Dynamic Integration of Virtual Predictors

Vagan Terziyan

**Department of Computer Science and Information Systems, University of Jyvaskyla, P.O.Box35, FIN-40351 Jyvaskyla, Finland, fax: +358142603011, e-mail:* vagan@jytko.jyu.fi

Abstract

Knowledge discovery is the process of finding previously unknown and potentially interesting patterns, relations and matches in large and possibly distributed databases. Numerous classification techniques, feature selection and distance evaluation methods have recently been developed and for each case it is necessary to evaluate and then select the most appropriate method among them. We base on the assumption that each classifier, each feature selection method, and each distance evaluation function are the best inside some sub domains of the application domain. However if we will separately select each of them for certain instance then it does not mean that together they provide the best classification result. That is why consider a team completed from available classifiers, feature selection and distance evaluation methods as a unit to be learned to classify instances. In this paper, we develop a dynamic integration technique, which estimates competence areas for such teams and benefit from this when solves classification problems. A team together with learning method used form a *virtual classifier*, which can possibly include in its team such combinations of methods that have not yet been considered or even can be inconsistent but nevertheless can be used to improve the classification result. We apply this technique to mobile e-commerce location-based services, where classification means assignment of appropriate service provider to a mobile customer based on his location and profile; feature selection is treated as profile data filtering; and distance evaluation as profile matching.

1 Introduction

Knowledge discovery is the process of finding previously unknown and potentially interesting patterns and relations in large databases [Fayyad *et al.*, 1997]. Numerous data mining methods have recently been developed to extract knowledge from databases. In many cases it is necessary to evaluate and then select the most appropriate data-mining method or a group of methods. Often the method selection is done statically without analyzing each new instance. When the method selection is done dynamically taking into account the characteristics of each instance better results are achieved usually.

In a variety of applications, researchers combining efforts to learn, create and combine an *ensemble of classifiers*. For example, in [Dietterich, 1997] integrating multiple classifiers has been shown to be one of the four most important directions in machine learning research. The main discovery is that ensembles are often much more accurate than the individual classifiers. In [Skalak, 1997] the two advantages of combining classifiers were shown: (i) the possibility that by combining a set of classifiers, we may be able to perform classification better than with any classifier alone, and (ii) the accuracy of a sophisticated classifier may be increased by combining its predictions with those made by an unsophisticated classifier.

Feature selection is an important focus of interest for good machine learning techniques. Feature selection methods try to pick a subset of features that are relevant to the target concept. Each of these methods has its strengths and weaknesses based on data types and domain characteristics. As is well known, there is no single feature selection method that can be applied to all applications. The choice of a feature selection method depends on various data set characteristics: (i) data types, (ii) data size, and (iii) noise. Based on different criteria from these characteristics in [Dash and Liu, 1997] it were given some guidelines to a potential user of feature selection as which method to select for a particular application.

Evaluation of distances or similarity measurements are very important in data mining. Distance between instances is used to recognize nearest neighbors of any classified instance, that is widely used by many classifiers' integration techniques [Skalak, 1997]. Distance between classes is necessary when classifiers learn based on training set to define the misclassification error for every classifier in every training instance (for example Cross-Validation Majority approach [Kohavi, 1995; Merz, 1996]). This distance helps to find out the areas of competence for classifiers [Koppel and Engelson, 1996]. Distance between classifiers is useful for example to evaluate weights of every classifier to be able to integrate classification results by weighted voting [Bauer and Kohavi, 1998]. There are many approaches to define distance between any two instances based on their numerical or semantic closeness [Dash and Liu, 1997]. The choice of a distance evaluation method also depends on various data characteristics as well as for feature selection.

We base on the assumption that each classifier, each feature selection method, and each distance evaluation function are the best inside some sub domains of the application domain. One can develop a dynamic selection (or dynamic integration) technique that can estimate the competence areas separately for classifiers, for feature selection methods, and for distance functions, which allows us to select (or to prefer) the best representatives of these three groups to classify every new instance. Such technique was developed in [Terziyan *at al.*, 1998; Tsymbal *at al.*, 1998] for dynamic integration of classifiers as one variation of the stacked generalization method [Wolpert, 1992; Skalak, 1997].

To be able to follow the above assumption for all tree groups of methods we should also take into account the well-known *team effect*: “Best players are not necessary form the best team”. If we will separately select the best classifier, the best feature selection method, and the best distance function for certain instance then it does not mean that together they provide the best classification result. Thus we should consider a team completed from available classifiers, feature selection and distance evaluation methods as a unit to be learned to classify instances. In this paper, we develop a dynamic integration technique, which tries to estimate competence areas for such teams and benefit from these competence areas to solve classification problems. A team together with learning method used (team direction) form a *virtual classifier*, which can possibly include in its team such combinations of methods that have not yet been considered in literature or even can be inconsistent but nevertheless can be used to improve the classification result.

2 A Virtual Classifier

Virtual Classifier is a seven:

$$<\underbrace{\overbrace{\boldsymbol{TC,TM,TP,TI}}^{\text{Constant Staff}}}_{\textit{Team Direction}},\underbrace{\overbrace{FS,DE,CL}^{\text{Flexible Staff}}}_{\textit{Team}}>,$$

which consists of two groups of cooperative agents:

- the Constant Staff group or *Team Direction* where ***TC*** is a Team Collector, ***TM*** is a Training Manager, ***TP*** is a Team Predictor, ***TI*** is a Team Integrator, and
- the Flexible Staff group or *Team* where *FS* is a Feature Selector, *DE* is a Distance Evaluator, and *CL* is a Classifier.

There are two groups of problems related to a virtual classifier and two appropriate phases of knowledge discovery with a virtual classifier.

The *learning problem* is as follows:

$$\textit{Training Set} \rightarrow \textbf{Virtual Classifier} \rightarrow \textit{Learned Teams},$$

where training set is a set of sample instances with known classification result, learned teams are those to which weights are assigned relatively to every sample instance.

The learning problem should be solved during the *learning phase* with a virtual classifier. The competence of a training manager from a virtual classifier is to perform an appropriate learning algorithm to train different teams of classifiers completed by a team collector.

The *classification problem* is as follows:

$$\textit{New Instance} \rightarrow \textbf{Virtual Classifier} \rightarrow \textit{Classification Result}.$$

The classification problem should be solved during the *application phase* of knowledge discovery with a virtual classifier, which consists of already learned teams. The team predictor evaluates every team relatively to the new instance. The competence of a team integrator is to combine classification results produced by all selected or evaluated teams. We suppose that all virtual classifiers are based on the same basic sets of a team's components: feature selectors, distance evaluators, and classifiers. The difference between virtual classifiers is based only on different selection of their team direction parts.

3 Team Members of a Virtual Classifier

We review feature selection and distance evaluation methods and classifiers and introduce the roles of the appropriate members of a virtual classifier team.

3.1 Feature Selectors

The majority of real-world classification problems require supervised learning where relevant features for each instance are often unknown *a priori*. In many applications with large datasets, learning might not work without removing irrelevant features. Feature selection methods try to pick a subset of features that are relevant to the target concept [Dash and Liu, 1997].

There have been quite a few attempts to study feature selection methods based on some framework or structure. Prominent among these are [Doak, 1992] and [Siedlecki and Sklansky, 1988] surveys. Siedlecki and Sklansky [1988] discussed the evolution of feature selection methods and grouped the methods into past, present, and future categories. In [Dash and Liu, 1997], a survey is conducted for feature selection methods starting from the early 1970's to the most recent methods. The two major steps of feature selection (generation procedure and evaluation function) are divided into different groups, and 32 different feature selection methods are categorized based on the type of generation procedure and evaluation function that is used (see Figure 1).

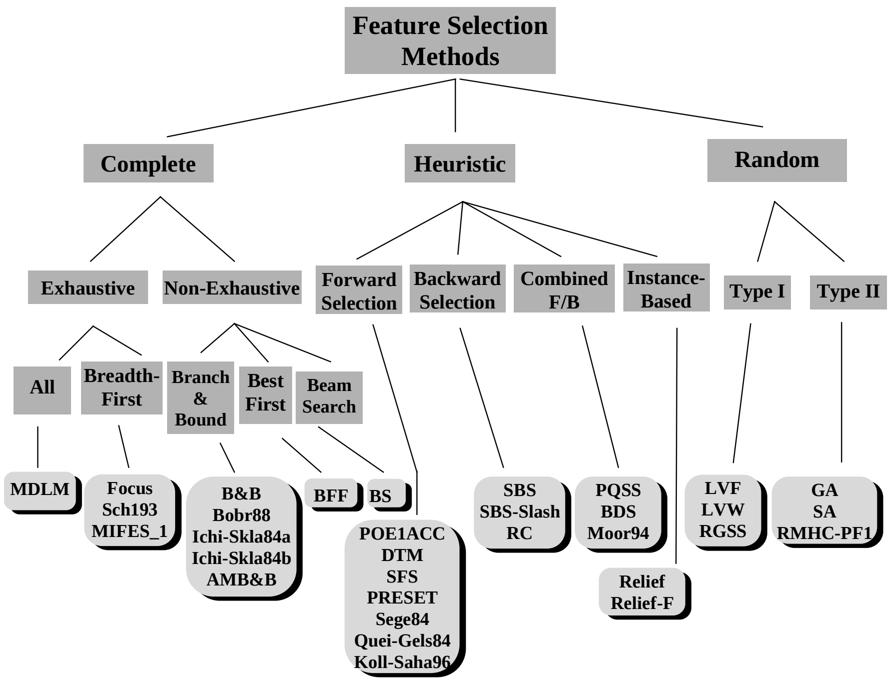

Figure 1: Classification of feature selection methods according to [Dash and Liu, 1997]

Main role of a feature selector: to find the minimally sized feature subset that is sufficient for correct classification of the instance. Difference between feature selectors is based on different generation, evaluation, and validation procedures and stopping criterion used to select features.

The important abilities of feature selectors are to handle when appropriate: irrelevant, correlated, redundant features, multiple classes, large datasets, noisy data.

3.2 Distance Evaluators

There are many approaches to define distance between any two instances based on their numerical or semantic closeness. For example the semantic closeness between terms is a measure of how closely terms are related in the classification schema [Tudhope and Tailor, 1997]. Distance metric used by Rada *et al.* [1989] represents the conceptual distance between concepts. Rada *et al.* [1989] uses only the path length to determine this conceptual distance, with no consideration of node or link characteristics. Distance is measured as the length of the path representing the traversal from the first classification term to the second one. Rocha [1991] has suggested a method to "fuzzify" conversation theory, by calculating continuously varying conceptual distances between nodes in an entailment mesh, on the basis of the number of linked nodes they share. In order to measure the distance between two concepts in a mind, Jorgensen measures a distance between two concepts, which he calls *psy* [Jorgensen, 1997]. It has been suggested to assign an arbitrary distance of *n* units to the separation between two concepts such as "Concept *A*" and "Concept *B*" and then ask a subject to tell us how far other concepts (*C* and *D*) are from each other in these units. Instance-based learning techniques typically handle continuous and linear input values well, but often do not handle nominal input attributes appropriately. Also a probabilistic metric of the PEBLS classification algorithm [Cost and Salzberg, 1993] can be used to compute the similarity between two instances. The distance d_i between two values v_1 and v_2 of certain attribute is:

$$d(v_1, v_2) = \sum_{i=1}^{k} \left(\frac{C_{1i}}{C_1} - \frac{C_{2i}}{C_2} \right)^2,$$

where C_1 and C_2 are the numbers of instances in the training set with values v_1 and v_2, C_{1i} and C_{2i} are the numbers of instances of the *i*-th class with values v_1 and v_2, and *k* is the number of classes. The value difference metric was designed by Wilson and Martinez [1997] to find reasonable distance values between nominal attribute values, but it largely ignores continuous attributes, requiring discretization to map continuous values into nominal values. As it was mentioned in the Wilson and Martinez [1997] review there are many learning systems that depend on a good distance function to be successful. A variety of distance functions are discussed in this review and available for such uses, including the Minkowsky, Mahalanobis, Camberra, Chebychev, Quadratic, Correlation, and Chi-square distance metrics; the Context-Similarity measure; the Contrast Model; hyperrectangle distance functions and others [Wilson and Martinez, 1997].

Main role of a distance evaluator is to measure distance between an instance to be classified and the sample instances based on their numerical or nominal attribute values. This distance evaluation is made in accordance with the requirements and used by the appropriate team classifier in its certain way. Difference between distance evaluators is based on different techniques, formulas, and algorithms to measure distance or evaluate similarity between various attributes.

3.3 Classifiers

In supervised learning a training set:

$$\{(\mathbf{x}_1,y_1), (\mathbf{x}_2,y_2),\ldots, (\mathbf{x}_n,y_n)\}$$

is given for some unknown function $y = f(\mathbf{x})$, where $\mathbf{x}_i$ are vectors of features and y_i values are defined on a discrete set of classes {1,2,…,K}. The classifier is a hypothesis about the true function *f*. Such hypotheses are typically based on different classification techniques. Among these are: *k-NN* classifiers, minimum distance classifiers, decision tree classifiers, linear discriminant classifiers, Bayessian classifiers, minimum error rate classifiers, multi-level slice classifiers, maximum likelihood classifiers, weighted likelihood classifiers, neural networks-based classifiers, genetic classifiers, self-organizing maps-based classifiers, wavelets-based classifiers and others.

Main role of a classifier is to assign class to any new instance based on its selected features and its location relatively to sample instances. Difference between classifiers is based on different formulas or algorithms of processing features' values and distances.

4 Team Direction

We consider one example of a virtual classifier where we fix our requirements to a team direction part: collector, predictor, integrator, and training manager.

4.1 Team Collector

Main role of a team collector is to complete different consistent teams from available feature selectors, distance evaluators, and classifiers to be trained on a sample set of instances. Difference between team collectors is based on different requirements for consistency control and different algorithms to restrict complete consideration of all possible teams. We supposed to use a so-called *nil team collector* which decisions are based on considering all possible teams without any consistency or other restrictions.

4.2 Training Manager

Main role of a training manager is to train all completed teams on sample instances by assigning the weight to every team relatively to every sample instance based on appropriate learning algorithm. Difference between training managers is based on different learning algorithms used. We supposed to use a training manager which learning algorithm is based on leave-one-out cross validation principle (stacking) [Wolpert, 1992; Skalak, 1997.]. It is supposed that any inconsistent team will also receive some weight accordingly to appropriate misclassification error or it will be assigned weight equal to zero if the team cannot classify the instance at all. Training manager apply all completed teams to every training instance and form the performance matrix $\boldsymbol{M}_{nxm}$, where *n* is the number of sample instances in the training set and *m* is the number of teams completed by the team collector. Each value $\boldsymbol{M}_{ij}$ of the performance matrix is the weight obtained on the basis of the value of misclassification error made by *j*-th team in classification of *i*-th sample instance. The classification result for *i*-th sample instance is calculated by every team on the basis of known outcomes of all sample instances except *i*-th one according stacked generalization technique and then compared with known outcome for *i*-th instance.

4.3 Team Predictor

Main role of a team predictor is to predict weights of every team relatively to the new instance to be classified based on learning results prepared by the training manager. Difference between team predictors is based on different decision metarule used to calculate weights.

We supposed to use a team predictor which decision metarule is based on the Weighted Nearest Neighbors (WNN) algorithm [Cost and Salzberg, 1993]. WNN uses results of the learning phase prepared by the training manager i.e. the performance matrix for all completed teams for every instance. In the application phase, nearest neighbors of a new instance among the training instances are found out and corresponding team performances are used to calculate the predicted performance for every team (a degree of belief to the team in the area of the new instance). In this calculation WNN sums up corresponding performance values of a team using weights that depend on the distances between the new instance and its nearest neighbors. The use of WNN as a team predictor is agreed with the assumption that each team has certain subdomains in the space of instance attributes, where it is more reliable than the others. This assumption is supported by the experiences, that classifiers usually work well not only in certain points of the domain space, but in certain subareas of the domain space [Quinlan, 1993]. If a team does not work well with the instances near a new instance, then it is quite probable that it will not work well with this new instance also.

4.4 Team Integrator

Main role of a team integrator is to produce classification result for a new instance by integrating appropriate outcomes of learned teams using a decision metarule. Difference between team integrators is based on a difference in a decision metarule used, which can be either dynamic integration of teams' results by weighted or unweighted voting or dynamic selection of the best team result.

We supposed to use a team predictor which decision metarule is based on the Weighted Voting algorithm [Bauer and Kohavi, 1998] based on weighted average over all teams' classification results.

5 Dynamic Selection Theorem

Let us assume that we have two different classification teams and they have been learned on a same sample set with n instances. Let the first team classifies correctly m_1, and the second one m_2 sample instances respectively. Assume that both teams correctly classified k same instances. We consider two possible team selection cases: a *static selection* and a *dynamic selection*.

Static selection means that we try all teams on a sample set and for further classification select one, which achieved the best classification accuracy among others for the whole sample set, i.e. we select a team only once and then use it to classify all new instances.

Dynamic selection means that the appropriate team is being selected for every new instance depending on where this instance located in a domain space. If, based on nearest sample neighbors, it has been predicted that certain team can better classify this new instance than other teams, then this team is used to classify this new instance. In such case we say that the new instance belongs to the "competence area" of that team.

The following theorem compares these two approaches to select teams for classification.

Theorem:

The average classification accuracy in the case of (dynamic) selection of a team for every instance is expected to be not worse than the one in the case of (static) selection the best classifier for the whole domain. The accuracy of these two cases can be equal if and only if $\min(m_1, m_2) = k$, where m_1, m_2, and k are as they were mentioned above

Proof:

The overall classification accuracy of both teams on the sample set can be evaluated as the probability of corrects classification of sample instances as follows:

$$P_1 = \frac{m_1}{n},\ P_2 = \frac{m_2}{n}$$

In a static selection the team with the best classification accuracy among others achieved on the sample set is selected to further classify the domain instances. Thus the classification accuracy P^I for the static case is:

$$P^I = \frac{\max(m_1, m_2)}{n}.$$

In a dynamic selection case the team with the best predicted accuracy is selected separately for every instance, according to which team "competence area" this instance belongs to. The competence area of the first team includes m_1 instances and the probability that some instance finds itself in this area is equal to P_1. The competence area of the second team includes m_2 instances and the respective probability is equal to P_2. There are k common instances among m_1 and m_2, which belong to the intersection of appropriate competence areas. Thus the classification accuracy P^{II} for the dynamic case can be evaluated as follows:

$$P^{II} = P_1 + P_2 - \frac{k}{n} = \frac{m_1}{n} + \frac{m_2}{n} - \frac{k}{n} = \frac{m_1 + m_2 - k}{n}.$$

Thus we have:

$$P^{II} = \frac{m_1 + m_2 - k}{n} \overset{k \le \min(m_1, m_2)}{\Longrightarrow} P^{II} \ge$$

$$\ge \frac{m_1 + m_2 - \min(m_1, m_2)}{n} \Rightarrow$$

$$\Rightarrow P^{II} \ge \frac{\max(m_1, m_2)}{n} \Rightarrow P^{II} \ge P^I.$$

Now we prove the second part of the Theorem:

$$a)\ P^I = \frac{\max(m_1, m_2)}{n};$$

$$\min(m_1, m_2) = k \Rightarrow P^{II} = \frac{m_1 + m_2 - k}{n} =$$

$$= \frac{\max(m_1, m_2) + \min(m_1, m_2) - k}{n} =$$

$$= \frac{\max(m_1, m_2) + k - k}{n} = \frac{\max(m_1, m_2)}{n};$$

$$\underline{\min(m_1, m_2) = \mathrm{k} \Rightarrow P^I = P^{II};}$$

$$b)\ P^I = P^{II} \Rightarrow \frac{\max(m_1, m_2)}{n} = \frac{m_1 + m_2 - k}{n} \Rightarrow$$

$$\Rightarrow \max(m_1, m_2) = \max(m_1, m_2) +$$

$$+ \min(m_1, m_2) - k \Rightarrow \min(m_1, m_2) = k;$$

$$\underline{P^I = P^{II} \Rightarrow \min(m_1, m_2) = \mathrm{k};} \quad \blacksquare$$

This theorem can be easily expanded to the case of several (more than two) teams. It shows that the approach with the dynamic selection of an appropriate team for every new case should outperform the static selection of a best team once for all the instances.

6 Implementation in Mobile Electronic Commerce

Advances in wireless network technology and the continuously increasing number of users of hand held terminals make the latter a possible channel for offering personalized services to mobile users and give space to the rapid development of Mobile Electronic Commerce (m-commerce). Ericsson, Motorolla and Nokia, as key facilitators of the mobile Internet and the mobile information society, together have established the MeT initiative [MeT, 2000], the first major cooperative effort to provide common open-core technology for the m-commerce market. MeT is targeted to define a common framework for m-commerce applications by extending existing industry standards and technologies and to ensure a consistent user experience across multiple phones, access technologies and usage scenarios.

An m-commerce user has any time certain location, which contains features of geographical coordinates and preferences. Consider m-commerce-brokering service, which can use this location information to find and offer to the user the most appropriate e-service among available ones. For such a location-based service we can use the above framework of virtual classifiers by a following way. We have a set of mobile objects - users. Every user has a set of features (location including preferences). We have also set of class labels - e-services. Assume that we have a collected database of an assignment: user - e-service. The problem is to assign for a new user appropriate (the best or nearest one) e-service, i.e. to classify a user in terms of e-services.

Feature selectors in this case are filtering techniques that are making user profile based on proper selection of location features relevant for precise classification.

Distance evaluators in m-commerce context are profile matching techniques, which can measure a distance between user profile and certain e-service's history cases profiles in its nearest neighborhood.

Classifiers here are engines that finally assign the best profile match (i.e. e-service) to a certain user in its current location. One of well-known examples of such an engine can be e-Speak [e-Speak, 2000] of Hewlett Packard.

The virtual classifier approach above allows in the best way to select and integrate profiling, filtering and profile matching techniques to connect dynamically a mobile user with the best available e-service in Web.

7 Conclusions

In this paper we base on the assumption that knowledge discovery with an ensemble of classifiers is more accurate than with any classifier alone [Dietterich, 1997]. We try to go further with this assumption. If a classifier somehow consists of certain feature selection algorithm, distance evaluation function and decision rule, then why not to consider these parts also as ensembles making a classifier itself more flexible? In such case we can also assume that the multiple ensembles (we call them teams) completed from different feature selection, distance evaluation, and decision-making methods will be more accurate in classification than any ensemble of known classifiers alone. Teams with methods of their training form virtual classifiers that differ according to the way of training and combining teams.

The next step in the assumptions might be consideration of ensembles of virtual classifiers. Recursively we come to the virtual metaclassifier framework where teams are completed from different virtual classifiers. Thus there is no limit to improve reasonably the accuracy of the classification by making more flexible classification techniques. On the other hand within the dynamic integration framework any classification method is no bad and can be used if at least in one case and in one team it works better than others.

Implementation shows that the framework can be used in mobile electronic commerce environment to integrate known methods and concepts of location, personalization, and profile matching to provide a mobile user with the best e-service in any place and in any time.

Acknowledgement

The economic support for this research from the Academy of Finland and later from the National Technology Agency of Finland (TEKES), Nokia, Hewlett Packard Finland, and Yomi Vision under contract 40599/99 as well as support from Information Technology Research Institute (Univ. of Jyvaskyla) is highly appreciated.

References

[Bauer and Kohavi, 1998] Eric Bauer and Ron Kohavi. An empirical comparison of voting classification algorithms: bagging, boosting, and variants. *Machine Learning* 33, 1998.

[Cost and Salzberg, 1993] Scott Cost and Steven Salzberg. A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning* 10(1): 57--78, 1993.

[Dash and Liu, 1997] M. Dash and H. Liu. Feature selection for classification. *Intelligent Data Analysis* 1 (3) 1997.

[Dietterich, 1997] Thomas G. Dietterich. Machine learning research: four current directions. *AI Magazine* 18(4):97--136, Winter 1997.

[Doak, 1992] Justin Doak. An evaluation of feature selection methods and their application to computer security. *Technical report CSE-92-18*, Davis, CA: Univ. of California, Dep. of Computer Science, 1992.

[e-Speak, 2000] e-Speak Overview, available in: http://www.e-speak.net/overview.html.

[Fayyad *et al.*, 1997] Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth and Ramasamy Uthurusamy. *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, 1997.

[Jorgensen, 1997] Peter Jorgensen. *P Jorgensen's COM515 Project Page*, Fall 1997, Available in WWW: http://jorg2.cit.bufallo. edu/COM515/project.html.

[Kohavi, 1995] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. *Proceedings of the Fourteen International Joint Conference on Artificial Intelligence*, pages 1137--1143, Montreal, Quebec, Canada, August 1995.

[Koppel and Engelson, 1996] Moshe Koppel and Sean P. Engelson. Integrating multiple classifiers by finding their areas of expertise. *Proceedings of the AAAI-96 Workshop On Integrating Multiple Learning Models for Improving and Scaling Machine Learning Algorithms*, pages 53--58, Portland, Oregon, August 1996.

[Merz, 1996] Christopher J. Merz. Dynamical selection of learning algorithms. *Learning from Data: Artificial Intelligence and Statistics*, NY: Springer Verlag, New York 1996.

[MeT, 2000] MeT Initiative - Enabling Mobile E-Commerce, *Met Overview White Paper*, 2 October, 2000, available in: http://www.mobiletransaction.org/pdf/MeT_WhitePaper.pdf.

[Quinlan, 1993] Ross Quinlan. *C4.5 Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.

[Rada *at al.*, 1989] Roy Rada, Hafedh Mili, Ellen Bicknell and Maria Blettner. Development and application of a metric on semantic nets. *IEEE Transactions on Systems, Man, and Cybernetics* 19(1): 17--30, January 1989.

[Rocha, 1991] Luis M. Rocha. Fuzzification of conversation theory, *Principia Cybernetica Conference*, Free University of Brussels, Brussels, June 1991.

[Siedlecki and Sklansky, 1988] W. Siedlecki and Jack Sklansky. On automatic feature selection. *International Journal of Pattern Recognition and Artificial Intelligence* 2: 197-220, 1988.

[Skalak, 1997] David B. Skalak. *Combining Nearest Neighbor Classifiers*. Ph.D. Thesis, Department of Computer Science, University of Massachusetts, Amherst, MA, 1997.

[Tudhope and Tailor, 1997] Douglas Tudhope and Carl Tailor. Navigation via similarity: automatic linking based on semantic closeness. *Information Processing and Management* 33(2): 233--242, 1997.

[Terziyan *at al.*, 1998] Vagan Terziyan, Alexey Tsymbal and Seppo Puuronen. The Decision Support System for Telemedicine Based on Multiple Expertise. *International Journal of Medical Informatics* 49(2): 21--229, April 1998.

[Tsymbal *at al.*, 1998] Alexey Tsymbal, Seppo Puuronen and Vagan Terziyan. Advanced Dynamic Selection of Diagnostic Methods. *Proceedings of the Eleventh IEEE Symposium on Computer-Based Medical Systems*, pages 50--54, IEEE CS Press, June 1998.

[Wilson and Martinez, 1997] D. Randall Wilson and Tony R. Martinez. Improved heterogeneous distance functions. *Journal of Artificial Intelligence Research* 6: 1--34 , 1997.

[Wolpert, 1992] David H. Wolpert. Stacked generalization. *Neural Networks* 5: 241-259, 1992.