IMPROVEMENT OF THE GENERIC CLASSIFICATION MODEL BASED ON A MULTIPLE KERNEL DATA FUSION

Kristina Spirovska
Faculty of Computer Science and Engineering
Skopje, Republic of Macedonia

Ana Madevska Bogdanova, Ph.D
Faculty of Computer Science and Engineering
Skopje, Republic of Macedonia

ABSTRACT

Creating a generic classification system has always been a challenging problem in the machine learning world. Our previous work [1] proposed a model of such system.
That model describes a system that does a data pre-processing, intelligent choice of a kernel-based classification method, automatically tuning the parameters, and proposes a solution. It's main goal is creation of a simple system that will focus the user on a method that he/she would further tune to obtain the final solution of a given classification problem.
This paper focuses on improvements of that model and presents the evolved model of the Generic Classification System based on a Multiple Kernel Data Fusion.

Keywords — models of classification systems, generic classification, multiple kernel data fusion, kernel parameters, parameter selection

I. INTRODUCTION

The problem of construction of a good classifier can be considered as one of the most explored and researched problem in the machine learning community. Indisputably, thousands of methods and algorithms are proposed as solution. But the tricky part about the classifiers is that they are problem-dependent and the quality of the learned models severely depends on the nature of the problem.

In order to get accurate models for the classification problems, we must first make the right choice of the learning algorithm and afterward choose the right parameters of the method, which is of crucial importance for the classification process.
In paper [1] we described a model of a generic classification system which is focused on multiple-kernel learning method. Here we propose improvements of that model.
To begin with, this section is devoted to a brief introduction to the main method for classification used in [1] - multiple kernel learning method. Section 2 presents the basic model of [1]. Section 3 introduces the improvements of the model. Finally, the last section concludes this work.

A. Kernel methods

In the past decade, kernel methods have proved to be one of the most effective tools in problems such as classification. One of the most basic and most widely used kernel method in machine learning used for classification is Support Vector Machine (SVM).
SVM and other kernel methods, approach the problem by mapping data into a high dimensional feature space where every coordinate represents one attribute of the data entities. The mapping is done by a kernel function. If we have two data instances x_i, $x_j \in X$ and a mapping ϕ: $X \rightarrow \Re^N$, we can define kernel function as:

$$k(x_i, x_j) = <\varphi(x_i), \varphi(x_j)> . \quad (1)$$

The data instances, x_i and x_j, could be elements from any set. On the other hand, their mappings $\phi(x_i)$ and $\phi(x_j)$ represents vector in $\Re^N$. The matrix $K_{ij}=(x_i,x_j)$, where x_i, x_j are data points, is called "kernel matrix" or "Gram matrix". The kernel matrix is a symmetric positive definite matrix. Because it specifies the inner products between all pairs of data points, it completely determines the relative positions between those points in the embedding space [2] [3].
In other words, a kernel matrix is a representation of the input data space through defined similarities between pairs of its instances, which is achieved through the kernel function [4]. Each kernel function provides a partial description or view of the data and extracts a specific type of information, i.e. the kernel function defines generalized similarity relationships between the pairs of instances.

B. Multiple kernel learning

Multiple kernel learning (MKL) has been pioneered by Lanckriet et al. [5] as an extension of the single kernel SVM. The idea behind MKL is the fact that multiple sources could be seen as multiple partial descriptions of the data instances. The benefit of using them comes from the fact that different data sources are likely to contain different and partly independent information about the task. If we combine those complementary pieces of information we might enhance the total information about the problem at hand [6].

An example could be predicting the function of a protein using multiple sources like variable length amino acid strings, real-valued gene expression data and a graph of protein-protein interactions.
During the past decade, it has been shown that classifiers that use combinations of multiple kernels instead of classical single kernel-based ones attain better results in certain problems [4] [20] [21].
This mathematical problem could be solved with combination of those partial descriptions using a convex optimization method known as semi-definite programming (SDP) [15] [16]. This SDP-based approach gives us a general methodology for combining many partial descriptions of data that is statistically sound, as well as computationally efficient and robust.
The solutions found by kernel-based algorithms such as the support vector machine (SVM) are linear functions in the feature space:

$$f(x) = w^T \Phi(x) . \quad (1)$$

for the optimal weight vector w. The kernel can be exploited whenever the weight vector can be expressed as a linear combination of the training points

$$w = \sum_{i=1}^{m} \alpha_i \Phi(x_i) . \quad (2)$$

, implying that we can express *f* as follows:

$$f(x) = \sum_{i=1}^{m} \alpha_i k(x_i, x) . \quad (3)$$

, an important issue in the kernel applications is the choice of a kernel k for a given learning task. Intuitively, we try to choose a kernel that induces the "right" metric in the space [16].

DESCRIPTION OF THE BASIC MODEL

The classification system that we described in paper [1] is a generic one, which means that it could be used for different kind of classification problems with heterogeneous data sources. The model of the system is given in Fig. 1 is completely described in paper [1], and here we will give just an rough and basic description.
As we can see from Fig. 1, it is composed of four main component units:

- Preprocessor unit
- Training unit
- Data Fusion Unit
- Classification Unit

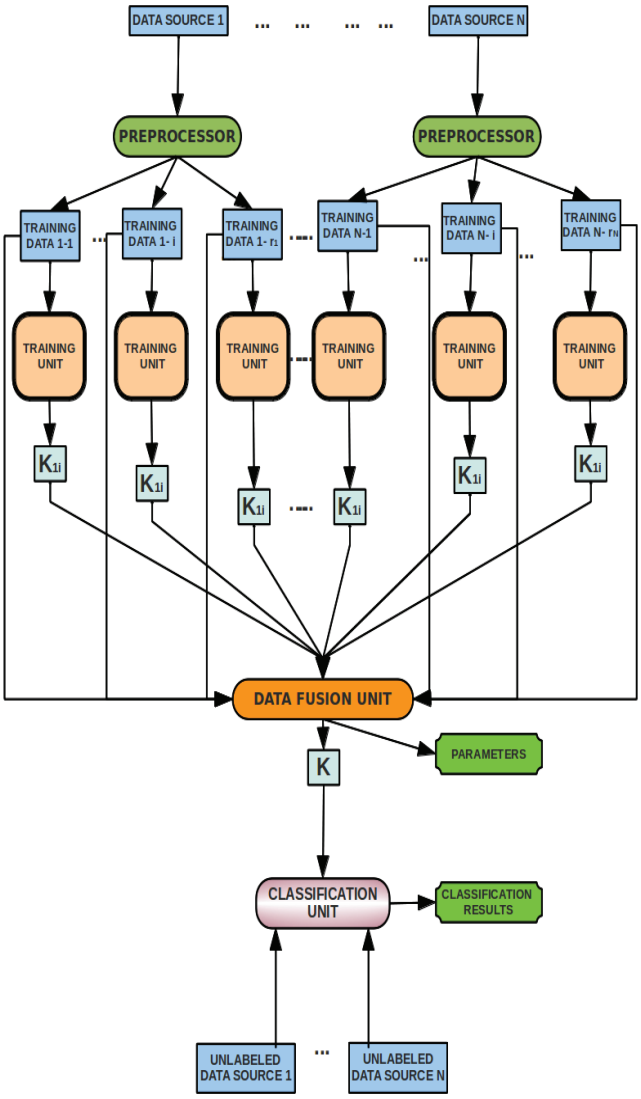
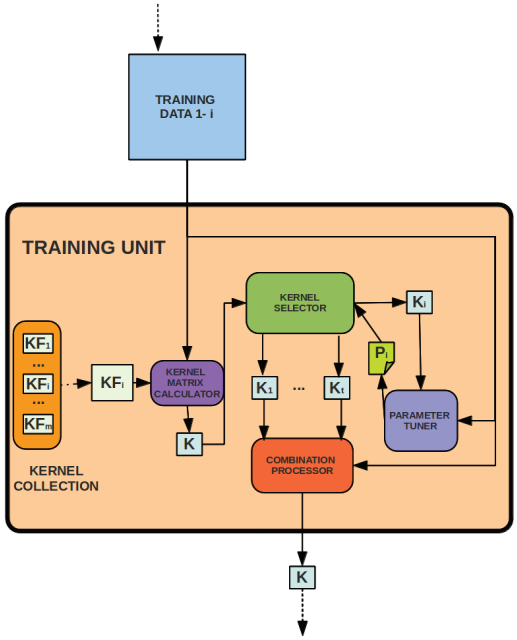


Figure 1: Data flow diagram of the generic classification system based on MKDF

The training unit is a complex unit whose role is to create kernel matrix representations from every training data source retrieved from the Preprocessor unit. Its data flow diagram is shown on Fig. 2.
The TU is composed of five components:

- Kernel Collection (KC)
- Kernel Matrix Calculator (KMC)
- Kernel Selector (KS)
- Parameter Tuner (PT)
- Combination Processor (CP) components.



The output from this unit is the kernel matrix build from the best fitting kernel function.Figure 2: Data flow diagram of the Training Unit

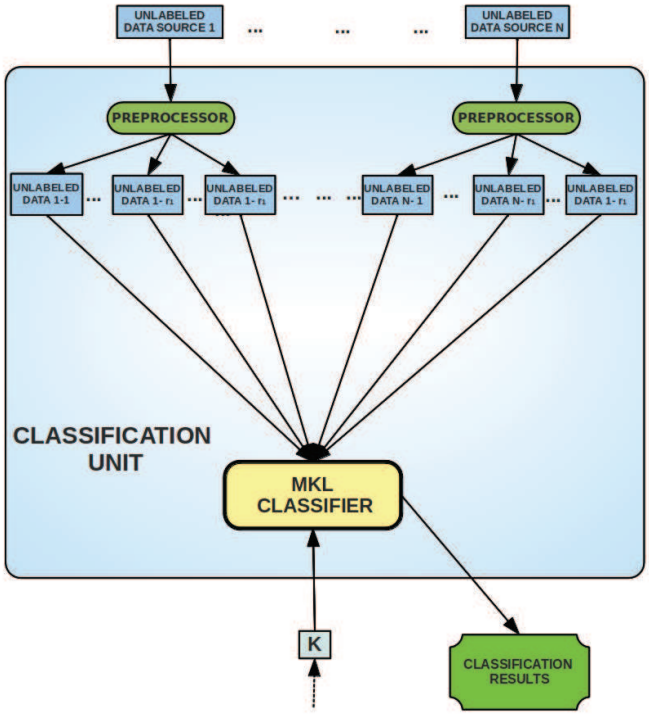


Figure 3: Data flow diagram of the Classification Unit

The output from the CU is the output from the whole system. It gives the classification results or the labels or classes of the unlabeled data that we wanted to classify / label. It's data flow diagram is shown on Fig. 3.

IMPROVEMENTS OF THE MODEL

In this section we will present some improvements of the model.

The first improvement was an automatic data sources separation process. The user would only provide a training set containing all the sources for the data, and the system will separate the data sources by the Sources Separation Unit (Fig. 5).

The second improvement is regarding the process of parameter calibration. As we mentioned earlier, the choice of the kernel function implies defining the mapping from the input space to the feature space. Because the problem is not a priori known, and data distribution may change in different feature spaces, the performance of the system depends greatly on the choice of the kernel. Nonetheless, there is no theoretical method for determining the kernel function and its parameters.This also applies for the SVM parameter C and fusion parameters too. There are different methods for determining these parameters such as grid search algorithm, genetic algorithms, simulated annealing techniques, heuristic techniques etc. In this system the grid search algorithm is used (Algorithm 1). In a grid search algorithm, a uniform grid is defined in the parameter space. After that, all the points in the grid are evaluated in order to find a global minimum. The algorithm will find the global minimum of all the points in the parameter grid. The coarseness of the grid determines the quality of the solution found and the efficiency of the search. [13,11,12]

Algorithm 1: Using the grid search method to choose a desired classifier.

```
for each C, kernel parameter combination

    train a classifier with the training data

    use the classifier to classify the validation
    data

pick the classifier with highest validation
accuracy for testing

process usage
```

In the model of [1] there are three places where determining of kernel or MKL fusion parameters takes place. They are the parameter tuner, the combination maker and the data fusion unit. Nevertheless are all of those tunning really necessary? Considering the accuracy gain and time consumed is it worthed to do so many tunings? The answer is no.

So that is why we propose a new and enhanced model. The improvements are made in the training unit which are shown on Figure 4.

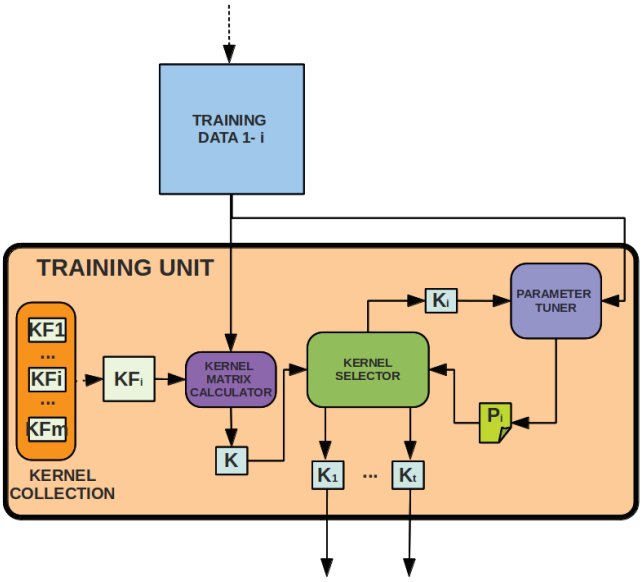


Figure 4: Data flow diagram of the improved Training Unit

It can be noticed that now the output of the training unit is not one kernel matrix, but t-kernel matrices. And one component of the previous model of the training unit is missing – the Combination Processor.

The Combination Processor combined the kernel matrices of the homogeneous data, or different representations of the same source.

After that that kernel matrix was used as an input to the Data Fusion unit. By experiments it has been shown that the precision gained by doing this model selection by the Combination processor does not pay off the time-consumed by this tunning. So in the improved model this component is left out. The new model is shown on Fig. 5.

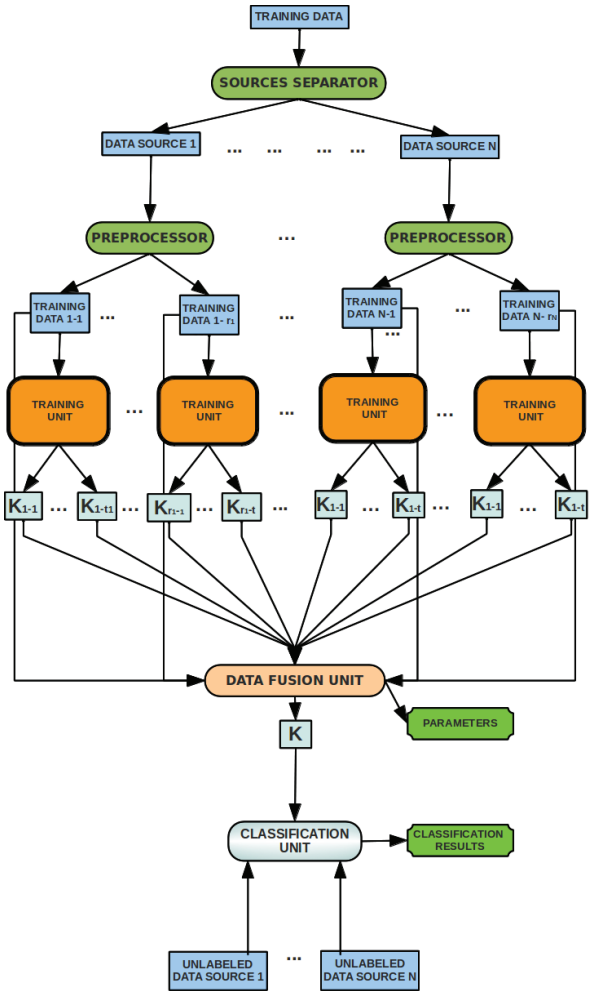


Figure 5: Data flow diagram of the improved generic classification system based on MKDF.

CONCLUSION

In this paper an improvement of the Model of Generic Classification System is proposed in order to reduce the time and resources cost and does not affect the accuracy of the system. The next step is implementation of the model and its parallelization.

REFERENCES

[1] Kristina Spirovska and Ana Madevska Bogdanova, "Model of a Generic Classification System based on a Multiple Kernel Data Fusionn," *ICT Innovations, Web Proceedings 2012*.

[2] Lanckriet, G.R.G., Cristianini, N., Bartlett, P.L., Ghaoui, L.E., Jordan, M.I.: Learning the Kernel Matrix with Semi-Definite Programming. ; In ICML(2002)323-330

[3] Yu, S., Tranchevent, L., Moor, B.D., Moreau, Y.: Kernel-based Data Fusion for Machine Learning - Methods and Applications in Bioinformatics and Text Mining. ;Studies in Computational Intelligence(2011)1-208

[4] Bach, F.R., Lanckriet, G.R.G., Jordan, M.I.: Multiple kernel learning, conic duality, and the SMO algorithm. ;In ICML(2004)

[5] Lanckriet, G.R.G., Cristianini, N., Bartlett, P.L., Ghaoui, L.E., Jordan, M.I.: L6. ;Journal of Machine Learning Research(2004)27-72

[6] Gert R. G. Lanckriet, Tijl De Bie, Nello Cristianini, Michael I. Jordan, William Stafford Noble; A Framework for Genomic Data Fusion and its Application to Membrane Protein Prediction (2003)

[7] Zhang, S., Zhang, C., Yang, Q.: Data Preparation for Data Mining; Applied Artificial Intelligence(2003)375-381

[8] Rakotomamonjy, A., Bach, F., Canu, S., Grandvalet, Y.: More efficiency in multiple kernel learning. ;In ICML(2007)775-782

[9] Sonnenburg, S., Rätsch, G., Schäfer, C., Schölkopf, B.: Large Scale Multiple Kernel Learning.;Journal of Machine Learning Research(2006)1531-1565

[10] Kuo-Ping Wu, Sheng-De Wang

„Choosing the kernel parameters for support vector machines by the inter-cluster
distance in the feature space
“, Pattern Recognition 42 (2009) 710 – 717

[11] C.-W. Hsu, C.-J. Lin, A comparison of methods for multiclass support vector
machines, IEEE Trans. Neural Networks 13 (2) (2002) 415–425.

[12] Bergstra, James; Bengio, Yoshua (2012). "Random Search for Hyper-Parameter Optimization". J. Machine Learning Research 13: 281—3
[13] Chin-Wei Hsu, Chih-Chung Chang and Chih-Jen Lin (2010). A practical guide to support vector classification. Technical Report, National Taiwan University.
[12] C.-W. Hsu, C.-J. Lin, A comparison of methods for multiclass support vector
machines, IEEE Trans. Neural Networks 13 (2) (2002) 415–425.