Efficient Case Based Feature Construction for Heterogeneous Learning Tasks

Ingo Mierswa and Michael Wurst

No. CI-194/05

Technical Report ISSN 1433-3325 April 2005

This work is a product of the Collaborative Research Center 531, "Computational Intelligence," at the University of Dortmund and was printed with financial support of the Deutsche Forschungsgemeinschaft.

Efficient Case Based Feature Construction for Heterogeneous Learning Tasks[*]

Ingo Mierswa and Michael Wurst

Artificial Intelligence Unit, Department of Computer Science, University of Dortmund, Germany

Abstract

Feature construction is essential for solving many complex learning problems. Unfortunately, the construction of features usually implies searching a very large space of possibilities and is often computationally demanding. In this work, we propose a case based approach to feature construction. Learning tasks are stored together with a corresponding set of constructed features in a case base and can be retrieved to speed up feature construction for new tasks. The essential part of our method is a new representation model for learning tasks and a corresponding distance measure. Learning tasks are compared using relevance weights on a common set of base features only. Therefore, the case base can be built and queried very efficiently. In this respect, our approach is unique and enables us to apply case based feature construction not only on a large scale, but also in distributed learning scenarios in which communication costs play an important role. We derive a distance measure for heterogenious learning tasks by stating a set of necessary conditions. Although the conditions are quite basic, they constraint the set of applicable methods to a surprisingly small number. We also provide some experimental evidence for the utility of our method.

[*]This work was supported by the Deutsche Forschungsgemeinschaft (DFG) as a part of the Collaborative Research Center "Computational Intelligence" (SFB 531)

1 Introduction

Many inductive learning problems cannot be solved accurately by using the original feature space. This is due to the fact that standard learning algorithms cannot represent complex relationships as induced for example by trigonometric functions. Feature construction can help to deal with such problems by enriching the original feature space with additional features [6, 8, 13]. Many algorithms can be formulated in terms of kernel functions [16, 19]. Since the search for an appropriate kernel is equivalent to the search for an appropriate feature space transformation the success of these algorithms also underlines the need for feature construction.

Feature construction is an extension of feature selection which aims to find good subsets of features. Existing approaches include filtering methods like FOCUS [2] or Relief [11] and wrappers like forward selection, backward elimination [1] or genetic algorithms [22]. The latter can easily be extended by the construction or extraction of new features [14, 18] A (manual) selection and construction of the optimal feature set can be guided by some relevance measure of the features. Several approaches for feature relevance calculation exist, based for example on the entropy [15], the weighting vector derived of the hyperplane of a Support Vector Machine [19], or wrapper approaches [12]. Statistical algorithms for dimensionality reduction like Principal Component Analysis [10] also induce a weighting of the features. Unfortunately, feature construction is a computationally very demanding task often requiring to search a very large space of possibilities [21].

In this work we consider a scenario in which several learners face the problem of feature construction on different learning problems. The idea is to transfer constructed features between similar learning tasks to speed up the generation in such cases in which a successful feature has already been generated by another feature constructor. Such approaches are usually referred to as Meta Learning [20].

Meta Learning was applied to a large variety of problems and on different conceptual levels. The importance of the representation bias, which is closely related to feature construction, was recognized since the early days of Meta Learning research [3, 4].

The key to many Meta Learning methods is the definition of similarity between different learning tasks [5, 17]. In this work we propose a Meta Learning scheme that compares two learning tasks using only relevance weights assigned to a set of base features by the individual learners. Our approach is motivated by our work in the field of distributed multimedia learning. Concretely, we face a scenario in which several users independently arrange multimedia items according to some personal classification scheme. New items should be added to this scheme automatically by classification using a set of base features which is the same for all classification schemes and a set of features generated to increase the accuracy for an individual classification problem. Although the classification schemes created by the users may differ in any possible way, we can assume that at least some of them resemble each other to some extend, e. g. many users arrange their music according to the genre.

This scenario poses some additional constraints on Meta Learning. Firstly, the retrieval of similar learning tasks and relevant features has to be very efficient, as the system is designed for interactive work. This also means that methods should enable a best effort strategy, such that the user can stop the retrieval process at any point and get the current best result. Secondly, the system should scale well with an increasing number of users and thus learning tasks. Also, it has to deal with a large variety of heterogeneous learning tasks, as we cannot make any strict assumptions on the classification problems users create. Finally, as the system is distributed, communication cost should be as low as possible. As a consequence, methods that are based on exchanging examples or many feature vectors are not applicable. In our opinion, these constraints are quite general and apply to many Meta Learning problems in heterogeneous, distributed domains, such as distributed data mining or robot learning [17].

Considering the given constraints we develop and analyze a method for feature construction based on Meta Learning that estimates the distance of two learning tasks using only base feature weights. All learners weight a common set of base features according to their relevance to the given learning task. Some of the learners, that have already performed feature construction, send these relevance weights and the constructed features to a case base. From this case base, learners can retrieve possibly relevant features by sending their own base feature weights. The central challenge is to find a measure that compares two learning tasks using only base feature weights.

1.1 Outline

We give some definitions and conceptual ideas in section 2. In section 3 we analyze the problem of comparing learning tasks based only on feature weights by stating some basic conditions. In section 4, negative results are discussed for some well known weighting approaches and distance measures. Finally, we show in section 5 that the weights calculated by Support Vector Machines in combination with the manhattan distance fulfill all conditions described in section 3. These theoretical results are also confirmed by experiments that will be discussed in section 6.

2 Basic Concepts

Before we state the conditions which must be met by any method comparing learning tasks using feature weights only, we first introduce some basic definitions. Let T be the set of all learning tasks, a single task is denoted by t_i. Let X_i be a vector of random variables for task t_i and Y_i another random variable, the target variable. These obey a fixed but unknown probability distribution $Pr(X_i, Y_i)$. The components of X_i are called *features* X_{ik}. The objective of every *learning task* t_i is to find a function $h_i(X_i)$ which predicts the value of Y_i. We assume that each set of features X_i is partitioned in a set of *base features* X_B which are common for all learning tasks $t_i \in T$ and a set of *constructed features* $X_i \setminus X_B$.

We now introduce a very simple model of feature relevance and interaction. The feature X_{ik} is assumed to be irrelevant for a learning task t_i if it does not improve the classification accuracy:

Definition 1. *A feature X_{ik} is called* IRRELEVANT *for a learning task t_i iff X_{ik} is not correlated to the target feature Y_i, i. e. if $Pr(Y_i|X_{ik}) = Pr(Y_i)$.*

The set of all irrelevant features for a learning task t_i is denoted by IF_i.

Two features X_{ik} and X_{il} are alternative for a learning task t_i, denoted by $X_{ik} \sim X_{il}$ if they can be replaced by each other without affecting the classification accuracy. For linear learning schemes this leads to:

Definition 2. *Two features X_{ik} and X_{il} are called* ALTERNATIVE *for a learning task t_i (written as $X_{ik} \sim X_{il}$) iff $X_{il} = a + b \cdot X_{ik}$ with $b > 0$.*

This is a very limited definition of alternative features. However, we will show that most weighting algorithms are already ruled out by conditions based on this simple definition.

3 Comparing Learning Tasks Efficiently

The objective of our work is to speed up feature construction and improve prediction accuracy by building a case base containing pairs of learning tasks and corresponding sets of constructed features. We assume that a learning task t_i is completely represented by a feature weight vector w_i. The vector w_i is calculated from the base features X_B only. This representation of learning tasks is motivated by the idea that a given learning scheme approximate similar constructed features by a set of base features in a similar way, e. g. if the constructed feature "$\sin(X_{ik} \cdot X_{il})$" is highly relevant the features X_{ik} and X_{il} are relevant as well.

Our approach works as follows: for a given learning task t_i we first calculate the relevance of all base features X_B. We then use a distance function $d(t_i, t_j)$ to find the k most similar learning tasks. Finally, we create a set of constructed features as union of the constructed features associated with these tasks.

This set is then evaluated on the learning task t_i. If the performance gain is sufficiently high (above a given threshold) we store task t_i in the case base as additional case. Otherwise, the constructed features are only used as initialization for a classical feature construction that is performed locally. If this leads to a sufficiently high increase in performance, the task t_i is also stored to the case base along with the locally generated features.

While feature weighting and feature construction are well studied tasks, the core of our algorithm is the calculation of d using only the relevance values of the base features X_B. In a first step, we define a set of conditions which must be met by feature weighting schemes. In a second step, a set of conditions for learning task distance is defined which makes use of the weighting conditions.

Weighting Conditions. *Let w be a* WEIGHTING FUNCTION *$w : X_B \rightarrow \mathbb{R}$. Then the following must hold:*

(W1) $w(X_{ik}) = 0$ *if* $X_{ik} \in X_B$ *is irrelevant*

(W2) $F_i \subseteq X_B$ *is a set of alternative features. Then*

$$\forall S \subset F_i : \sum_{X_{ik} \in S} w(X_{ik}) = \sum_{X_{ik} \in F_i} w(X_{ik}) = \hat{w}$$

(W3) $w(X_{ik}) = w(X_{il})$ *if* $X_{ik} \sim X_{il}$

(W4) Let AF be a set of features where

$$\forall X_{ik} \in AF : X_{ik} \in IF_i \quad \vee \quad \exists X_{il} \in X_B : X_{ik} \sim X_{il}.$$

Then

$$\forall X_{il} \in X_B : \nexists X_{ik} \in AF : X_{il} \sim X_{ik} : w'(X_{il}) = w(X_{il})$$

where w' is a weighting function for

$$X'_B = X_B \cup AF$$

These conditions state that irrelevant features have weight 0 and that the sum of weights of alternative features must be constant independently of the actual number of alternative features used. Together with the last condition it will be guaranteed that a set of alternative features is not more important than a single feature. In the following we assume that for a modified space of base features X'_B the function w' denotes the weighting function for X'_B according to the definition in (W4).

Additionally, we can define a set of conditions which must be met by distance measures for learning tasks which are based on feature weights only:

Distance Conditions. *A* DISTANCE MEASURE *d for learning tasks is a mapping $d : T \times T \rightarrow \mathbb{R}^+$ which should fulfill at least the following conditions:*

(D1) $d(t_1, t_2) = 0 \Leftrightarrow t_1 = t_2$

(D2) $d(t_1, t_2) = d(t_2, t_1)$

(D3) $d(t_1, t_3) \leq d(t_1, t_2) + d(t_2, t_3)$

(D4) $d(t_1, t_2) = d(t'_1, t'_2)$ *if*

$X'_B = X_B \cup IF$ *and* $IF \subseteq IF_1 \cap IF_2$

(D5) $d(t_1, t_2) = d(t'_1, t'_2)$ *if*

$X'_B = X_B \cup AF$ *and*

$\forall X_k \in AF : \exists X_l \in X_B : X_k \sim X_l$

(D1)–(D3) represent the conditions for a metric. These conditions are required for efficient case retrieval and indexing, using e. g. M-Trees [7]. (D4) states that irrelevant features should not have an influence on the distance. Finally, (D5) states that adding alternative features should not have an influence on distance.

4 Negative results

In this section we will show that many feature weighting approaches do not fulfill the conditions (W1)–(W4). Furthermore, one of most popular distance measures, the euclidian distance, cannot be used as a learning task distance measure introduced above.

Lemma 1. *Any feature selection method does not fulfill the conditions (W1)–(W4).*

Proof. For a feature selection method, weights are always binary, i. e. $w(X_{ik}) \in \{0, 1\}$. We assume a learning task t_i with no alternative features and $X'_B = X_B \cup \{X_{ik}\}$ with $\exists X_{il} \in X_B : X_{il} \sim X_{ik}$, then either $w'(X_{il}) = w'(X_{ik}) = w(X_{il}) = 1$, leading to a contradiction with (W2), or $w'(X_{il}) \neq w'(X_{ik})$ leading to a contradiction with (W3). □

Lemma 2. *Any feature weighting method for which $w(X_{ik})$ is calculated independently of $X_B \setminus X_{ik}$ does not fulfill the conditions (W1)–(W4).*

Proof. We assume a learning task t_i with no alternative features and $X'_B = X_B \cup \{X_{ik}\}$ with $\exists X_{il} \in X_B : X_{il} \sim X_{ik}$. If w is independent of $X_B \setminus X_{ik}$ adding X_{ik} would not change the weight $w'(X_{il})$ in the new feature space X'_B. From (W3) follows that $w'(X_{ik}) = w'(X_{il}) = w(X_{il})$ which is a violation of (W2). □

Lemma 2 essentially covers all feature weighting methods that treat features independently such as information gain [15] or Relief [11].

The next theorem states that the euclidian distance cannot be used as a distance measure based on feature weights.

Theorem 3. *Euclidean distance does not fulfill the conditions (D1)–(D5).*

Proof. We give a counterexample. We assume that a weighting function w is given which fulfills the conditions (W1)–(W4). Further assume that learning tasks t_i, t_j are given with no alternative features. We add an alternative feature X_{ik} to X_B and get $X'_B = X_B \cup \{X_{ik}\}$ with $\exists X_{il} \in X_B : X_{il} \sim X_{ik}$. We infer from conditions (W2) and (W3) that

$$w'(X_{ik}) = w'(X_{il}) = \frac{w(X_{il})}{2}$$
$$w'(X_{jk}) = w'(X_{jl}) = \frac{w(X_{jl})}{2}.$$

and from condition (W4) that

$$\begin{aligned}\forall p \neq k : & w'(X_{ip}) = w(X_{ip}) \\ \forall p \neq k : & w'(X_{jp}) = w(X_{jp}).\end{aligned}$$

In this case the following holds for the euclidian distance

$$\begin{aligned}d(t'_i, t'_j) &= \sqrt{S + 2\left(w'(X_{ik}) - w'(X_{jk})\right)^2)} \\ &= \sqrt{S + 2\left(\frac{w(X_{ik})}{2} - \frac{w(X_{jk})}{2}\right)^2} \\ &= \sqrt{S + \frac{1}{2}\left(w(X_{ik}) - w(X_{jk})\right)^2} \\ &\neq \sqrt{S + \left(w(X_{ik}) - w(X_{jk})\right)^2} \\ &= d(t_i, t_j)\end{aligned}$$

with

$$\begin{aligned}S &= \sum_{p=1, p\neq k}^{|X_B|} \left(w'(X_{ip}) - w'(X_{jp})\right)^2 \\ &= \sum_{p=1, p\neq k}^{|X_B|} \left(w(X_{ip}) - w(X_{jp})\right)^2 . \qquad \square\end{aligned}$$

5 Positive results

We have shown that many common feature weighting algorithms and distance measures cannot be used for learning task distance in our scenario. In this section we will prove that the feature weights delivered by a linear Support Vector Machine (SVM) obeys the proposed weighting conditions. Afterwards, we also discuss a distance measure fulfilling the distance conditions.

Support Vector Machines are based on the work of Vapnik in statistical learning theory [19]. They aim to minimize the regularized risk $R_{reg}[f]$ of a learned function f which is the weighted sum of the empirical risk $R_{emp}[f]$ and a complexity term $||w||^2$:

$$R_{reg}[f] = R_{emp}[f] + \lambda ||w||^2.$$

The result is a linear decision function $y = \text{sgn}(w \cdot x + b)$ with a minimal length of w. The vector w is the normal vector of an optimal hyperplane with a maximal margin to both classes. One of the strengths of SVMs is the use of kernel functions to extend the feature space and allow linear decision boundaries after efficient nonlinear transformations of the input [16]. Since our goal is the construction of (nonlinear) features during preprocessing we can just use the most simple kernel function which is the dot product. In this case the components of the vector w can be interpreted as weights for all features.

Theorem 4. *The feature weight calculation of SVMs with linear kernel function meets the conditions (W1)–(W4).*

Proof. Since these conditions can be proved for a single learning task t_i we write X_k and w_k as a shortcut for X_{ik} and $w(X_{ik})$.

(W1) We assume that the SVM finds an optimal hyperplane. The algorithm tries to minimize both the length of w and the empirical error. This naturally corresponds to a maximum margin hyperplane where the weights of irrelevant features are 0 if enough data points are given.

(W2) SVMs find the optimal hyperplane by minimizing the weight vector w. Using the optimal classification hyperplane with weight vector w can be written as

$$y = \text{sgn}\left(w_1 x_1 + \ldots + w_i x_i + \ldots + w_m x_m + b\right).$$

We will show that this vector cannot be changed by adding the same feature more than one time. We assume that all alternative features can be transformed into identical features by normalizing the data. Adding $k-1$ alternative features will result in

$$y = \text{sgn}\left(\ldots + \underbrace{\left(w_i^1 + \ldots + w_i^k\right)}_{\text{alternative features}} x_i + \ldots + b\right).$$

However, the optimal hyperplane will remain the same and does not depend on the number of alternative attributes. This means that the other values w_j will not be changed. This leads to

$$w_i = \sum_{l=1}^{k} w_i^l$$

which proofs condition (W2).

(W3) The SVM optimization minimizes the length of the weight vector w. This can be written as

$$w_1^2 + \ldots + w_i^2 + \ldots + w_m^2 \overset{!}{=} \min .$$

We replace w_i using condition (W2):

$$w_1^2 + \ldots + \left(\hat{w} - \sum_{j \neq i} w_j \right)^2 + \ldots + w_m^2 \overset{!}{=} \min .$$

In order to find the minimum we have to partially differentiate the last equation for all weights w_k:

$$\frac{\partial}{\partial w_k} \left(\ldots + \left(\hat{w} - \sum_{j \neq i} w_j \right)^2 + w_k^2 + \ldots \right) = 0$$

$$\Leftrightarrow 2w_k - 2 \left(\hat{w} - \sum_{j \neq i} w_j \right) = 0$$

$$\Leftrightarrow w_k + \sum_{j \neq i} w_j = \hat{w}$$

The sum on the left side contains another w_k. This leads to a system of linear equations of the following form:

$$\begin{array}{rcl} & & \vdots \\ \ldots + 0 \cdot w_i + \ldots + 2 \cdot w_k + \ldots & = & \hat{w} \\ & & \vdots \end{array}$$

Solving this system of equations leads to $w_p = w_q$ (condition (W3)).

(W4) We again assume that a SVM finds an optimal hyperplane given enough data points. Since condition (W1) holds adding an irrelevant feature would not change the hyperplane and thus the weighting vector w for the base features will remain. The proofs of conditions (W2) and (W3) state that the optimal hyperplane is not affected by alternative features as well. □

In order to calculate the distance of learning tasks based only on a set of base feature weights we still need a distance measure that met the conditions (D1)–(D5).

Theorem 5. *Manhattan distance does fulfill the conditions (D1)–(D5).*

Proof. The conditions (D1)–(D3) are fulfilled due to basic properties of the manhattan distance. Therefore, we only give proofs for conditions (D4) and (D5).

(D4) We follow from the definition of the manhattan distance that

$$\begin{aligned} d(t_i', t_j') &= \sum_{X_{ip}, X_{jp} \in X_B} |w_i'(X_{ip}) - w_j'(X_{jp})| + \\ & \underbrace{\sum_{X_{iq}, X_{jq} \in IF} |w_i'(X_{iq}) - w_j'(X_{jq})|}_{0} \\ &= d(t_i, t_j) \end{aligned}$$

from (W4).

(D5) Sketch We show the case for adding k features with $\forall X_{ik} : X_{ik} \sim X_{il}$ for a fixed $X_{il} \in X_B$:

$$\begin{aligned} d(t_i', t_i') &= \sum_{p=1, p \neq k}^{|X_B|} |w_i'(X_{ip}) - w_j'(X_{jp})| + \\ & (k+1) \cdot |w_i'(X_{ik}) - w_j'(X_{jk})| \\ &= \sum_{p=1, p \neq k}^{|X_B|} |w_i(X_{ip}) - w_j(X_{jp})| + \\ & |w_i(X_{ik}) - w_j(X_{jk})| \\ &= d(t_i, t_j) \end{aligned}$$

from (W4) and (W2). □

Therefore, we conclude that SVM feature weights in combination with manhattan distance fulfill the necessary constraints for a learning task distance measure based on feature weights.

6 Experiments

Experiments were performed on synthetical data. A case base containing maximal 1000 cases was generated. For each case the following was done:

Example set generation: 300 examples with five base features were generated. The target variable obeys a randomly created regression function containing the building blocks +, $*$, sin and exp. The maximal depth of the target function was 3, the probability for leafs which does not contain another function was 0.3. These parameters results in functions of type

$$X_5 \cdot e^{X_2} + \sin(X_3).$$

The features X_1 and X_4 were irrelevant for this learning task.

Weight calculation: The weights of the base features were calculated. We tried both linear SVM weights and weights calculated by Relief.

Feature construction: An evolutionary approach for feature construction was used to generate new features from the base features. We used the following parameters: 200 generations, population size 10, and crossover probability 0.5.

Adding the case: the weights and the constructed features were added to the case base. Please note that the case base does not contain the data itself, information about the target variable, or the learned hypothesis.

After creation of the case base the test data (200 example sets) was created in the same manner but without feature construction. We only performed feature construction on the test set to determine the optimal relative error which can be achieved in the optimal feature space. The case based feature construction was performed with the constructed features of the 20 cases with minimal distance. All experiments were performed with the machine learning environment YALE [9].

Figure 1 shows the results. The relative error using only base features is 7.00%. The best error which can be achieved with help of evolutionary feature construction is 2.93% (the lower bound in the plot). The upper bound indicates the average error which can be achieved by simply taking a random sample of 20 cases from the case base instead of using the distance measures suggested in this paper. This benchmark error is 4.70%.

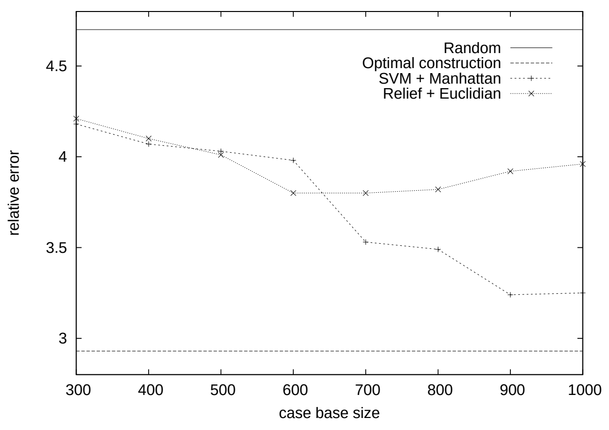

Figure 1: The results of case based feature construction. The averaged relative error of all 200 test cases is plotted against the number of cases used as case base. The combination SVM weights plus manhattan distance clearly outperforms the combination Relief plus euclidian distance.

The averaged relative errors of all 200 test cases was plotted against the number of cases used for the case base. The plot shows two curves for the combination of SVM feature weighting and manhattan distance and Relief weighting and euclidian distance. Both approaches reduced the error beneath the benchmark error of 4.70% for all examined case base sizes. As expected the combination SVM plus manhattan distance outperforms the combination of Relief plus euclidian distance, at least for sufficiently large case bases. This combination almost reaches the minimal error 2.93% achieved by evolutionary feature construction.

Our approach fulfills the constraints presented above and can speed up the process of feature construction considerably. The average time needed for the automatic construction of an optimal feature set without using the case base was 76.8 seconds. Using the feature construction induced by the 20 most similar cases reduces the time needed for learning to 0.8 seconds.

7 Conclusion and Outlook

We presented a Meta Learning approach to feature construction that compares tasks using relevance weights on a common set of base features only. After stating some very basic conditions for such a distance measure, we have shown that a SVM as base feature weighting algorithm and the manhattan distance fulfill these conditions, while several other popular feature weighting methods and distance measures do not. We have presented some experimental results indicating that our method can speed up feature construction considerably.

Some limitations of the work presented here are the following. Firstly, our definition for alternative or exchangeable features is rather simple and should be generalized to a weaker concept as e. g. highly correlated features. Also, complex interactions between features are not covered by our conditions. However, it is very interesting that the conditions stated in this work are already sufficient to rule out large sets of feature weighting methods and distance measures. Finally, the assumption of estimating the distance of constructed features by the distance of base features is well motivated, though it would be interesting to analyze this relationship analytically to get a better estimation in which cases our approach can be successfully applied.

For our future work on the subject we firstly plan to incorporate generated features in our distance measure. This distance measure should be used in a two step process, such that we first efficiently retrieve a larger number of cases similar on base features and then compare the generated features in this set with the query using methods that perform a syntactical comparison on the constructed features. Another direction of work is to analyze the relationship between base feature weights and generated features to get further insight which weighting methods are suitable. Finally, we plan to perform empirical experiments using real world data from the domain of distributed multimedia learning.

References

[1] David W. Aha and Richard L. Bankert. A comparative evaluation of sequential feature selection algorithms. In Doug Fisher and Hans-J. Lenz, editors, *Learning from Data*, chapter 4, pages 199–206. Springer, New York, 1996.

[2] H. Almuallim and T.G. Dietterich. Learning with many irrelevant features. In *9th National Conference on Artificial Intelligence*, pages 547–552. MIT Press, 1991.

[3] Jonathan Baxter. Learning internal representations. In *Proceedings of the eighth annual conference on Computational learning theory COLT '95*, pages 311–320. ACM Press, 1995.

[4] Jonathan Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12:149–198, 2000.

[5] Shai Ben-David and Reba Schuller. Exploiting task relatedness for multiple task learning. In *Proc. of the Sixteenth Annual Conference on Learning Theory COLT 2003*, 2003.

[6] Avrim L. Blum and Pat Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, pages 245–271, 1997.

[7] Paolo Ciaccia, Marco Patella, and Pavel Zezula. M-tree: An efficient access method for similarity search in metric spaces. In *Proceedings of 23rd International Conference on Very Large Data Bases VLDB'97*, pages 426–435. Morgan Kaufmann, 1997.

[8] M. Dash and H. Liu. Feature selection for classification. *International Journal of Intelligent Data Analysis*, 1(3):131–156, 1997.

[9] Simon Fischer, Ralf Klinkenberg, Ingo Mierswa, and Oliver Ritthoff. Yale: Yet Another Learning Environment – Tutorial. Technical Report CI-136/02, Collaborative Research Center 531, University of Dortmund, Dortmund, Germany, 2002. ISSN 1433-3325.

[10] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer series in statistics. Springer, 2001.

[11] K. Kira and I. A. Rendell. The feature selection problem: Traditional methods and a new algoirthm. In *10th National Conference on Artificial Intelligence*, pages 129–134. MIT Press, 1992.

[12] R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324, 1997.

[13] D. Koller and M. Sahami. Toward optimal feature selection. In *Proceedings of the Thirteenth International Conference on Machine Learning (ML96)*, pages 129–134, 1996.

[14] Ingo Mierswa and Katharina Morik. Automatic feature extraction for classifying audio data. *Machine Learning Journal*, 58:127–149, 2005.

[15] R.J. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[16] B. Schölkopf and A. J. Smola. *Learning with Kernels – Support Vector Machines, Regularization, Optimization, and Beyond.* MIT Press, 2002.

[17] S. Thrun and J. O'Sullivan. Discovering structure in multiple learning tasks: The TC algorithm. In L. Saitta, editor, *Proceedings of the 13th International Conference on Machine Learning ICML-96*, San Mateo, CA, 1996. Morgen Kaufmann.

[18] Haleh Vafaie and Kenneth De Jong. Evolutionary feature space transformation. In Huan Liu and Hiroshi Motoda, editors, *Feature Extraction, Construction, and Selection – A Data Mining Perpective*, pages 307–323. Kluwer, 1998.

[19] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory.* Springer, New York, 1995.

[20] Ricardo Vilalta and Youssef Drissi. A perspective view and survey of meta-learning. *Artificial Intelligence Review*, 18(2):77–95, 2002.

[21] D.H. Wolpert and W.G. Macready. No free lunch theorems for optimisation. *IEEE Trans. on Evolutionary Computation*, 1:67–82, 1997.

[22] Jihoon Yang and Vasant Honovar. Feature subset selection using a genetic algorithm. In Huan Liu and Hiroshi Motoda, editors, *Feature Extraction, Construction, and Selection – A Data Mining Perpective.* Kluwer, 1998.