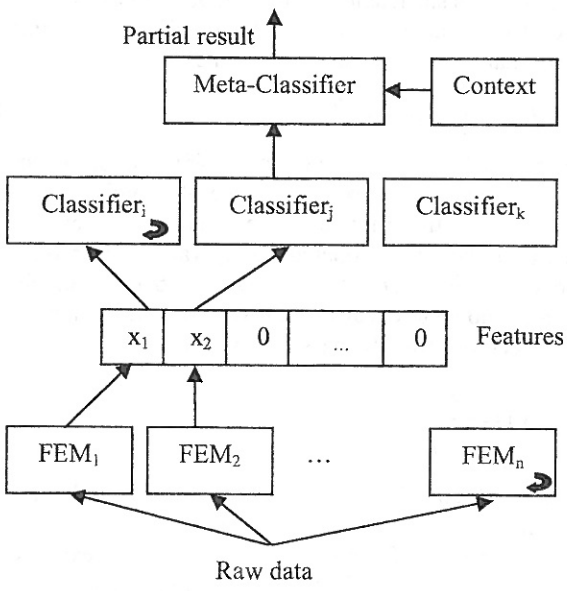

Fig. 1 The pipeline architecture. Results are propagated out even though some of the modules have not yet finished.

The estimate is updated every time another classifier completes its execution. The combination of results can be done by any standard method like voting, Borda count etc.

Using this model, the reaction time T becomes:

$$min(T_{Fi}) + min(T_{Cj}) \le T \le \sum_i T_{Fi} + max(T_{Cj}) . \quad (2)$$

T is closer to the lower bound if the features required by the fastest classifier are extracted first.

Under no time pressure, the model described above eventually produces the same final classification result in the same amount of time as (1). However, (2) shows that the proposed architecture can significantly reduce the reaction time of the system by generating early partial results that might be good enough to support a quick decision. The reaction time can be improved even further by carefully selecting the feature extraction order when designing the system. The possibility of automatically adjusting the priorities of the feature extraction modules based on the performance of the classifiers, so that the features required by the best classifier are extracted first, still needs to be investigated.

III. DYNAMIC SELECTION OF CLASSIFIERS

The dynamic classifier selection step in the model introduced above consists of three parts:

- determining the set S_A of classifiers that can be activated given the current partial feature vector X and recognition context R
- ordering the set based on classifier performance estimates
- executing the classifiers in S_A in order, until either all of them have been executed or the predefined reaction time limit is reached.

Finding S_A means determining if X and R contain all the information needed by the classifier C_i. It requires detailed knowledge about the available classifiers and their dependency on features and context. It can be formalized as an activation function $W_C(X, R)$ that returns 1 if the classifier has sufficient information or 0 otherwise.

Ordering the elements of S_A requires estimating the performance of each classifier, defined as:

$$perf(C) = A_c(R)u(T_c), \quad (3)$$

where $A_C(R)$ is the accuracy of classifier C in the context R, T_c is the average execution time of C and u is *urgency function* of the meta-classifier. The accuracy of the classifier can be estimated using various methods. We propose a function of the total number N_R of feature vectors known to occur in context R and the number N_{CR} of such vectors correctly classified by C before:

$$A_c(R) = \frac{N_{CR}}{N_R}, \quad (4)$$

Both $A_C(R)$ and T_c can be easily learned during the training or validation phase of each classifier C.

The urgency function u is a way to express the importance of classification speed relative to accuracy. Any monotonically descending function such as $1/T_c$ is a good candidate. The meta-classifier could use different urgency functions for different feature vectors or contexts, but must use the same one for the entire recognition process of any given vector X.

IV. EXPERIMENTAL RESULTS

We used the proposed meta-classifier architecture in a word prediction scenario. The pattern to be recognized is a sequence of letters representing the beginning of a word being typed by the user and the result of the classification is the most likely word to start with that sequence.

The training and test corpus consisted of 2000 real-world emails sent by a single user. Out of these, 1800 were used for classifier training and the remaining 200 for testing. The training corpus contained about 8000 different 'words.

We employed three different base classifiers: dictionary, recipient dictionary and bi-gram+. The dictionary classifier uses the word frequency map of all the words extracted from the training corpus. The recipient dictionary classifier uses a similar approach, except it has one such map for every known destination email recipient. The bi-gram+ classifier uses the frequency map of all pairs of vocabulary words known to appear immediately one after another and the frequency map of all the words in the vocabulary that can appear at the beginning of a sentence.

In modeling the word prediction problem as a classification problem, each letter in the sequence to be recognized was interpreted as a feature. The recognition context was defined to contain the recipient of the test email, the previous word in the sentence and the number of letters in the current sequence. Once the base classifiers have been trained, an estimate of each classifier's performance was generated using the training data, for every known recipient, word sequence and substring length. This information was built into the meta-classifier.

Table I shows the results obtained for the test data. The first three rows show the individual accuracy and speed of each base classifier (accuracy was measured as the percent of letters predicted correctly). It is interesting to note that the recipient dictionary classifier is about an order of magnitude more expensive than the other two. The fourth row shows the performance of a simple Borda count meta-classifier that combines the results from all three base classifiers. The next two rows show the performance of the proposed meta-classifier without an urgency function. Both rows exhibit higher average accuracy than any of the base classifiers. The execution time in both cases is significant though, due to the extensive use of the recipient dictionary

TABLE I. Experimental results for a word prediction problem.

Classifier	Success rate	Execution time
Dictionary	29%	35ms
Recipient dictionary	24%	209ms
Bi-gram +	25%	34ms
All 3	34%	399ms
Top 1, $u(T)=1$	32%	85ms
Top 2, $u(T)=1$	35%	241ms
Top 1, $u(T)=T^{1/2}$	**33%**	**38ms**
Top 2, $u(T)=T^{1/2}$	**35%**	**121ms**

classifier.

The last two rows show the results of the proposed meta-classifier using the urgency function $u(T)=T^{1/2}$. Not only is the accuracy still excellent in both cases, but the execution time is now considerably less, because the recipient dictionary classifier is not used so often anymore.

V. CONCLUSIONS

The paper introduced a meta-classifier architecture for time-critical pattern classification systems, designed to generate partial classification results even before all the features have been extracted. The architecture uses a context-based, dynamic classifier selection method to determine the optimal execution order of the base classifiers such that good partial estimates are produced as quickly as possible. Optimality is defined as a function of both accuracy and speed. Experimental results on a word prediction problem have shown that the model has great potential, but future work is needed to validate it in the computer vision and image recognition domain.

VI. REFERENCES

[1] Tin Kam Ho, Jonathan J. Hull, and Sargur N. Srihari, "On multiple classifier systems for pattern recognition", in *Proceedings of the 11th International Conference on Pattern Recognition*, 1992, vol 2, pp. 84–87

[2] A. K. Jain, R. P. W. Duin, J. Mao, "Statistical pattern recognition: A review", *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 22, pp. 4–37.

[3] J. Kittler, "A framework for classifier fusion: is it still needed?", *Advances in Pattern Recognition. Lectures Notes in Computer Science*, vol. 1876, pp. 45-56, F. J. Ferri, J. M. Inesta, A. Amin and P. Pudil, Eds., Springer-Verlag, 2000

[4] G. Giacinto, F. Roli, "A Theoretical Framework for Dynamic Classifier Selection", in *Proceedings of the 15th International Conference on Pattern Recognition, 2000,* pp. 2008-2011

[5] G. Giacinto, F. Roli, "Methods for Dynamic Classifier Selection", in *10th International Conference on Image Analysis and Processing,* 1999, pp. 659-664

[6] A. Tsymbal, S. Puuronen, V. Terziyan, "Arbiter Meta-Learning with Dynamic Selection of Classifiers and its Experimental Investigation", *Advances in Databases and Information Systems, Lecture Notes in Computer Science* , Vol. 1691, pp. 205-217, J. Eder, I. Rozman, and T. Welzer, Eds., Springer-Verlag, 1999

[7] R. O. Duda, P. E. Hart, D. G. Stork, *Pattern Classification,* Wiley-Interscience, 2001.

[8] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules", in *Proceedings of the 20th International Conference on Very Large Databases*, 1994, pp. 487- 499

[9] L. Vuurpijl, L. Schomaker, "Multiple-agent architectures for the classification of handwritten text.", in *Proceedings of the International Workshop on Frontiers in Handwriting Recognition*, 1998, pp. 335-346.