MULTIPLE-AGENT ARCHITECTURES FOR THE CLASSIFICATION OF HANDWRITTEN TEXT

LOUIS VUURPIJL AND LAMBERT SCHOMAKER
Nijmegen Institute for Cognition and Information
P.O. Box 9104, 6500 HE Nijmegen, The Netherlands
vuurpijl@nici.kun.nl, schomaker@nici.kun.nl
http://hwr.nici.kun.nl

Novel pattern recognition techniques using multiple agents for the recognition of handwritten text are proposed in this paper. The concept of intelligent agents and innovative multi-agent architectures for pattern recognition tasks is introduced for combining and elaborating the classification hypotheses of several classifiers. The architecture of a distributed digit-recognition system dispatching recognition tasks to a set of recognizers and combining their results is presented. This concept is being developed in the IART project, where intelligent agent architectures are built for pattern recognition tasks.

1 Introduction

Typical pattern recognition tasks are, e.g., image classification and retrieval, computer vision, motion detection, optical character recognition and handwritten text recognition. In such applications, the goal is to classify an unknown pattern into a number of known categories. In its core form, the pattern represents raw data (i.e., any data acquired through some sensory system) and in the traditional pattern recognition pipeline four stages can be distinguished to establish classification: 1) preprocessing, 2) segmentation, 3) feature extraction and 4) classification. The classification stage is usually implemented in the form of, e.g., a neural network, a statistical classifier, metric classifier, or rule-based classifier. Algorithms available at each of the different stages are called *pattern-recognition modules*. At the NICI, various of such classifiers exist for the application of handwritten (online) text recognition. Many of these classifiers use the same preprocessing and segmentation modules, but differ in the feature extraction and classification phases.

- **preprocessing** A word-delineated online handwriting signal is preprocessed.
- **segmentation** The resulting signal is segmented into velocity-based strokes[1].
- **feature extraction** Depending on the classifier in use, features from the segmented signal are extracted as stroke features, allographs features or image features.

- **classification** Each of the classifiers produces stroke-delineated character hypotheses, which are used for postprocessing.

1.1 Combining multiple experts

The use of multiple classifiers and the combination of their classification results has gained considerable interest in the last few years. This approach has a powerful potential because it may exploit the advantages of different feature representations and classification methods. For example in [2,3,4,5,6,7], some of this work is reported. In the systems described, two architectures can be distinguished:

- Individual classifiers with some combination scheme.
- Multi-stage or hierarchical classifiers.

The first set of classifiers may share the same input features, or use completely different representations of the feature space. Multi-stage classifiers feature a set of relatively simple classifiers in the first stages, which distinguish between a number of coarse classes or output one or more rejects. During the later decision stages a further class distinction is made, where more specialized classification tasks may be performed. Both individual classifiers and multi-stage classifiers require a combination of their output hypotheses. Several combination schemes are possible, like majority vote, max/min/median rule, BKS [5], the Dempster-Shafer rule or Borda count. These use either class labels, rank order or score combinations [8]. Substantial classification improvements are reported of 5% up to 15%.

1.2 Towards multiple agents

The combination of classifier hypotheses in the area of pattern recognition is an example of the more general and fundamental problem of integration of information from multiple sources. This problem, formerly only known as the bottom-up vs top-down processing problem, takes place at all stages of information processing in pattern recognition. Consider, for example, two character classifiers in an optical character recognition system, one for the digits, and the other for the letters in the alphabet. If the letter classifier claims that the letter 'o' is the best matching pattern, whereas the digit classifier claims that the digit '0' is the best match, then more information is needed from an independent source. This source, however, may reside at several possible abstraction levels. The disambiguation may come from the signal (the image elements and their detailed shape) itself, from higher-level current expectancies on possible character classes, or from sibling classifiers at the same processing level. We propose to make use of a new and promising paradigm, i.e., the use of so-called 'multiple agents' [9]. The idea of multiple independent agents which are in a negotiation process is not new (it has its roots in Selfridge's Pandemonium [10] (SP) of specialized daemons in a processing task). A discussion of the SP is given in Bates and Elman [11]. The multiple-agent paradigm has evolved considerably since the SP and nowadays is an important research topic. A large number of definitions of multiple agents are discussed by Franklin and Graesser [12]. In their opinion, an autonomous agents is "a system situated within and part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to effect what it senses in the future". A number of other agent definitions are, e.g., The Maes Agent [13], The KidSim Agent [14], The Hayes-Roth Agent [15], or The Wooldridge-Jennings Agent [16]. The multiple-agent paradigm offers a number of new and intriguing characteristics:

- A number of negotiation methods have been developed, which are borrowed from Game theory or from the research in economics. Many of these algorithms have been formally proven to converge [17].

- The research area of multi-sensor fusion has produced a number of algorithms for the integration of quantitative information from multiple sources [18].

- The formal definition of an intelligent agent following Wooldridge and Jennings [16], allows for the use of algorithms based on belief systems, and also allows for the use of traditional AI inference methods

- New learning algorithms are currently developed for individual and co-operative learning in multiple-agent systems [19,20,21], including genetic algorithms and genetic programming [22,23,24]

This approach allows for alternative ways of combining classifier outputs, where agents discuss the different outcomes and produce an *educated guess* based on their combined reasoning.

2 A multiple-agent architecture for pattern recognition tasks

In Figure 1, an architecture is proposed using multiple agents combining the recognition hypotheses of several classifiers.

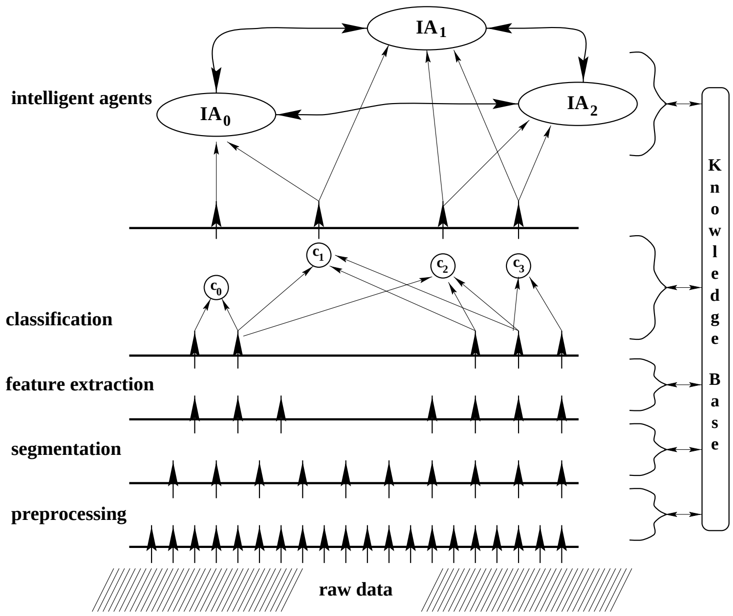


Figure 1: Proposed architecture of a pattern recognition system using multiple intelligent agents.

Here, we consider one or more *feature extractors*, each using information provided via one or more *preprocessors* and *segmentors*. Each out of a set of *classifiers* may use one or more sets of extracted features, resulting in a number of guesses as to what class a pattern may belong. A number of intelligent agents each uses one or more of these hypotheses to reason about them and come up with an educated guess. A significant property of this system is that agents can discuss their outcomes and decision backgrounds with each other. Their knowledge may origin from different levels in the pattern recognition pipeline, depicted in Figure 1 as the *knowledge base*. Via this concept, agents may also re-parameterize pattern recognition modules.

3 The knowledge base

In the pattern recognition architecture we envisage, the knowledge base (KB) plays an important role. It contains information about the pattern recognition modules that are available, and it contains knowledge about which modules are suited for a specific pattern recognition task. The concept of agents discussing information and requesting services from pattern recognition modules requires *domain knowledge*. For example, a number of classifiers claim to recognize a $<a>$ and another set of classifiers recognizes a $<u>$. Based on this situation, an agent claiming to be a $<a>$-expert may become active and require some features (d, ω, p, r) (see Figure 2).

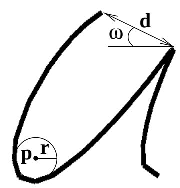


Figure 2: Features (d, ω) describe closure, (p, r) describe roundness of a $<a>$ or $<u>$. Note that these features are specific for a discrimination between these letters, while the same features constitute a source of noise in the discrimination of other letter classes.

Via the KB, a preprocessed and segmented input signal may be examined by different feature extractors producing the required features. Based on these, the $<a>$-expert can decide whether it is an $<a>$ or not. Subsequently, it has to negotiate with the other agents and try to convince them about the input being a $<a>$ or not. Note that in this approach agents become active when required, involving that features are computed and classifiers become involved only on demand. This reduces the curse of dimensionality (having so many features of which only a subset is relevant), while not throwing away features in an early stage. It also reduces the amount of computation because as opposed to a multi-expert system, not all classifier outputs have to be known.

The idea of agents exchanging educated guesses requires an infrastructure and some language via which they can communicate. Currently, several standards of inter-agent communication exist like KIF [25] (knowledge interchange format) and KQML [26] (knowledge query and manipulation language). These are known as negotiation languages, via which agents can discuss their opinions like:

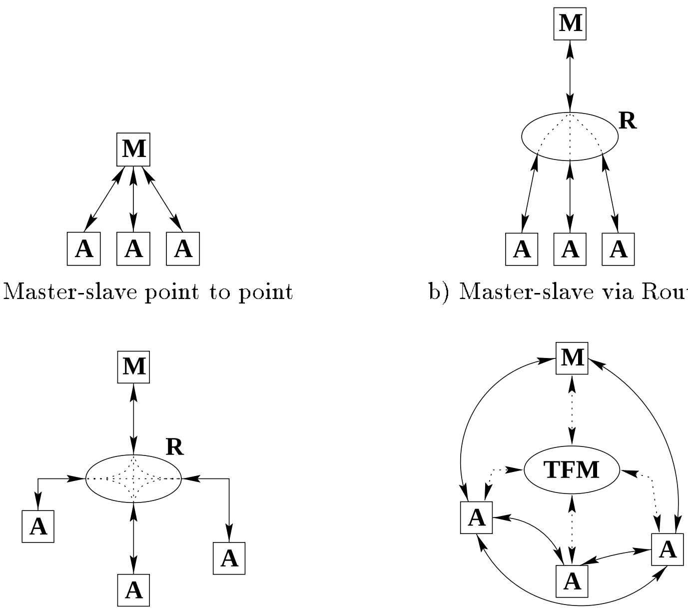
> "I believe that the signal from sample 0 to 156 represents an $<a>$ with a certainty of 0.94.", or "Yes, but consider the (d, ω) features, which indicate it should definitely be a $<a>$.", or "The allograph contains a central crossing, so it cannot be a $<u>$."

In agent theory, each agent can have its own *universe of discourse*. The sample inter-agent discussions above indicate that agents share the same universe of discourse or at least are aware of each others universes. The concepts of KIF and KQML allow for this premise, but require a concise formal specification of each agent's objects, functions and relations, all comprising the knowledge base. In our context, the universe of discourse contains objects, like a raw handwriting signal, a preprocessed signal or its extracted features. Furthermore, it contains functions on these objects, which are for example the implemented feature extraction or classification algorithms. And it contains annotated objects, annotated with extra information like character or word hypotheses.

The following intuition describes why the approach of agent negotiation may be useful in pattern recognition. A recurring question is, whether the combination of classifier outputs should not be interpreted as a new type of classification problem, searching for class centroids (good clusters of experts for a particular knowledge subdomain or character class) and class-separation boundaries between clusters of experts. However, a characteristic of the intermediate-level representations in a multi-level classification scheme is that they are largely different from the signals recorded at the sensory level. Rather than interval or ratio-scale real numbers, the intermediate outputs of classifiers consist of nominal and ordinal propositions, with a optional likelihood or quality value attached to each proposition. It is unlikely that simple linear or mildly non-linear separatrices can easily be found for the meta-classifier which combines the multiple-expert opinions. If trainable statistical or neural-network meta-classifiers are used, there will be the problem of cumbersome training at multiple levels in the architecture. In such an approach, there will be the risk of a rigid information processing architecture which cannot easily be adapted to changes in operation conditions and application contexts. The paradigm of a mixed quantitative and symbolic negotiation protocol, on the other hand, allows for both a flexible adaptation to varying conditions, as well as a powerful mechanism to represent highly non-linear metaclass boundaries.

4 IART, A pattern recognition system using intelligent agents

At the moment, we are developing pattern recognition systems using the concepts described above. The first requirement for any agent architecture is the communication infrastructure. In the current implementation, network socket I/O libraries are being used to transfer KQML queries between JATLite[27] agents, and provide four communication architectures between a master process and several agents (see Figure 3). In this setup, the master process acts as data provider. It accepts an online handwriting signal, preprocesses it and segments it into velocity-based strokes. The resulting annotated signal is called S_a. The agent processes are each capable of receiving S_a, extracting features from it, and producing an ordered list of character hypotheses.



a) Master-slave point to point
b) Master-slave via Router
c) Master-slave point to point via Router
d) Full connectivity

Figure 3: Four communication architectures between pattern recognition agents.

The master may be directly coupled to a number of agents (Figure 3a), agents and the master may be connected via a router (Figure 3b), agents may have virtual point-to-point connections via a router (Figure 3c), or all agents and the master may be fully connected (Figure 3d). In the latter case, a router acts as *task force manager*, administrating which agents are alive and where they are located on the Internet. Each of these communication architectures uses the TCP/IP socket libraries, which involves that all tasks can be performed in parallel on different machines. Interfaces are developed to the JATLite system, which supports the KQML negotiation standard. The system we implemented is called IART, an acronym for Intelligent Agent system for pattern Recognition Tasks.

5 A digit recognition system using IART

As a first proof of concept, we implemented a digit recognition system using IART. A description of this system is given in [9]. The data that is used for this experiment stems from the NIST UNIPEN data set (Release 7). From this dataset, 15557 digits are extracted, of which a randomly selected number of 7998 digits are used for training, the other 7779 for testing. The system contains one metric classifier *hclus* [28], which matches an unknown input to a number of prototype templates found via hierarchical clustering of the training set. Recognition performance of *hclus* is 95.9%. Using the training set, a confusion profile is generated. For example, *hclus* has 30 cases of 1-7 confusion and 27 cases of 7-2 confusion. For this system, a number of agents are designed, where each agent has the task of solving a particular conflict. Potential conflicts are detected if the distance between the input and the best matching prototype is above a certain threshold. Only if a conflict occurs, the corresponding conflict solver is engaged. With this application we have set a first approach to using multiple intelligent agents for pattern recognition tasks. The architecture described in Section 4 is proven to work well. Advantages are:

1. **Speedup**, as each agent and classifier (process) may be running in parallel on different workstations in the Internet.
2. **Encapsulation**, as each process is a stand-alone program with its own specific implementation. This approach also facilitates fast development of new classifiers.
3. **Problem-driven** approach, where specific features and algorithms are used on demand. Note that this also provides a compartmentalization of feature space.

The disadvantage is that each conflict solver must be designed manually.

6 IART: Requirements and future directions

In the IART project, we are building and testing the infrastructure for the intelligent agents described above. Other applications are being constructed, in particular recognition of unconstrained handwritten text, and searching in image databases on the Internet. In the architecture described above, the intelligent agent concept is not fully exploited yet. Intelligent agents as defined in Sections 1.2 and 2 are able to discuss results with each other using a negotiation language. As mentioned above, the agent approach provides encapsulation: agents do not have to know *how* other agents perform their tasks. What is required is *what* services are provided by agents and a specification of the language that agents speak: the inter-agent protocol. This is comprised in the KB, via which agents learn about how to interact with the agents from whom they require services or with whom they negotiate about classifier outputs. In IART, the intelligent agent concept adheres to the following design criteria, some of which are described in [29].

1. **Data objects.** Data can be the raw input signal, preprocessed or segmented signal S_a.

2. **Data properties.** Depending on the application, a S_a can have statistical properties (e.g., color histogram in case of images), structural properties (e.g., *contains crossing at* (x, y), *has 4 strokes* in case of handwritten text). These properties are absolute; they are computed by the preprocessor, segmentation and feature extractor pattern recognition modules depicted in Figure 1.

3. **Belief properties.** Agents can express hypotheses about a S_a as a set of tuples $(segm, prop, reason)$. The segmentation information *segm* specifies the part of the signal of which properties are given. The properties *prop* can be, e.g. *is an* $<a>$ *with certainty 0.9* and are always accompanied by a belief factor, the certainty. The *reason* specifies the sources based on which the belief is founded.

4. **Internal agent architecture.** Each agent receives one of the IART data objects, possibly annotated with data and belief properties, followed by a number of requests regarding the data. Following the idea of encapsulation, no implementation restrictions are made, as long as agents adhere to the interaction and protocol specifications listed below.

5. **Control and interaction.** Agents are started via a script. Upon initiation, each agent registers itself to a central task force manager (TFM, see Figure 3d). Via the TFM, other agents can find out whether an agent is available or not, and which services it provides. Agents await requests from other agents in the task force and respond according to the associated protocol.

6. **Protocol and interaction style.** Agents interact via basic information exchange of IART data and properties, or they interact via a negotiation scenario. The first level of interaction follows a client-server protocol and assumes that agents know via the KB which other agents can aid in, e.g., segmentation, feature extraction or other data processing operations. The digit recognizer application discussed above uses this type of basic information exchange: the digit experts are requested to consider a piece of handwriting and decide whether it could belong to a specific class. For the second level of interaction, several specifications are being developed, both in the context of handwriting recognition and information retrieval [30].

We are currently implementing IART based on these design principles. Running prototypes are being tested and results of the performance effects are are being analyzed.

7 Conclusions

A new, promising architecture for pattern recognition applications is proposed in this paper. The IART system provides a framework for using intelligent agents. As a first proof of concept, we implemented a digit-recognition system using IART. The use of intelligent agents offers interesting dynamic properties, because problem-related features and algorithms are engaged on demand, and because decision boundaries are created on a problem-driven basis. As such, the increased use of multiple agents in pattern recognition represents a shift from geometric matching towards algorithmic search. The intrinsic modularity of using software agents provides a means of fast, standardized development and parallelism. Furthermore, it offers a compartmentalization of feature space, as each agent only has to compute the features it requires. Future research will be directed at learning capabilities of our system. Whereas the detection of classification conflicts is performed automatically and can be trained, building agents solving the conflicts is still done by hand.
The use of multiple agents in the context presented here allows for innovative combinations of classifier outputs, which are orthogonal and complimentary to existing approaches. This requires the use of negotiation methods which are currently under investigation to be incorporated in IART.

References

1. L. Schomaker and H-L. Teulings. A handwriting recognition system based on the properties and architectures of the human motor system. In *Proceedings of the International Workshop on Frontiers in Handwriting Recognition (IWFHR)*, pages 195–211, Montreal: CENPARMI Concordia, 1990.
2. R. Lindwurm, T. Breuer, and K. Kreuzer. Multi expert system for handprint recognition. In S. Impedovo and A. Downton, editors, *Progress in Handwriting Recognition*, pages 293–298. IAPR, World Scientific Publishing, September 1996.
3. A.F.R. Rahman and M.C. Fairhurst. A new approach to handwritten character recognition using multiple experts. In *Progress in Handwriting Recognition*, pages 321–325, September 1996.
4. J. Kittler. Improving recognition rates by classifier combination: a theoretical framework. In S. Impedovo and A. Downton, editors, *Progress in Handwriting Recognition*, pages 231–247. World Scientific Publishing, September 1996.

5. Y.S. Huang and C.Y. Suen. An optimal method of combining multiple classifiers for unconstrained handwritten numeral recognition. In *IWFHR III, Frontiers in handwriting recognition*, pages 11–19, May 1993.
6. Y.S. Huang and C.Y. Suen. A method of combining multiple experts. *Trans. pattern analysis and machine intelligence*, 17(1):90–94, Jan 1995.
7. T. Tsutsumida, F. Kawmata, S. Ymaguchi, K. Nagat, and T. Wakahara. The third iptp character recognition competition and study on multi-expert systems for handwritten Kanji recognition. In *Progress in Handwriting Recognition*, pages 299–304, September 1996.
8. S. Madhvanath and S. Srihari. Modes and measures: A framework for design of parallel word classifier combination schemes. In S. Impedovo and A. Downton, editors, *Progress in Handwriting Recognition*, pages 309–313. World Scientific Publishing, September 1996.
9. L. Vuurpijl and L. Schomaker. A framework for using multiple classifiers in a multiple-agent architecture. In *Third international workshop on Handwriting Analysis and Recognition*, pages 8/1–8/6, July 1998.
10. O.G. Selfridge. Pandemonium: a paradigm for learning in mechanisation of thought processes. In *Proceedings of a Symposium Held at the National Physical Laboratory*, pages 513–526, London, November 1958 1958. HMSO.
11. E.A. Bates and J.L. Elman. Connectionism and the study of change. In M. Johnson, editor, *Brain Development and Cognition: A Reader*, pages 623–642. Blackwell Publishers, Oxford, 1993.
12. S. Graesser, A.and Franklin. Is it an agent,or just a program? a taxonomy of internet agents. In *Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages*. Springer-Verlag, 1996.
13. P. Maes. Artificial life meets entertainment: Life-like autonomous agents. *Communications of the ACM*, 108-114.
14. D.C. Smith, A. Cypher, and J. Spohrer. Kidsim: Programming agents without a programming language. *Communications of the ACM*, 37(7):54–67, July 1994.
15. B. Hayes-Roth. An architecture for adaptive intelligent systems. *Artificial Intelligence: Special Issue on Agents and Interactivity*, 329-365.
16. M. Wooldridge and N.R. Jennings. Agent theories, architectures, and languages: a survey. In Wooldridge and Jennings, editors, *Intelligent Agents*, pages 1–22. Springer-Verlag, Berlin, 1995.
17. A. Rubenstein. A perfect equilibrium in a bargaining model. *Econometrica*, 50:97–109, 1982.
18. R.R. Brooks and S. Sitharama Iyengar. Robust distributed computing and sensing algorithm. *IEEE Computer*, 29(6):53–60, June 1996.
19. Albers, Wulf, and J.D. Laing. Prominence, competition, learning, and the generation of offers in computer-aided experimental spatial games. In R. Selten, editor, *Game Equilibrium Models*, volume III: Strategic Bargaining, pages 141–185. Springer Verlag, 1991.
20. H. Matsubara, I. Noda, and H. Hiraki. Learning of cooperative actions in

multiagent systems: A case study of pass play in soccer. In *Adaptation, Co-evolution and Learning in Multiagent Systems: Papers from the 1996 AAAI Spring Symposium*, page 63. AAAI Press, 1996. Technical Report SS-96-01r, ISBN 0-929280-99-7.

21. J. Schmidhuber. A general method for multi-agent reinforcement learning in unrestricted environments. In *Adaptation, Coevolution and Learning in Multi-agent Systems: Papers from the 1996 AAAI Spring Symposium*, page 86. AAAI Press, 1996. Technical Report SS-96-01, ISBN 0-929280-99-7.
22. D.E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, Reading, MA, 1989.
23. J.R. Koza. *Genetic Programming*, volume II. MIT Press, Cambridge, MA, 1994.
24. G.O. Dworman, S.O. Kimbrough, and J.D. Laing. On automated discovery of models using genetic programming: Bargaining in a three agent coalition game. *Journal of Management Information Systems*, 1995.
25. M.R. Genesereth and R.E. Fikes. Knowledge interchange format. version 3.0 reference manual. Technical Report Loci-92-1, Computer Science Department, Stanford University, 1992.
26. T. Finin, J. Weber, and *et al* Widerhold, G. Specification of the kqml agent-communication language. Technical Report EIT TR 92-04, Enterprise Integration Technologies, Palo Alto, CA, 1992.
27. H.R. Frost and M.R. Cutkosky. Design for manufacturability via agent interaction. In *Proceedings of the 1996 ASME Computers in Engineering Conference*, pages 1–8, Irvine, CA, August 1996.
28. L. Vuurpijl and L. Schomaker. Finding structure in diversity: A hierarchical clustering method for the categorization of allographs in handwriting. In *ICDAR*, pages 387–393. IEEE, August 1997.
29. F. Farhoodi and I. Graham. A practical approach to designing and building intelligent software agents. In *PAAM '96, the practical application of intelligent agents and multi-agent technology*, pages 181–204, April 1996.
30. L. Schomaker. Towards an intelligent-agent architecture: Profile negotiation language. Technical report, NICI, Institute of Cognition and Information, 1998. in progress.