Tensor-based Restricted Kernel Machines for Multi-View Classification

Lynn Houthuys[1,*], Johan A. K. Suykens

Department of Electrical Engineering ESAT-STADIUS, KU Leuven
Kasteelpark Arenberg 10 B-3001 Leuven, Belgium

Abstract

Multi-view learning deals with data that is described through multiple representations, or views. While various real-world data can be represented by three or more views, several existing multi-view classification methods can only handle two views. Previously proposed methods usually solve this issue by optimizing pairwise combinations of views. Although this can numerically deal with the issue of multiple views, it ignores the higher order correlations which can only be examined by exploring all views simultaneously. In this work new multi-view classification approaches are introduced which aim to include higher order statistics when three or more views are available. The proposed model is an extension to the recently proposed Restricted Kernel Machine classifier model and assumes shared hidden features for all views, as well as a newly introduced model tensor. Experimental results show an improvement with respect to state-of-the art pairwise multi-view learning methods, both in terms of classification accuracy and runtime.

Keywords: Multi-view learning, Tensor, Kernel-based learning

1. Introduction

In many real world applications, data is described through a number of different features. Often, these features can be naturally partitioned into

*Corresponding author

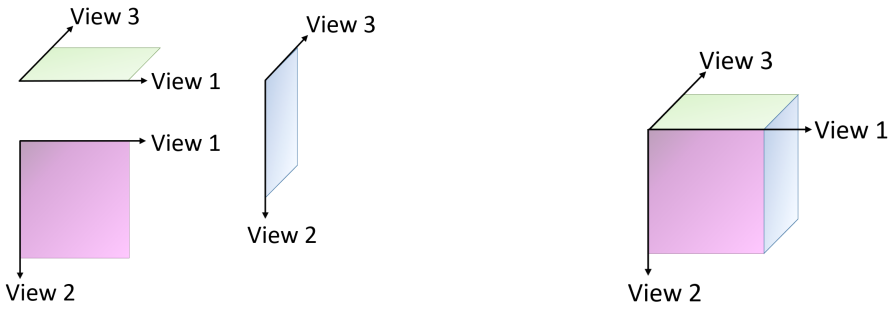
Email addresses: lynn.houthuys@esat.kuleuven.be (Lynn Houthuys), johan.suykens@esat.kuleuven.be (Johan A. K. Suykens)

[1]Present affiliation: Thomas More University of Applied Sciences, Jan De Nayerlaan 5 B-2860 Sint-Katelijne-Waver, Belgium

groups. Think, for example, about learning with social media data, where one could have features related to the user profile as well as features describing the friend links [1], or when predicting an Alzheimer's disease diagnosis both neuroimaging and genetics data could be used [2], and so on. These groups of features can be referred to as *views*, and multi-view learning techniques deal with data that is represented by multiple views.

Several existing multi-view classification techniques are a form of late fusion. This means that the coupling, or fusion, of the information from the different views, is done late in the training process. Typical examples are committee-like [3] methods, where different models are trained on all views separately and the final prediction is done using a (weighted) combination of these view-specific models, as was e.g. done by Wang et al. [4] for multi-view clustering. The advantage of late fusion techniques is that the separate submodels have a high degree of freedom to model the views differently, which is a strong advantage when the data is inherently different over the views. A drawback of late fusion is that since the information is coupled late, the submodels take little advantage of the information provided by the other views. Early fusion techniques aim to include the information from all views as soon as possible in the training process. A typical example is simply concatenating the features of all views, as was done e.g. by Karevan et al. [5] to perform temperature prediction based on measurements in multiple cities. In order to combine the advantages of both early and late fusion, some multi-view classification techniques aim to exploit the information from multiple views early on while still allowing for some degree of freedom to model the views differently.

Another issue for multi-view learning is the ability to handle more than two views. While data can often be represented by numerous views, several multi-view methods are only able to account for two views. E.g. multi-view GEPSVMs [6] is only defined for two views where it maximizes the agreement between them. Even the methods that can handle more than two views, often do this by coupling the views in a pairwise fashion. Multi-View Least Squares Support Vector Machines (MV-LSSVM) Classification [7], for example, consist of a coupling term that minimizes the product of the error variables of two views. For three or more views, the model optimizes the sum of the pairwise coupling terms. Another example is Multi-View Learning with Least Squares loss function, a multi-view semi-supervised classification model proposed by Minh et al. [8], that introduces between-view regularization by adding up pairwise regularization terms between two views. While



(a) Pairwise correlations (b) Higher-order correlations

Figure 1: An example with three views to motivate tensor learning in multi-view learning. (a) Standard coupling: only the pairwise correlations between the views are taken into account. (b) Tensor approach: the higher-order correlations between all views are modeled in a third order tensor.

this is a popular approach in the existing multi-view learning techniques, it fails to incorporate higher-order correlations (correlations between three or more views) that could only be discovered by simultaneously considering all views. This issue was raised by Luo et al. [9], where the authors propose an extension to Canonical Correlation Analysis (CCA) [10], called Tensor CCA, that analyzes a covariance tensor over the data from all views. See Figure 1 for an illustration of the advantage of using a tensor to account for higher-order correlations. While Tensor CCA is solved by performing a tensor decomposition, we will propose a model that results in a linear system, which decreases the computational cost significantly.

Note that the existence of higher-order correlations does not necessarily mean that the views are strongly correlated. As we know from previous work (e.g. Houthuys et al. [7]), when the information provided by the views is too similar it often reduces the performance, and hence this should be dealt with in pre-processing.

This paper explores novel multi-view classification methods which strive to incorporate higher-order correlations when three or more views are available by incorporating principles of tensor learning. They can furthermore be seen as a combination of both early and late fusion techniques. The proposed methods are extensions of the Restricted Kernel Machine (RKM) Classification [11] method where the RKM model is extended to the multi-view setting by assuming shared hidden features over all different views. To introduce more coupling, a weight tensor model is included which contains the weights

corresponding to all views. Three multi-view methods are proposed, where it is shown that the first two are special cases of the last method.

The main contributions of this paper can be summarized as:

- A novel multi-view classification model, called ϱTMV-RKM, is proposed.
- Tensor learning is incorporated to account for higher order correlations.
- Experimental results show the merit of using a weight tensor.
- Comparisons with other methods show improvement in both accuracy and time complexity.
- Multiple approaches to handle a large-scale dataset are proposed.

Related work. Tensor learning is appearing more and more in machine learning problems. For example, sometimes data is naturally described in a multidimensional manner. Video data, for example, can be described by a third-order tensor with each frame a matrix, and time being the third dimension. For instance, Vinayak et al. [12] propose a third order tensor to represent three questions queries in order to perform crowdsourced clustering. Cichochi et al. [13] and Sidiropoulos et al. [14] provide a thorough overview of the use of tensors and tensor decompositions in signal processing and machine learning.

Instead of introducing tensors at a data level, they can also be used to represent the model which is to be learned. For example, Signoretto et al. [15] propose a tensor-based framework that represents the model and show how it can be applied to learn from multi-dimensional data.

For multi-modal learning schemes, a third order tensor can be used where an extra dimension is added to a two-dimensional representation of the data. For example, Wimalawarne et al. [16] propose a multi-task learning scheme where the weight vectors belonging to the different tasks are stacked to form a third order weight tensor, where the third dimension indicates the task index. Similarly Adeli et al. [17] stacks the weight tensors corresponding to different measuring time points to perform multilinear regression for prediction of infant brain development. Liu et al. [18] and Wu et al. [19] stack similarity matrices for each view to form a similarity third order tensor, with the third view indicating the view index. Zhang et al. [20] and Xie et al. [21]

use a similar technique to perform multi-view clustering where the subspace representations of each view are stacked.

Instead of adding only one dimension to represent the index of the view, a tensor could also be used to model the interactions, or correlations, between the views (as shown in Figure 1). For example, Lu et al. [22] extends matrix factorization to multilinear factorization machines for multi-view multi-task learning. A weight model tensor is formulated to simultaneously model the weights corresponding to all views and tasks. The representation of the features is manipulated, however, such that only the lower-order (pairwise) interactions between the views are taken into account. Additionally, various multi-view dimensionality reduction methods [23, 24] incorporate tensor learning to account for higher-order correlations. Furthermore, Blondel et al. [25, 26] present an efficient algorithm to train higher-order factorization machines (HOFM), which model higher-order interactions between features. Different from the model proposed in our paper, also the lower-order correlations are taken into account. A low-rank tensor containing the weights of the feature combinations is used. While factorization machines can be used to perform multi-view learning, they generally disregard the view segmentation by exploring interactions between all features, regardless of the corresponding view. Cao et al. [27] furthermore extended factorization machines to explore the full-order interactions by proposing the Multi-View Machines (MVM) method. In contrast to factorization machines, MVM models the full-order interactions between views in a tensor which are factorized collectively.

Several subspace learning based multi-view methods also consider all views simultaneously [28, 29, 30, 31]. For example, Zheng et al. [28] perform low-rank regression in the subspace of each view, with a shared regression parameter matrix over all views. More recently, Yang et al. [29] proposed a method where the features are mapped to a discriminative low-dimensional subspace. Moreover, Xie & Sun [32] introduced a multi-view method for binary classification which contains a pairwise regularization term as well as a combination weight. The latter explores the complementary information among different views simultaneously.

Finally, multi-view learning is naturally related to the field of multiple kernel learning (MKL) [33, 34, 35], where a linear or non-linear combination of different kernel functions is used. Although generally MKL is used to model the same data with different feature maps, it can also be applied to different views.

Notation. We will denote matrices as bold uppercase letters, vectors as bold lowercase letters and higher-order tensors by calligraphic letters. The superscript $^{[v]}$ will denote the v -th view for the multi-view method. Whereas the superscript $^{(l)}$ will denote the l -th binary classification problem in case there are more than two classes.

2. Background

This section briefly reviews the methods Restricted Kernel Machine (RKM) and Multi-View Least Squares Support Vector Machines (MV-LSSVM).

2.1. RKM Classification

This section summarizes the *Restricted Kernel Machine (RKM)* classification model as described by Suykens [11] which is closely related to the well known *Least Squares Support Vector Machine (LS-SVM)* [36] model. In analogy with Restricted Boltzmann Machines (RBM) [37], RKM offers an expression in terms of visible and hidden layers related to respectively the primal and dual variables. The dual formulation is obtained by means of conjugate feature duality. Suykens [11] further shows how multiple RKM's can be stacked together to form a deep RKM formulation. As for LS-SVM, RKM uses the kernel trick to map the data into a high dimensional feature space in which one constructs a linear separating hyperplane.

By formulating a lower bound on the primal formulation of LS-SVM, one obtains the RKM objective. Given a training set of N data points $\{(\mathbf{x}_k, y_k)\}_{k=1}^N$ where $\mathbf{x}_k \in \mathbb{R}^d$ denotes the k-th input pattern and $y_k \in \{-1, 1\}$ the k-th label, the objective $\mathcal{J}$ of RKM classification is:

$$\mathcal{J} = \frac{\eta}{2}\mathbf{w}^T\mathbf{w} + \sum_{k=1}^{N} \left(1 - (\varphi(\mathbf{x}_k)^T\mathbf{w} + b)y_k\right) h_k - \frac{\lambda}{2}\sum_{k=1}^{N} h_k^2 \tag{1}$$

where b is a bias term, λ and η are positive real regularization constants and $h_k \in \mathbb{R}$ are the hidden features. The feature map $\varphi : \mathbb{R}^d \to \mathbb{R}^{d_h}$, which maps the input to a high dimension, is usually not explicitly defined but rather implicitly by the use of the kernel trick. Based on Mercer's theorem [38] we can use a positive definite kernel function $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ and define $K(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i)^T\varphi(\mathbf{x}_j)$. This allows to work in a high, even infinite, dimensional feature space without having to explicitly define it. The RKM model can be represented graphically as in Figure 2.

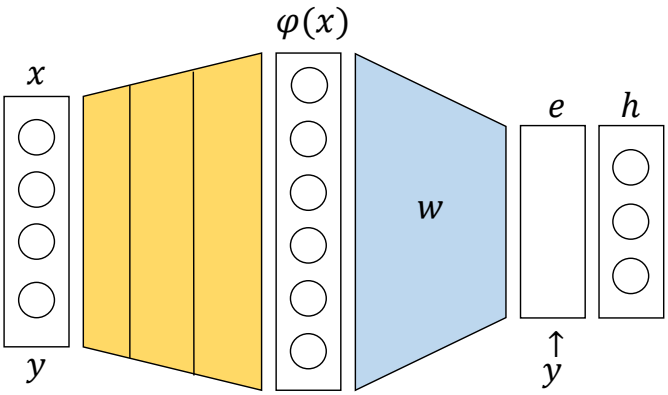


Figure 2: A graphical representation of the RKM model for classification [11]. The feature map $\varphi(\mathbf{x})$ maps the input vector $\mathbf{x}$ to a high dimensional feature space (this mapping is depicted in yellow). The hidden features are obtained through an inner pairing $\mathbf{e}^T\mathbf{h}$ where $\mathbf{e}$ denotes the error on the input $\mathbf{x}$ given by $\mathbf{e} = 1 - (\mathbf{w}^T\varphi(\mathbf{x}) + b)\mathbf{y}$, where $\mathbf{w}$ is depicted in blue.

By characterizing the stationary points of $\mathcal{J}$ and eliminating the unknown weight vector $\mathbf{w}$ the linear problem as stated in [11, Eq.(3.22)] is obtained.

The formulation can easily be extended to the multiclass setting by introducing multiple outputs $\mathbf{y}^{(l)} \in \mathbb{R}^N$ for $l = 1, \ldots, m$. The number of output values m depend on the type of coding used to encode n_c classes. E.g., one could choose the one-versus-all (OVA) encoding where $m = n_c$, which results in binary decisions between each class and all other classes. Another popular encoding is the minimum output encoding (MOC), which is mostly used when the number of classes is very high as it uses m outputs to encode up to $n_c = 2^m$ classes.

Suykens introduced multiclass RKM by formulating one linear system [11, Eq.(3.22)] with all binary subproblems included. Due to the block structure, however, it is equivalent to solving the linear system for the binary class RKM for each output:

$$\left[\begin{array}{c|c} \frac{1}{\eta}\mathbf{\Omega}^{(l)} + \lambda^{(l)}\mathbb{I}_N & \mathbf{1}_N \\ \hline \mathbf{1}_N^T & 0 \end{array}\right] \left[\begin{array}{c} \mathbf{y}^{(l)} \odot \mathbf{h}^{(l)} \\ \hline b^{(l)} \end{array}\right] = \left[\begin{array}{c} \mathbf{y}^{(l)} \\ \hline 0 \end{array}\right] \tag{2}$$

where $^{(l)}$ denotes the l-th output, $\odot$ denotes the element-wise product, $\mathbf{1}_N$ is a one column vector of dimension N and $\mathbb{I}_N$ is the identity matrix of dimension $N \times N$. Note that the hidden features for a certain datapoint

$\mathbf{x}_k$ are comprised of the values $h_k^{(l)}$ for all outputs $l = 1, \ldots, m$. The kernel matrix $\mathbf{\Omega}^{(l)}$ can be determined as follows:

$$\begin{aligned} \Omega_{ij}^{(l)} &= \varphi^{(l)}(\mathbf{x}_i)^T \varphi^{(l)}(\mathbf{x}_j) \\ &= K^{(l)}(\mathbf{x}_i, \mathbf{x}_j), \quad i, j = 1, \ldots, N. \end{aligned} \tag{3}$$

When $\eta = 1$, The solution of Eq. (2) is equivalent to the LS-SVM dual formulation.

2.2. Multi-View LS-SVM Classification

This section summarizes the *Multi-View Least Squares Support Vector Machines (MV-LSSVM) Classification* model as described by Houthuys et al. [7]. This model was proposed as a multi-view extension to LS-SVM classification where a coupling term is introduced which minimizes the product of the error variables of two views. When there are more than two views, the coupling is done in a pairwise manner, i.e. the objective function includes an addition of the coupling between all pairs of views.

Given a training set of N data points $\{(\mathbf{x}_k^{[v]}, y_k^{(l)})\}_{k=1,l=1}^{k=N,l=m}$ for each view $v = 1, \ldots, V$ where $\mathbf{x}_k^{[v]} \in \mathbb{R}^{d^{[v]}}$ denotes the k-th input pattern and $y_k^{(l)} \in \{-1, 1\}$ the l-th output unit corresponding to the k-th input, the primal formulation of the MV-LSSVM model is formulated as:

$$\begin{aligned} \min_{\substack{\mathbf{w}^{[v]^{(l)}}, \\ \mathbf{e}^{[v]^{(l)}}, \\ b^{[v]^{(l)}}}} \; & \frac{1}{2} \sum_{l=1}^{m} \sum_{v=1}^{V} \mathbf{w}^{[v]^{(l)T}} \mathbf{w}^{[v]^{(l)}} + \frac{1}{2} \sum_{l=1}^{m} \sum_{v=1}^{V} \gamma^{[v]^{(l)}} \mathbf{e}^{[v]^{(l)T}} \mathbf{e}^{[v]^{(l)}} + \rho \sum_{l=1}^{m} \sum_{v,u=1; v \neq u}^{V} \mathbf{e}^{[v]^{(l)T}} \mathbf{e}^{[u]^{(l)}} \\ \text{s.t. } & \mathbf{Z}^{[v]^{(l)T}} \mathbf{w}^{[v]^{(l)}} + \mathbf{y}^{(l)} \, b^{[v]^{(l)}} = \mathbf{1}_N - \mathbf{e}^{[v]^{(l)}} \\ & \text{for } v = 1, \ldots, V \text{ and } l = 1, \ldots, m \end{aligned} \tag{4}$$

where $^{(l)}$ denotes the lth output. $\mathbf{e}^{[v]^{(l)}} \in \mathbb{R}^N$ are error variables, $b^{[v]^{(l)}}$ are bias terms and $\mathbf{y}^{(l)} = [y_1^{(l)}; \ldots; y_N^{(l)}]$. The regularization parameters $\gamma^{[v]^{(l)}}$ and the coupling parameter ρ are positive real constants. The feature matrices $\mathbf{Z}^{[v]^{(l)T}} \in \mathbb{R}^{N \times d_h^{[v]^{(l)}}}$ are defined as $\mathbf{Z}^{[v]^{(l)T}} = [y_1^{(l)} \varphi^{[v]^{(l)}}(\mathbf{x}_1^{[v]})^T; \ldots; y_N^{(l)} \varphi^{[v]^{(l)}}(\mathbf{x}_N^{[v]})^T]$ where $\varphi^{[v]^{(l)}} : \mathbb{R}^{d^{[v]}} \to \mathbb{R}^{d_h^{[v]^{(l)}}}$ are the mappings to high (possibly infinite) dimensional feature spaces.

By taking the Lagrangian of the primal problem, deriving the KKT optimality conditions and eliminating the primal variables $\mathbf{w}^{[v]^{(l)}}$ and $\mathbf{e}^{[v]^{(l)}}$, we obtain the dual problem as shown in [7, Eq. (11)].

3. Including Tensor learning in Multi-View Classification

In this section novel multi-view classification algorithms are proposed where the correlation between all views is coupled simultaneously instead of pairwise.

3.1. Multi-View RKM Classification

We first introduce the *Multi-View Restricted Kernel Machine (MV-RKM) Classification* model. This is an extension to RKM classification where data comes from multiple views. The views are coupled by means of shared hidden features.

Given a number of V views, a training set of N data points $\{(\mathbf{x}_k^{[v]}, y_k)\}_{k=1}^{k=N}$ for each view $v = 1, \ldots, V$ where $\mathbf{x}_k^{[v]} \in \mathbb{R}^{d^{[v]}}$ denotes the k-th input pattern and $y_k \in \{-1, 1\}$ the k-th label, we aim at maximizing

$$\sum_{v=1}^{V} \sum_{k=1}^{N} \Big(1 - (\varphi^{[v]}(\mathbf{x}_k^{[v]})^T \mathbf{w}^{[v]} + b) y_k\Big) h_k - \frac{\lambda}{2} \sum_{k=1}^{N} h_k^2 \tag{5}$$

where b is a common bias term, and the hidden features h_k are shared over all views [2]. The full objective including regularization terms of the proposed MV-RKM classification model is:

$$\mathcal{J} = \frac{\eta}{2} \sum_{v=1}^{V} \mathbf{w}^{[v]^T} \mathbf{w}^{[v]} + \sum_{v=1}^{V} \sum_{k=1}^{N} \Big(1 - (\varphi^{[v]}(\mathbf{x}_k^{[v]})^T \mathbf{w}^{[v]} + b) y_k\Big) h_k - \frac{\lambda}{2} \sum_{k=1}^{N} h_k^2 \tag{6}$$

where λ and η are positive real regularization constants. $\varphi^{[v]} : \mathbb{R}^{d^{[v]}} \to \mathbb{R}^{d_h^{[v]}}$ are the view-specific feature maps which map the input of each view to a high dimension. Similarly to RKM, we will not work with this feature map explicitly, but use a positive definite kernel function $K^{[v]} : \mathbb{R}^{d^{[v]}} \times \mathbb{R}^{d^{[v]}} \to \mathbb{R}$. This model is presented in Figure 3a.

The stationary points for this objective function can be found in Appendix A.

[2] This expression (5) is a lower bound on the L_2 loss function on the errors. The full objective (6) is hence a lower bound on the LS-SVM objective function, consisting of a loss function and regularization term, see [11]

By eliminating the weights $\mathbf{w}^{[v]}$, the following linear system is obtained:

$$\left[\begin{array}{c|c} \frac{1}{\eta}\sum_{v=1}^{V}\boldsymbol{\Omega}^{[v]} + \lambda\mathbb{I}_N & \mathbf{V}_N \\ \hline \mathbf{1}_N^T & 0 \end{array}\right]\left[\begin{array}{c} \mathbf{y}\odot\mathbf{h} \\ \hline b \end{array}\right] = \left[\begin{array}{c} V\mathbf{y} \\ \hline 0 \end{array}\right] \tag{7}$$

where $\mathbf{V}_N$ is a column vector of dimension N where each element equals V and $\boldsymbol{\Omega}^{[v]}$ is the (unlabeled) kernel matrix corresponding to view v.

Note that this solution is similar to the single-view RKM problem in Eq. (2), with the main difference being the addition of the view-specific kernel matrices.

3.2. Tensor Multi-View RKM Classification

Even though in the MV-RKM formulation the views are coupled by the shared hidden features, there is still a model weight vector $\mathbf{w}^{[v]} \in \mathbb{R}^{d_h^{[v]}}$ for each view v. Here the *Tensor Multi-View Restricted Kernel Machine (TMV-RKM) Classification* model is presented which introduces a model tensor $\mathcal{W} \in \mathbb{R}^{d_h^{[1]}\times\ldots\times d_h^{[V]}}$ comprised of the weights of all views.

The objective of the TMV-RKM model is given by:

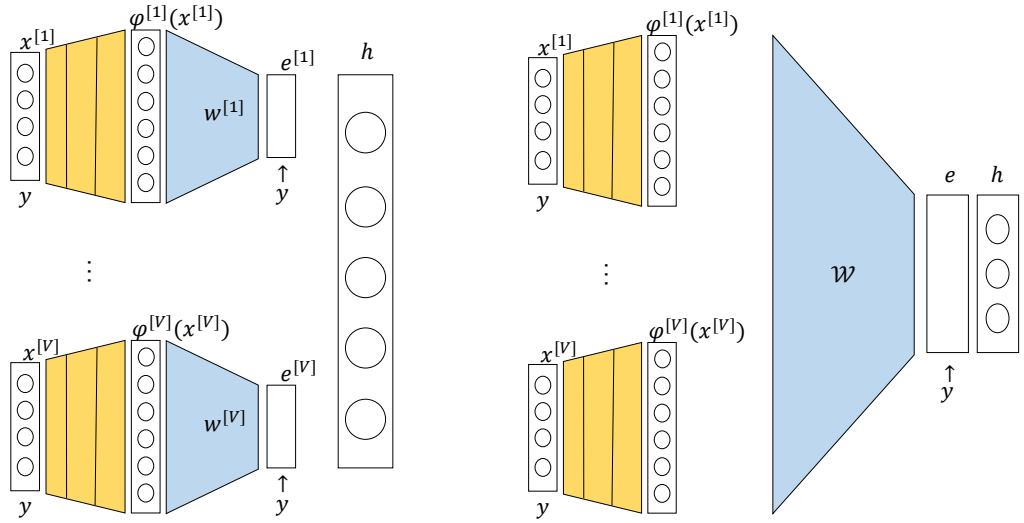
$$\mathcal{J} = \frac{\eta}{2}\langle\mathcal{W},\mathcal{W}\rangle + \sum_{k=1}^{N}\left(1 - (\langle\Phi_{(k)},\mathcal{W}\rangle + b)y_k\right)h_k - \frac{\lambda}{2}\sum_{k=1}^{N}h_k^2 \tag{8}$$

where $\mathcal{W} \in \mathbb{R}^{d_h^{[1]}\times\ldots\times d_h^{[V]}}$ is a Vth order weight tensor and $\Phi_{(k)} \in \mathbb{R}^{d_h^{[1]}\times\ldots\times d_h^{[V]}}$ is a rank-1 tensor composed by the outer product of the view-specific feature maps, i.e. $\Phi_{(k)} = \varphi^{[1]}(x_k^{[1]}) \otimes \ldots \otimes \varphi^{[V]}(x_k^{[V]})$. The notation $\langle\cdot,\cdot\rangle$ denotes the tensor inner product, defined as

$$\langle\mathcal{A},\mathcal{B}\rangle := \sum_{i_1=1}^{I_1}\cdots\sum_{i_M=1}^{I_M}\mathcal{A}_{i_1\cdots i_M}\mathcal{B}_{i_1\cdots i_M} \tag{9}$$

for two M-th order tensors $\mathcal{A},\mathcal{B} \in \mathbb{R}^{I_1\times\ldots\times I_M}$.

The TMV-RKM model can be represented graphically as in Figure 3b. When comparing Figure 3a and Figure 3b it is apparent that TMV-RKM does not only introduce more coupling through the model tensor $\mathcal{W}$, but also that the coupling is done earlier in the input transformation process. In other words, we can hence note that in TMV-RKM the information is fused earlier on than in MV-RKM, while still being able to model the views differently by using different feature maps for each view.



(a) MV-RKM (b) TMV-RKM

Figure 3: A graphical representation of MV-RKM and TMV-RKM for classification for V views. Each feature map $\varphi^{[v]}(\mathbf{x}^{[v]})$ maps the input vector $\mathbf{x}^{[v]}$ to a high dimensional feature space (this mapping is depicted in yellow). For MV-RKM there is a separate error $\mathbf{e}^{[v]} = 1 - (\mathbf{w}^{[v]^T}\varphi^{[v]}(\mathbf{x}^{[v]}) + b)\mathbf{y}$ on each input which is paired with the common hidden features as $\mathbf{e}^{[v]^T}\mathbf{h}$. For TMV-RKM, the outer product of the view-specific features maps make up the feature tensor Φ. The hidden features are shared over all views and are obtained through the inner pairing $\mathbf{e}^T\mathbf{h}$ where $\mathbf{e}$ denotes the error on the input $\mathbf{x}^{[1:V]}$ given by $\mathbf{e} = 1 - (\langle \mathcal{W}, \Phi \rangle + b)\mathbf{y}$, where the interconnection tensor $\mathcal{W}$ is depicted in blue.

The stationary points for this objective function can be found in Appendix B.

By eliminating the weights, the following linear system is obtained:

$$\left[\begin{array}{c|c} \frac{1}{\eta}\bigodot_{v=1}^{V}\mathbf{\Omega}^{[v]}+\lambda\mathbb{I}_N & \mathbf{1}_N \\ \hline \mathbf{1}_N^T & 0 \end{array}\right]\left[\begin{array}{c} \mathbf{y}\odot\mathbf{h} \\ \hline b \end{array}\right]=\left[\begin{array}{c} \mathbf{y} \\ \hline 0 \end{array}\right] \tag{10}$$

where $\bigodot_{v=1}^{V}$ denotes the element-wise multiplication over the indices $v = 1,\ldots,V$.

Note that the obtained linear system results in an element-wise multiplication of the view-specific kernel matrices. This multiplication can have some unwanted results. To illustrate this, take for example two kernel matrices $\mathbf{\Omega}^{[v1]}$ and $\mathbf{\Omega}^{[v2]}$ comprised of linear kernel functions and two elements $\Omega_{ij}^{[v1]}$ and $\Omega_{ij}^{[v2]}$ which are both $\ll 0$. Since the values are very low, this indicates that the similarity between $\mathbf{x}_i^{[v1]}$ and $\mathbf{x}_j^{[v1]}$ (and between $\mathbf{x}_i^{[v2]}$ and $\mathbf{x}_j^{[v2]}$) is very low. However, the element-wise multiplication will result in a highly positive value for the pair of input data, indicating a strong similarity. Even for kernel matrices comprised of Radial Basis Function (RBF) kernel functions, where the values lie between zero and one, a poor view indicating a certain data point pair incorrectly as non-similar and hence assigning a value close to zero, could influence the final result too harshly.

In order to counteract these effects, a third model, called *ϱ-Tensor Multi-View Restricted Kernel Machine (ϱTMV-RKM) Classification*, is proposed which combines the principles of the two previously proposed methods. A parameter $\varrho \in [0,1]$ is added in order to determine the influence of each principle. The objective is formulated as:

$$\begin{aligned}\mathcal{J} =& -\frac{\lambda}{2}\sum_{k=1}^{N}h_k^2+\frac{\eta(1-\varrho)}{2}\sum_{v=1}^{V}\mathbf{w}^{[v]^T}\mathbf{w}^{[v]} \\ &+(1-\varrho)\sum_{v=1}^{V}\sum_{k=1}^{N}\Big(1-(\mathbf{w}^{[v]^T}\varphi^{[v]}(\mathbf{x}_k^{[v]})+b)y_k\Big)h_k \\ &+\frac{\eta\varrho}{2}\langle\mathcal{W},\mathcal{W}\rangle+\varrho\sum_{k=1}^{N}\big(1-\langle\Phi_{(k)},\mathcal{W}\rangle+b)y_k\big)h_k.\end{aligned} \tag{11}$$

By deriving the stationary points of the objective and eliminating the

weights, the following linear problem is obtained:

$$\left[\begin{array}{c|c} \frac{1}{\eta}\left((1-\varrho)\sum_{v=1}^{V}\boldsymbol{\Omega}^{[v]}+\varrho\bigodot_{v=1}^{V}\boldsymbol{\Omega}^{[v]}\right)+\lambda\mathbb{I}_N & \boldsymbol{\tau}_N \\ \hline \mathbf{1}_N^T & 0 \end{array}\right]\left[\begin{array}{c} \mathbf{y}\odot\mathbf{h} \\ \hline b \end{array}\right]=\left[\begin{array}{c} \tau\mathbf{y} \\ \hline 0 \end{array}\right] \tag{12}$$

where $\tau = (1-\varrho)V + \varrho$ and $\boldsymbol{\tau}_N$ is a column vector of dimension N where each element equals τ. The full derivation can be found in Appendix C.

Note that if $\varrho = 0$ this model equals the MV-RKM model, and that if $\varrho = 1$ the model equals the TMV-RKM model. The parameter ϱ could hence be seen as an indicator of the early versus late fusion importance, where a small value indicates a more late, and a large value a more early, type of fusion.

Similarly to RKM, it is possible to formulate the ϱTMV-RKM model for multi-class classification by solving the problem in Eq. (12) for each binary subproblem. Let m be the number of outputs encoding the n_c classes, the multi-class ϱTMV-RKM model is formulated as follows:

$$\begin{aligned}&\left[\begin{array}{c|c} \frac{1}{\eta}\left((1-\varrho^{(l)})\sum_{v=1}^{V}\boldsymbol{\Omega}^{[v]^{(l)}}+\varrho^{(l)}\bigodot_{v=1}^{V}\boldsymbol{\Omega}^{[v]^{(l)}}\right)+\lambda^{(l)}\mathbb{I}_N & \boldsymbol{\tau}_N^{(l)} \\ \hline \mathbf{1}_N^T & 0 \end{array}\right]\left[\begin{array}{c} \mathbf{y}^{(l)}\odot\mathbf{h}^{(l)} \\ \hline b^{(l)} \end{array}\right] \\ &=\left[\begin{array}{c} \tau^{(l)}\mathbf{y}^{(l)} \\ \hline 0 \end{array}\right]\end{aligned} \tag{13}$$

for $l = 1, \ldots, m$. Note that one could define a different influence parameter ϱ for each output. For ease of notations we will omit the $^{(l)}$ superscript when the statement is true for all outputs.

3.3. Decision rule

The model in Eq. (13) will be solved based on the available training data. The extracted variables $\mathbf{h}$ and bias term b are used to construct the classifier $\hat{y}(\mathbf{x_t}^{[1:V]})$ that is able to classify a new unseen test data point $\mathbf{x_t}^{[1:V]}$ where the superscript $^{[1:V]}$ is shorthand for 'for all views $v = 1, \ldots, V$'. Both the primal (P) as the dual (D) representation will be shown.

This classifier can be defined in two ways:

1. Through a combination of kernel functions:

$$(P): \hat{y}(\mathbf{x_t}^{[1:V]}) = \text{sign}\left((1-\varrho)\sum_{v=1}^{V}\mathbf{w}^{[v]^T}\varphi^{[v]}(\mathbf{x_t}^{[v]}) + \varrho\langle\Phi_{(\mathbf{x_t}^{[1:V]})}, \mathcal{W}\rangle + b\right)$$

(14)

$$(D): \hat{y}(\mathbf{x_t}^{[1:V]}) = \text{sign}\left(\frac{1}{\eta}\sum_{k=1}^{N} y_k h_k \left[(1-\varrho)\sum_{v=1}^{V} K^{[v]}(\mathbf{x_t}^{[v]}, \mathbf{x}_k^{[v]}) + \varrho \prod_{v=1}^{V} K^{[v]}(\mathbf{x_t}^{[v]}, \mathbf{x}_k^{[v]})\right] + b\right) \tag{15}$$

where $\Phi_{(\mathbf{x_t}^{[1:V]})} = \varphi^{[1]}(\mathbf{x_t}^{[1]}) \otimes \ldots \otimes \varphi^{[V]}(\mathbf{x_t}^{[V]})$. Notice that ϱ plays a similar role here in the classifier as in the ϱTMV-RKM training phase.

2. Similarly to the classifier in pairwise MV-LSSVM:

$$(P): \hat{y}(\mathbf{x_t}^{[1:V]}) = \text{sign}\left(\sum_{v=1}^{V} \beta^{[v]} \mathbf{w}^{[v]^T} \varphi^{[v]}(\mathbf{x_t}^{[v]}) + b\right) \tag{16}$$

$$(D): \hat{y}(\mathbf{x_t}^{[1:V]}) = \text{sign}\left(\frac{1}{\eta}\sum_{v=1}^{V} \beta^{[v]} \sum_{k=1}^{N} y_k h_k K^{[v]}(\mathbf{x_t}^{[v]}, \mathbf{x}_k^{[v]}) + b\right) \tag{17}$$

where usually $\sum_{v=1}^{V} \beta^{[v]} = 1$. Note that while the decision rule in itself does not include the higher-order tensor terms, the model is still trained with these terms included. It can be noted that, while there are multiple ways to determine the values for $\beta^{[v]}$, taking the mean, and hence taking $\beta^{[1]} = \ldots = \beta^{[V]} = 1/V$, produces overall good results. Therefore we will use this throughout the rest of the paper. If prior knowledge is available about the usefulness of certain views regarding the decision rule, these weights could be altered to obtain a weighted average.

3.4. Model Selection

The number of tuning parameters increases with the number of classes and views. Therefore, to decrease the tuning complexity, the same regularization parameters and kernel function (including parameters) for each output is chosen. Thus $\lambda = \lambda^{(1)} = \ldots = \lambda^{(m)}$, $\varrho = \varrho^{(1)} = \ldots = \varrho^{(m)}$ and $K^{[v]}(\cdot,\cdot) = K^{[v]^{(1)}}(\cdot,\cdot) = \ldots = K^{[v]^{(m)}}(\cdot,\cdot)$. Notice that different views can still have different corresponding kernel functions. Furthermore, the parameter η is set to 1, since in RKM, the influence of this parameter is of most importance when multiple RKMs are stacked to form a deep RKM. Hence, in total there are two hyperparameters to tune, in addition to potential kernel

parameters. As we will later show, the proposed method is relatively fast in comparison to other multi-view methods, so the tuning overhead should not pose a problem. However, if needed one could decrease the tuning complexity even further by assuming the same kernel function over all views, or choosing parameter-free kernel functions (like e.g. a linear kernel). The resulting algorithm is described in Algorithm 1, where $\theta^{[1:V]}$ denotes the kernel parameters (if any) and 'decision_rule' indicates whether the classifier is defined by Eq. (17) (decision_rule=mean) or by Eq. (15) (decision_rule=add). The superscript $^{(1:m)}$ is shorthand for 'for all binary subproblems $l = 1, \ldots, m$'.

Algorithm 1 ϱTMV-RKM training and prediction

Input: $\mathcal{X}^{[1:V]} = \{y_k^{(l)}, \mathbf{x}_k^{[1:V]}\}_{k=1,l=1}^{k=N,l=m}, K^{[1:V]}, \theta^{[1:V]}, \lambda, \quad \varrho, \mathcal{X}_t^{[1:V]} = \{\mathbf{x_t}_k^{[1:V]}\}_{k=1}^{k=N_t}$, decision_rule

for $l = 1$ **to** m **do**
  for $v = 1$ **to** V **do**
    $\mathbf{\Omega}^{[v]} \leftarrow$ Eq.(3)$(\mathcal{X}^{[v]}, K^{[v]}, \theta^{[v]})$
  end for
  $b^{(l)}, \mathbf{h}^{(l)} \leftarrow$ Eq.(13)$(\mathbf{\Omega}^{[1:V]}, \lambda, \varrho, \mathbf{y}^{(l)})$
  if decision_rule == add **then**
    $\hat{y}^{(l)}(\mathbf{x_t}^{[1:V]}) \leftarrow$ Eq.(15)$(\mathbf{h}^{(l)}, \mathbf{y}^{(l)}, b^{(l)}, \varrho, K^{[1:V]}, \theta^{[1:V]}, \mathcal{X}_t^{[1:V]})$
  else if decision_rule == mean **then**
    $\hat{y}^{(l)}(\mathbf{x_t}^{[1:V]}) \leftarrow$ Eq.(17)$(\mathbf{h}^{(l)}, \mathbf{y}^{(l)}, b^{(l)}, K^{[1:V]}, \theta^{[1:V]}, \mathcal{X}_t^{[1:V]})$
  end if
end for

Output: $\hat{y}^{(1:m)}(\mathbf{x_t}^{[1:V]})$

The optimal parameters are found through Simulated Annealing and 5-fold cross validation using only the training set. The model is then evaluated using an independent test set. This model selection process is described in Algorithm 2.

4. Experiments

In this section the results of ϱTMV-RKM are shown and compared to other state-of-the-art multi-view classification methods. Note that since MV-RKM and TMV-RKM are special cases of ϱTMV-RKM (with $\varrho = 0$ and

Algorithm 2 Model selection

Input: for $\mathcal{X}^{[1:V]} = \{y_k^{(l)}, \mathbf{x}_k^{[1:V]}\}_{k=1,l=1}^{k=N,l=m}$, $K^{[1:V]}, \mathcal{X}_t^{[1:V]} = \{\mathbf{x_t}_k^{[1:V]}\}_{k=1}^{k=N_t}$, decision_rule

1: $\theta^{[1:V]}, \lambda, \varrho \leftarrow$ Simulated Annealing & 5-fold crossvalidation (Algorithm 1, $\mathcal{X}^{[1:V]}, K^{[1:V]}$, decision_rule) with criteria: classification accuracy
2: $\hat{y}^{(1:m)}(\mathbf{x_t}^{[1:V]}) \leftarrow$ Algorithm 1 $(\mathcal{X}^{[1:V]}, K^{[1:V]}, \theta^{[1:V]}, \lambda, \varrho, \mathcal{X}_t^{[1:V]}$,decision_rule)

Output: $\hat{y}^{(1:m)}(\mathbf{x_t}^{[1:V]})$

$\varrho = 1$ respectively), we implicitly also compare with these methods. First, the accuracy and runtime for several real-world datasets is discussed. Next, a parameter study is performed showing the stability of the trade-off parameter ϱ. Furthermore, the statistical significance of the results is demonstrated using the Wilcoxon signed-rank test and a confidence interval on the test accuracy. The section ends with a discussion of approaches to handle large-scale datasets.

4.1. Datasets

A brief description of the real-world datasets used is given. The important statistics of them are summarized in Table 1.

- **Ads dataset:** The Ads dataset[3], as described by Kushmerick [39], consists of hyperlinks which are labeled as an advertisement or not an advertisement. The features are divided over three views in the same way as was done by Luo et al. [9], where the first view describes the images, the second view describes the URL of the website and the last view describes the anchor URL. The dataset consists of 458 advertisements, and 2821 non-advertisements

- **Flower species dataset:** This dataset describes the classification of 17 flower species and was originally proposed by Nilsback & Zisserman [40, 41]. The data comes from 1360 images segmented from the

[3]Available at `http://archive.ics.uci.edu/ml/datasets/Internet+Advertisements`

background[4]. Similarly to the work of Minh et al., [8] seven features are extracted and used as views: HOG, HSV histogram, boundary SIFT, foreground SIFT, and three features derived from color, shape and texture vocabularies.

- **Image-caption web dataset:** Kolenda et al. [42] collected this dataset by retrieving sport, aviation and paintball images and their associated captions. We thank the authors of [42] for providing the dataset. The data is described through three views, where the first two views represent extracted features of the images (HSV color and image Gabor texture) and the third view consists of the term frequencies of their associated caption text[5].

- **YouTube Video dataset:** This dataset, describing YouTube videos of video games, was originally proposed by Madani et al. [43]. The videos are represented by three high-level feature families: textual, visual and auditory features[6]. For this paper we selected three features to be the different views, namely the textual feature LDA, the visual Motion feature through CIPD [44] and the audio feature MFCC [45]. From each of the seven most occurring labels (excluding the last label, since these data points represent videos not belonging to any of the other 30 classes) 300 videos were randomly sampled.

- **Digits dataset:** This dataset, originally proposed by van Breukelen et al. [46], represents handwritten digits (0-9) and is taken from the UCI repository [47][7]. The dataset consists of 2000 digits which are represented through six views describing the Fourier coefficients, profile correlations, Karhunen-Loève coefficients, pixel averages, Zernik moments and morphological features.

- **NUS-WIDE dataset:** The large NUS-WIDE dataset[8] was proposed by Chua et al. [48] and consists of images collected from Flickr. We

[4]Available at `http://www.robots.ox.ac.uk/~vgg/data/flowers/17/index.html` .

[5]Detailed description of these features can be found in Kolenda et al. [42].

[6]Available at `http://archive.ics.uci.edu/ml/datasets/youtube+multiview+video+games+dataset` .

[7]Available at `https://archive.ics.uci.edu/ml/datasets/Multiple+Features` .

[8]Available at `https://lms.comp.nus.edu.sg/research/NUS-WIDE.htm`

Table 1: Details of the datasets used in the experiments. N and N_t denote respectively the number of data points in the training and test set. V denotes the number of views and n_c the number of classes of the dataset.

Dataset	N	N_t	V	n_c	**Encoding**
Ads	2623	656	3	2	-
Flower species	1088	272	7	17	MOC
Image-caption web	960	240	3	3	OVA
YouTube Video Games	1680	420	3	7	MOC
Digits	1600	400	6	10	MOC
NUS-WIDE	4776	1194	5	10	MOC

use a subset that consists of 5970 images, each belonging to one of the ten nature-themed classes: sunset, water, beach, sky, clouds, snow, lake, street, ocean and tree. The images are described through five views: Color Histogram, Local Self-Similarity, Pyramid HOG, SIFT, Color SIFT and SURF features.

For the Flower species dataset, data is already provided as kernels. For the Ads dataset, the Digits dataset, the NUS-WIDE dataset and the first two views of the Image-caption web dataset, the radial basis function (RBF) kernel is chosen. For the YouTube Video dataset the dimension of the features range between 512 and 2000 and the dimension of the third view of the Image-caption web dataset is 3522. Because these features are so high dimensional (and sparse), using an RBF kernel, and hence bringing the data to a even higher feature space, is not recommended. Therefore a linear kernel is chosen for these views. Since this simple kernel function resulted in a good performance other appropriate kernel functions for text-data such as polynomial kernels, Chi-square kernels [49] or String kernels [50] were not considered.

The data is randomly divided into a test and training set three times where 80% of the data belongs to the training set. The results shown are averaged over the three splits.

4.2. Baseline Algorithms

The performances of the proposed method ϱTMV-RKM on the different datasets are compared to two single-view methods, two typical early and late fusion techniques and seven state-of-the-art multi-view methods:

- **Best Single View (BSV):** The results of applying RKM classification on the most informative view, i.e., the one which results in the best performance.
- **BSV**$_{b=0}$: RKM with bias $b = 0$ on the most informative view.
- **Feature Concatenation (FC):** A typical early fusion method where the features of all views are concatenated. RKM is used to do classification on this concatenated view representation.
- **Committee RKM (Comm):** A typical example of late fusion where a separate model is trained for each view and a weighted average is taken as the final classifier [3]. For this baseline method, RKM is applied on each view separately and the weights are calculated based on the training error covariance matrix in the same way as for Committee LS-SVM regression [36].
- **Multi-View LS-SVM classification (MV-LSSVM):** The pairwise multi-view classification method described in Section 2.2. In analogy with the experiments in [7] (and with the experiments done with ϱTMV-RKM), the same regularization parameter and kernel parameter are chosen for each binary subproblem, yet they can differ over the views. In addition to these parameters, also the coupling parameter is tuned.
- **Multi-View Learning with Least Square loss function (MVL-LS):** This method, proposed by Minh et al. [8], is a pairwise multi-view classification model based on SVM that can handle labeled as well as unlabeled data. To fairly compare with our proposed multi-view method we use the same (labeled) data and do not add unlabeled data. The method has three regularization parameters as well as kernel parameters to be tuned.
- **Multi-View Fisher Discriminant Analysis (MFDA):** This multi-view extension to Fisher Discriminant Analysis was proposed by Diethe

et al. [51]. The method aims to minimize the variance of the data along the projections and to maximize the distance between the average outputs of each class, over all views. The parameters to be tuned are a regularization parameter and kernel parameters.

- **SimpleMKL:** This multiple kernel learning method, based on SVM, is proposed by Rakotomamonjy et al. [52]. The kernel is defined as a linear combination of multiple kernels. The SimpleMKL problem is defined through a weighted 2-norm regularization formulation with a constraint on the weights that encourages sparse kernel combinations. It has one regularization parameter as well as kernel parameters to be tuned.

- **EasyMKL:** A popular multiple kernel learning method proposed by Aiolli & Donini [34]. EasyMKL is often used due to its scalability w.r.t. the number of kernels. The resulting kernel is defined as a weighted sum of multiple kernels, with positive weights. It has one regularization parameter ($\gamma \in [0, 1]$) as well as kernel parameters to be tuned.

- **Multilinear Factorization Machine (MFM):** This tensor-based method proposed by Lu et al. [22] is described as a multi-view multi-task method. However it is also used in the paper for classification, where the different tasks correspond to the different binary classification problems related to multi-class classification. The method models the feature interactions among the different views and tasks, as a tensor structure by taking the tensor product of their respective feature spaces. In our experiments we use the three variations as stated in [22] and reported the best result, moreover the fixed parameters are set in the same way. The regularization parameters remain to be tuned.

- **t-SVD based Multi-view Subspace Clustering (t-SVD-MSC):** This multi-view clustering method, proposed by Xie et al. [21], uses tensor learning by stacking the subspace representation matrices of the different views to form a tensor and subsequently rotating it such that the higher order correlations between the views are explored. Despite it being a clustering method, the method achieves a good performance and is therefore included in the experiments. It has one regularization parameter to be tuned, which will be tuned based on the labels (hence in a supervised manner).

Note that MV-LSSVM and MVL-LS handle multiple views in a pairwise fashion, while MFDA, SimpleMKL, EasyMKL, MFM and t-SVD-MSC consider the information from all views simultaneously. The parameters of these baseline algorithms are selected in the same way as for ϱTMV-RKM (see Algorithm 2). Since the data from the Flower Species dataset is provided as kernel matrices, the non-kernel methods MFM and t-SVD-MSC are not applied on it.

4.3. Experimental Results

Table 2 shows the accuracy of all baseline algorithms and of the proposed ϱTMV-RKM model on the real-world datasets. The BSV results were obtained with the anchor URL terms for the Ads dataset, with the foreground SIFT features for the Flowers species dataset, with the term frequencies of the caption for the Image-caption dataset, with the LDA text feature for the YouTube Video dataset, with the Fourier coefficients for the UCI Digits dataset and with the color histogram features for the NUS-WIDE dataset. ϱTMV-RKM$_{\text{add}}$ and ϱTMV-RKM$_{\text{mean}}$ represent the proposed method with the decision function in Eq. (15) and Eq. (17) respectively.

A first observation is that the proposed multi-view method has a higher accuracy than BSV on all datasets examined. This indicates the improvement of using multiple views. It further improves on the simple coupling schemes FC and Comm on all datasets examined. The other multi-view methods also improve on these simple schemes in most of the cases, indicating that these simple coupling schemes are not sufficient.

Furthermore the table shows that the proposed method ϱTMV-RKM is able to outperform the two pairwise multi-view methods on all studied datasets. Especially on the Flowers, Digits and NUS-WIDE datasets the improvement of including higher order correlations is significantly. Notice that MVL-LS is inherently a semi-supervised algorithm and is hence not optimized to handle a large amount of labeled data, as is especially the case for the NUS-WIDE dataset. So it is not surprising this resulted in an out-of-memory error. In addition, ϱTMV-RKM is certainly competitive with the last five baseline methods. In most experiments ϱTMV-RKM is able to achieve a higher accuracy. Only for the YouTube Video dataset en the NUS-WIDE dataset only SimpleMKL and MFM, respectively, are able to outperform ϱTMV-RKM. Since the NUS-WIDE dataset consist of a high number of classes, it is not surprising MFM outperforms all other methods, as MFM is designed to also take into account the higher and lower order

Table 2: Mean classification accuracy on the test set of the three splits for the real-world datasets. The standard deviation is shown between parentheses. "OM" is short for "out-of-memory error" while running the experiments. Since the Flowers dataset is provided as kernel matrices, only the kernel-based methods can be applied. For the proposed ϱTMV-RKM method, also the mean optimal ϱ over all splits is shown. The highest accuracies are indicated in bold.

Method	Ads	Flowers	Image-caption	YT Video	Digits	NUS-WIDE
BSV	95.73 (±0.46)	44.31 (±1.96)	96.81 (±0.24)	91.03 (±1.37)	74.25 (±10.03)	29.31 (±1.16)
$\text{BSV}_{b=0}$	95.83 (±0.93)	28.82 (±2.65)	98.89 (±0.24)	91.51 (±3.24)	78.67 (±0.88)	29.82 (±1.77)
FC	97.05 (±0.38)	5.49 (±1.39)	79.17 (±1.50)	91.03 (±2.16)	9.25 (±2.18)	31.75 (±0.91)
Comm	85.26 (±1.04)	30.98 (±14.3)	32.50 (±3.31)	66.67 (±12.47)	12.25 (±21.22)	18.79 (±7.99)
MV-LSSVM	97.20 (±1.56)	49.91 (±3.40)	98.06 (±1.97)	90.71 (±5.32)	74.42 (±15.46)	29.98 (±2.96)
MVL-LS	88.50 (±1.37)	8.43 (±4.42)	97.50 (±1.82)	89.21 (±5.41)	89.10 (±1.52)	OM
MFDA	96.95 (±0.79)	62.25 (±3.38)	99.44 (±0.48)	92.78 (±1.92)	79.83 (±8.61)	OM
SimpleMKL	85.26 (±1.04)	10.88 (±5.88)	97.64 (±1.27)	**95.24** (±1.04)	94.67 (±1.91)	32.48 (±1.92)
EasyMKL	95.93 (±0.92)	11.05 (±0.45)	89.44 (±3.37)	91.74 (±2.16)	85.42 (±8.29)	16.61 (±0.46)
MFM	94.00 (±1.03)	–	98.84 (±0.16)	91.71 (±6.00)	94.09 (±2.69)	**60.16** (±16.74)
t-SVD-MSC	85.26 (±1.04)	–	82.08 (±1.25)	62.38 (±1.86)	87.50 (±3.91)	27.67 (±0.78)
ϱTMV-RKM_{add}	**97.66** (±0.88) ($\varrho = 0.83$)	56.37 (±1.67) ($\varrho = 0.00$)	**99.72** (±0.24) ($\varrho = 0.27$)	91.98 (±1.82) ($\varrho = 0.8$)	**94.92** (±1.04) ($\varrho = 0.40$)	34.73 (±0.63) ($\varrho = 0.97$)
ϱTMV-RKM_{mean}	97.61 (±0.58) ($\varrho = 0.20$)	**66.67** (±1.45) ($\varrho = 0.00$)	**99.72** (±0.24) ($\varrho = 0.13$)	92.86 (±2.30) ($\varrho = 0.47$)	84.83 (±11.27) ($\varrho = 0.67$)	33.81 (±1.06) ($\varrho = 0.20$)

correlations between the different binary classification problems. Whereas ϱTMV-RKM considers each classification problem separately. However, as is shown later in this section, most methods have a much higher time cost than the proposed ϱTMV-RKM.

The table further notes the mean optimal ϱ over all splits. We can see that for the Flower dataset, ϱ equals zero. Another observation for this dataset is that the FC method performs quite badly, while Comm is able to achieve an acceptable performance. Both observations indicate a strong need for late fusion, and hence more freedom to model the views differently, for this particular dataset. A stability study of this parameter is performed in the next section.

4.4. Parameter Study

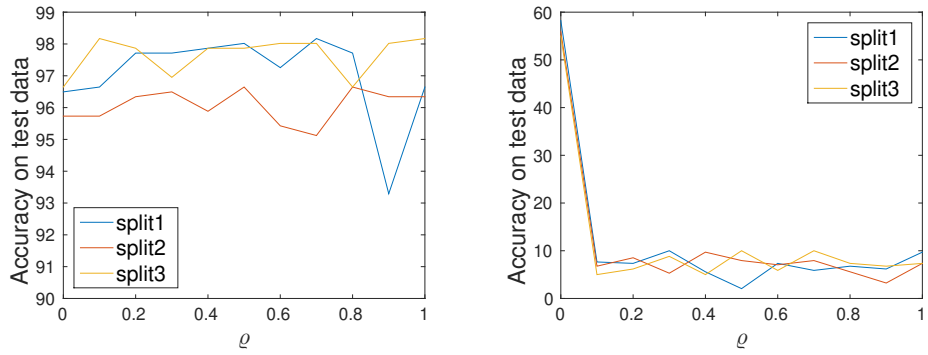
Another set of experiments were performed where for each value of $\varrho \in [0, 0.1, 0.2, \dots 1]$ the regularization and kernel parameters are tuned through simulated annealing. The resulting test accuracy of ϱTMV-RKM on the different datasets can be found in Figure 4.

A first observation is that the accuracy on the test set for a certain value of ϱ is fairly stable over the three data splits, the accuracy rarely differs more then $2 - 3\%$ in accuracy.

While Figure 4a shows that the value ϱ does not influence the performance on the Ads dataset considerably, the other other figures show that finding an appropriate value of ϱ is crucial for the performance. For the Image-caption and the YouTube Video dataset the results are fairly stable expect for the extreme value $\varrho = 1$. For the Flowers and the Digits dataset, however, the graphs indicate that lower values of ϱ are better than high values, where for the Flowers dataset the performance even drops drastically when $\varrho > 0$. This suggest that for these dataset late fusion is more appropriate, which is also supported by the accuracy results in Table 2, as the late fusion method Comm achieves a higher accuracy than the early fusion method FC. Figure 4f for the NUS-WIDE dataset, however, shows an opposite trend, indicating the need for early fusion. Again this is in line with the results of FC and Comm on this dataset, where FC performs better than Comm.

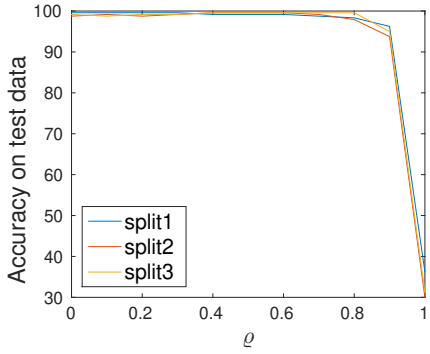
4.5. Statistical significance of the results

In order to argue the significance of the results, the Wilcoxon signed-rank test is used. Since there is no reason to assume that the performances across

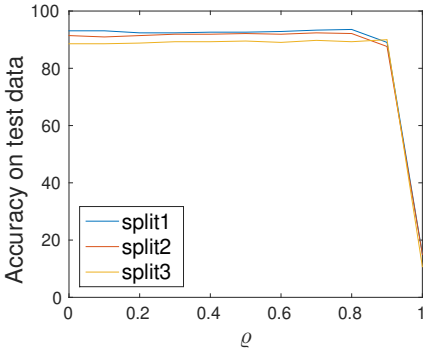


(a) Ads

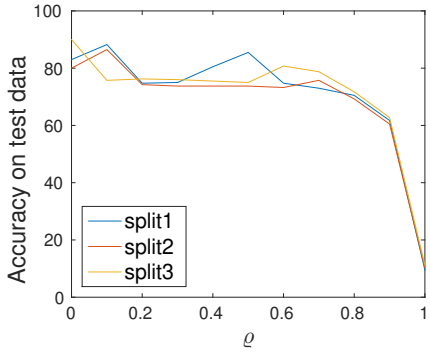
(b) Flowers



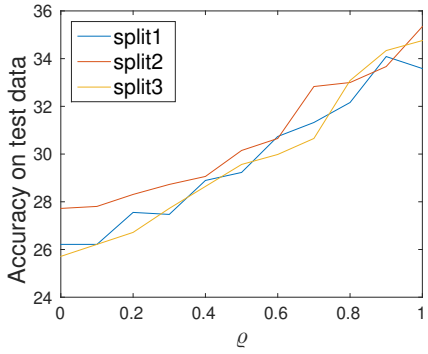
(c) Image-caption



(d) YT Video



(e) Digits



(f) NUS-WIDE

Figure 4: Test accuracy of ϱTMV-RKM applied on the three training-test data splits, with respect to ϱ, on the different datasets.

Table 3: Results of the Wilcoxon signed-rank test for the comparison of ϱTMV-RKM with each baseline algorithm. If $T < T_{0.2}$ we can reject the null-hypothesis that both methods perform equally well with confidence level 0.8.

ϱTMV-RKM	BSV	BSV$_{b=0}$	FC	Comm	MV-LSSVM	MVL-LS
T	0	0	0	0	0	0
$T_{0.2}$	3	3	3	3	3	2

ϱTMV-RKM	MFDA	SimpleMKL	EasyMKL	MFM	t-SVD-MSC
T	0	4	0	5	0
$T_{0.2}$	2	3	3	2	2

data sets are distributed normally, Demšar [53] recommends the Wilcoxon test over other alternatives (e.g. the t-test or sign test).

For each baseline algorithm the difference in performance w.r.t. ϱTMV-RKM is calculated for each dataset, and ranked according to the absolute value of the difference. Let R_1 be the sum of the ranks for the datasets on which ϱTMV-RKM outperforms the baseline, and R_2 the sum of the ranks on which the baseline outperforms ϱTMV-RKM. We can then define T as $T = \min(R_1, R_2)$. Table 3 shows the value T for each baseline method.

Define the null-hypothesis as the hypothesis that ϱTMV-RKM and the baseline algorithm perform equally well. We can then use the table of exact critical values for the Wilcoxon's test to find the critical value T_α for a confidence level $(1 - \alpha)$, and reject the null-hypothesis if $T < T_\alpha$. Note that T_α only depends on α and the number of datasets.

The critical values for $\alpha = 0.2$ are stated in Table 3. Note that since the number of datasets is relatively small (5 or 6), we can not choose a higher confidence level. We can see that for most baseline algorithms we can reject the null-hypotheses, hence we can state with 80% certainty that ϱTMV-RKM outperforms these baseline methods significantly.

We can furthermore determine the confidence intervals on the classification accuracy obtained on the test set. If we assume that the test samples are independent and identically distributed we can use the Wilson score interval [54]. Table 4 shows the 95% confidence intervals c_{95} for the test accuracy of ϱTMV-RKM (see Table 2) on each dataset.

Even though we chose a high confidence level (0.95), most confidence intervals are rather small. The largest interval is found for the Flowers dataset, but even if the true accuracy equalled the lower bound (61.07) ϱTMV-RKM

Table 4: The 95% confidence interval (c_{95}) for the test accuracy of ϱTMV-RKM.

	Ads	Flowers	Image-caption
c_{95}	$[96.50, 98.82]$	$[61.07, 72.27]$	$[99.05, 100]$

	YT Video	Digits	NUS-WIDE
c_{95}	$[90.40, 95.32]$	$[92.78, 97.07]$	$[32.03, 37.43]$

would still outperform almost all baseline algorithms.

4.6. Time complexity and large-scale experiment

Another advantage of the proposed method becomes evident when looking at the time complexity of the model. Table 5 shows the runtime of the different methods on all considered datasets. For these timing results all experiments were run in Matlab (R2014b) on an Ubuntu 16.04 LTS system with a 12-core Intel i7 (2.2GHz) CPU and 16.0 GB RAM.

Unsurprisingly it shows that multi-view learning takes more time than learning from only one view. Also, since RKM is a kernel based model, a higher number of features will not significantly increase the complexity. This is supported by the runtime results of FC, which are similar to the BSV results.

Another, more interesting, observation is that ϱTMV-RKM is much faster than the other seven multi-view methods, especially when compared to the runtime of MFDA, SimpleMKL, MFM and t-SVD-MSC. The improvement in runtime with regard to MV-LSSVM can be explained by looking at the time complexity. For both methods the time complexity of the training phase is heavily dominated by the time it takes to calculate the kernel matrices and by solving a linear problem ([7, Eq. (11)]) for MV-LSSVM and Eq. (13) for ϱTMV-RKM). The authors in [7] showed that for MV-LSSVM these steps have a time complexity of $O(VN^2\,\bar{d})$ and $O(mN^3V^3)$ respectively, where $\bar{d}$ is the mean of the data dimensions over all views. For ϱTMV-RKM the number of operations to calculate the kernel matrices is the same as for MV-LSSVM plus VN^2 additions and multiplications, which leads to the same big order time complexity $O(VN^2\,\bar{d})$. The left-hand size matrix in the linear problem of ϱTMV-RKM, however, is much smaller than in the dual problem of MV-LSSVM. The dimensions are $((N+1)V \times (N+1)V)$ and $(N+1 \times N+1)$ respectively. Since this needs to be calculated for all m outputs, it entails

Table 5: Mean runtime (in seconds) of training and test procedures for the real-world datasets on 15 runs. The standard deviation is shown between parentheses.

Method	Ads	Flowers	Image-caption	YT Video	Digits	NUS-WIDE
BSV	0.72 (±0.39)	0.13 (±0.01)	0.17 (±0.09)	0.33 (±0.13)	0.43 (±0.05)	4.71 (±0.07)
$\text{BSV}_{b=0}$	0.30 (±0.09)	0.13 (±0.05)	0.17 (±0.01)	0.22 (±0.01)	0.40 (±0.01)	5.10 (±0.27)
FC	0.84 (±0.14)	0.15 (±0.01)	0.19 (±0.03)	0.52 (±0.13)	0.43 (±0.07)	5.72 (±0.25)
Comm	2.11 (±0.17)	1.43 (±0.06)	0.48 (±0.13)	1.20 (±0.21)	3.01 (±1.61)	26.72 (±0.22)
MV-LSSVM	9.37 (±1.82)	13.30 (±0.57)	1.34 (±0.14)	4.03 (±1.57)	42.35 (±21.71)	367.89 (±2.81)
MVL-LS	16.24 (±7.96)	3.84 (±0.65)	0.94 (±0.16)	1.99 (±0.06)	29.31 (±63.57)	–
MFDA	$1.58 \cdot 10^3$ (±85.33)	$0.69 \cdot 10^3$ (±58.26)	33.89 (±2.99)	970.24 (±19.61)	225.68 (±2.16)	–
SimpleMKL	$3.05 \cdot 10^3$ (±4.99e+03)	$0.13 \cdot 10^3$ (±0.17)	4.92 (±0.13)	11.45 (±0.12)	182.95 (±90.37)	$1.73 \cdot 10^4$ (±688.30)
EasyMKL	17.88 (±7.40)	$0.16 \cdot 10^3$ (±1.12)	4.21 (±0.57)	15.04 (±0.98)	11.00 (±1.14)	124.25 (±3.69)
MFM	617.49 (±22.79)	–	44.02 (±1.53)	135.53 (±10.12)	673.56 (±88.09)	$6.15 \cdot 10^3$ (±7.97)
t-SVD-MSC	$1.01 \cdot 10^3$ (±14.00)	–	122.97 (±0.32)	375.18 (±6.64)	$1.79 \cdot 10^3$ (±4.45)	$6.31 \cdot 10^3$ (±4.08)
$\varrho\text{TMV-RKM}_{\text{add}}$	1.52 (±0.21)	0.46 (±0.02)	0.30 (±0.03)	0.67 (±1.20)	1.38 (±0.07)	14.68 (±1.03)
$\varrho\text{TMV-RKM}_{\text{mean}}$	1.64 (±0.24)	0.51 (±0.01)	0.30 (±0.01)	0.72 (±0.22)	1.44 (±0.10)	15.24 (±0.15)

Table 6: Time complexity of the training and test phase of MV-LSSVM method and ϱTMV-RKM method.

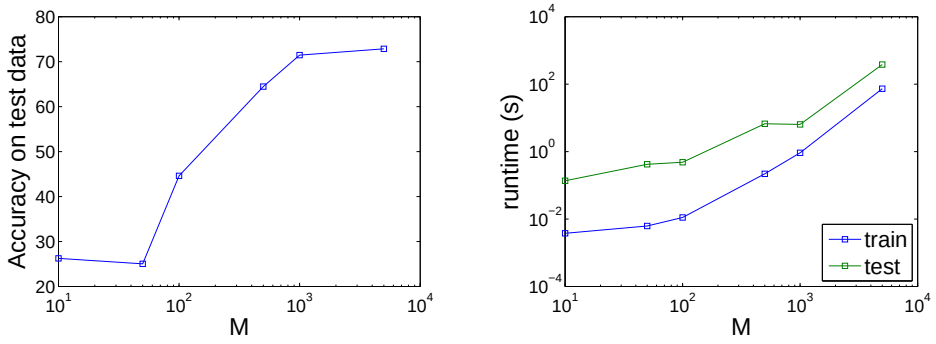
	MV-LSSVM	ϱTMV-RKM
Calculate kernel matrices	$O(VN^2\,\bar{d})$	$O(VN^2\,\bar{d})$
Solving linear system	$O(mN^3V^3)$	$O(mN^3)$
Total training	$O(mN^3V^3)$	$O(mN^3)$
Total test	$O(VNN_t\,\bar{d})$	$O(VNN_t\,\bar{d})$

a time complexity of $O(mN^3)$ for calculating the linear system of ϱTMV-RKM, which is smaller than the time complexity of MV-LSSVM with a factor of V^3. The dimensions of the datasets are usually either small or the features are very sparse. Since most numeric programming languages have fast routines to multiply sparse matrices (like e.g. Matlab), one can usually assume that the training time is mostly dominated by the second step. This training complexity is summarized in Table 6. The test phase consists of computing the classifiers (Eq. (17) or Eq. (15)) for all N_t test points. The time complexity of the test phase is also given in Table 6.

To investigate the behavior of ϱTMV-RKM when dealing with large-scale data, we use the Reuters dataset [47][9]. This dataset, described by Amini et al. [55], consists of documents originally written in five different languages and their translations in each of the other four languages. All documents belong to one of the six categories and are described by a bag-of-words style feature. We took the largest possible Reuters set, which contains 29953 documents and consists of documents written in German and translations of them in English, French, Spanish and Italian. Hence, considering that the data is split up for training and testing in the same way as in the previous section, for this large-scale dataset it holds that $V = 5$, $n_c = 6$, $N = 23962$ and $N_t = 5991$. Furthermore, the dimension of the data over the views range from 11547 to 34279 and the MOC encoding is used.

It is clear from the time complexity given in Table 6 that the training part of ϱTMV-RKM will be very time consuming when N is this large. We will present here two approaches for dealing with large-scale data.

[9]Available at `https://archive.ics.uci.edu/ml/datasets/Reuters+RCV1+RCV2+Multilingual,+Multiview+Text+Categorization+Test+collection` .



(a) Classification accuracy

(b) Runtime (s)

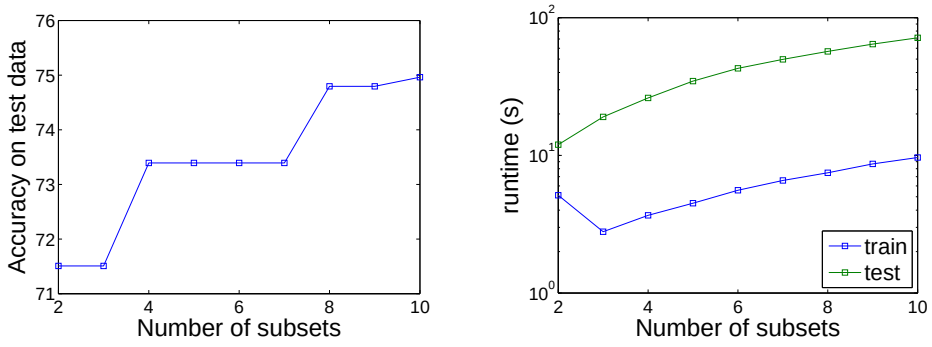
Figure 5: Accuracy and timing results with the first large-scale approach on the Reuters dataset with respect to M.

The first approach is simply taking only a small part of the data to train the model and assume that it will generalize well to the unseen test data. Similar to the work in [7, 56] we will randomly pick M points, with $M \ll N$, from the total training set. However note that the selection of this subset could also be based on certain criteria like e.g. a Rényi entropy based criteria [57] as was used by Mehrkanoon & Suykens or [58] for semi-supervised learning, or based on the angular similarity between the projected values of KSC [59] for community detection as used by Mall et al. [60].

A first experiment investigates the effect of the chosen subset size M of the first approach on the Reuters dataset. Figure 5 shows the accuracy on the test set and the runtime of the training and test phase for $M \in \{0.5{\cdot}10^2, 10^2, 0.5{\cdot}10^3, 10^3, 0.5{\cdot}10^4, 10^4\}$. Linear kernel functions are chosen for all views and the decision function given by Eq. (17) is used.

Figure 5a shows that the accuracy on the test set increases as the subset size increases. This is of course to be expected since more information is used in the training phase. It can also be noted that the accuracy increases drastically when $M \leq 500$ but only slightly afterwards. Which indicates that the model is able to generalize well with a relatively small training set. Figure 5b shows that the training time increases faster with regard to M than does the test time, which is in line with the time complexity given in Table 6.

A second approach is based on the concept of committee networks. A number of smaller subsets are randomly sampled out of the total training set, and the final classifier is defined as a linear combination of the classifiers



(a) Classification accuracy (b) Runtime (s)

Figure 6: Accuracy and timing results with the second large-scale approach on the Reuters dataset with respect to the number of subsets.

modeled for the subsets. This approach has been successfully applied for single-view large-scale kernel methods in the past [36, 61].

For the Reuters dataset, we randomly pick a number of subsets with size $M = 1000$. The weights of the final classifier are calculated in the same way as for the baseline Comm method, i.e. based on the training error covariance matrix. The regularization parameters belonging to the specific subsets are separately trained for each subset to decrease tuning complexity. A first experiment shows the influence of the number of subsets on the classification accuracy and training and test runtime. The results are depicted in Figure 6.

Figure 6a shows that the accuracy increases with the number of subsets used in the committee network, although this increase is not as drastic as in Figure 5a. This would indicate that the subset size is of more importance than the number of subsets used. Since for each extra subset, an extra model needs to be trained in the training phase and an extra classifier needs to be calculated in the test phase, the runtime will also increase with the number of subsets, as depicted in Figure 6b. Since $M = 1000$, the training will however always be significantly faster than the test phase.

Table 7 summarizes the performance of both approaches on the Reuters dataset. Three random subsets are selected with $M = 1000$. The results are averaged over three random training-test splits. Note that for the first approach the results are averaged over the three splits as well as the three subsets.

The table also depicts the performance of a third approach, which is an extension to the committee approach where the weights of the final classifier

Table 7: Mean classification accuracy on the test set over the three splits for the large-scale Reuters datasets. The standard deviation is shown between brackets.

	Simple approach	Committee approach	Committee approach + weights tuned
ϱTMV-RKM$_{\text{add}}$	64.46 (±2.13)	57.24 (±1.99)	69.53 (±0.80)
ϱTMV-RKM$_{\text{mean}}$	69.45 (±1.54)	71.95 (±1.09)	73.23 (±0.54)

are tuned, instead of based on the training error. Tuning of the weights seems to be favorable to the second approach in terms of performance, but it clearly increases the time it takes to do model selection. The table further indicates that the committee approach can improve on the simple approach and that it is a natural and effective way to use the proposed ϱTMV-RKM model for large-scale data.

5. Conclusion and perspectives

This paper proposes a novel multi-view method called ϱ-Tensor Multi-View Restricted Kernel Machine Classification (ϱTMV-RKM), to perform classification when data is described through multiple views. The model includes principles from tensor learning to account for higher order correlations when three or more views are available. A first extension to the RKM formulation is shown, where shared hidden features are introduced. To increase the degree of coupling, a second extension is shown that includes a model tensor, containing the weights of all views. Finally a combination of both methods, ϱTMV-RKM is proposed. The performance is shown on a variety of real-world datasets and compared to the performance of state-of-the-art multi-view methods. The experimental results show the merit of including a weight tensor, both in terms of accuracy and time complexity. Multiple approaches to handle a large-scale dataset are proposed.

Future research could investigate the possibility of stacking multiple ϱTMV-RKM formulations, to perform deep multi-view learning with weight tensors. Another possibility, in line with HOFM [25, 26], is to impose a low-rank restriction on the weight tensor by adding a regularization term which includes the nuclear norm on the tensor.

Appendix A. Stationary points of MV-RKM

The stationary points of the objective function $\mathcal{J}$ of MV-RKM, given by Eq. (6), are characterized by:

$$\begin{cases} \dfrac{\partial \mathcal{J}}{\partial h_k} = 0 \rightarrow V = \displaystyle\sum_{v=1}^{V} (\varphi^{[v]}(\mathbf{x}_k^{[v]})^T \mathbf{w}^{[v]} + b) y_k + \lambda h_k \\ \qquad \text{for } k = 1, \ldots, N \\ \dfrac{\partial \mathcal{J}}{\partial \mathbf{w}^{[v]}} = 0 \rightarrow \mathbf{w}^{[v]} = \dfrac{1}{\eta} \displaystyle\sum_{k=1}^{N} \varphi^{[v]}(\mathbf{x}_k^{[v]}) y_k h_k \\ \qquad \text{for } v = 1, \ldots, V \\ \dfrac{\partial \mathcal{J}}{\partial b} = 0 \rightarrow \displaystyle\sum_{k=1}^{N} y_k h_k = 0. \end{cases} \tag{A.1}$$

Appendix B. Stationary points of TMV-RKM

The stationary points of the objective function $\mathcal{J}$ of TMV-RKM, given by Eq. (8), are characterized by:

$$\begin{cases} \dfrac{\partial \mathcal{J}}{\partial h_k} = 0 \rightarrow 1 = \left(\langle \Phi_{(k)}, \mathcal{W} \rangle + b \right) y_k + \lambda h_k \\ \qquad \text{for } k = 1, \ldots, N \\ \dfrac{\partial \mathcal{J}}{\partial \mathcal{W}_{i_1 \ldots i_V}} = 0 \rightarrow \mathcal{W}_{i_1 \ldots i_V} = \dfrac{1}{\eta} \displaystyle\sum_{k=1}^{N} \prod_{v=1}^{V} \varphi^{[v]}(\mathbf{x}_k^{[v]})_{i_v} y_k h_k \\ \qquad \text{for } i_v = 1, \ldots, d_h^{[v]} \\ \dfrac{\partial \mathcal{J}}{\partial b} = 0 \rightarrow \displaystyle\sum_{k=1}^{N} y_k h_k = 0 \end{cases} \tag{B.1}$$

Appendix C. Derivation of ϱTMV-RKM solution to training problem

The stationary points of this objective function given by Eq. (11), denoted as $\mathcal{J}$, are characterized by:

$$
\begin{cases}
\dfrac{\partial \mathcal{J}}{\partial h_k} = 0 \rightarrow \tau = \lambda h_k + \varrho \ \left(\langle \Phi_{(k)}, \mathcal{W} \rangle + b \right) y_k + (1-\varrho) \displaystyle\sum_{v=1}^{V} (\varphi^{[v]}(\mathbf{x}_k^{[v]})^T \mathbf{w}^{[v]} + b) y_k \\
\qquad \text{for } k = 1, \ldots, N \\
\dfrac{\partial \mathcal{J}}{\partial \mathcal{W}_{i_1 \ldots i_V}} = 0 \rightarrow \mathcal{W}_{i_1 \ldots i_V} = \dfrac{1}{\eta} \displaystyle\sum_{k=1}^{N} \varphi^{[1]}(\mathbf{x}_k^{[1]})_{i_1} \cdots \varphi^{[V]}(\mathbf{x}_k^{[V]})_{i_V} y_k h_k \\
\qquad \text{for } i_v = 1, \ldots, d_h^{[v]} \\
\dfrac{\partial \mathcal{J}}{\partial \mathbf{w}^{[v]}} = 0 \rightarrow \mathbf{w}^{[v]} = \dfrac{1}{\eta} \displaystyle\sum_{k=1}^{N} \varphi^{[v]}(\mathbf{x}_k^{[v]}) y_k h_k \\
\qquad \text{for } v = 1, \ldots, V \\
\dfrac{\partial \mathcal{J}}{\partial b} = 0 \rightarrow \tau \displaystyle\sum_{k=1}^{N} y_k h_k = 0 \rightarrow \sum_{k=1}^{N} y_k h_k = 0
\end{cases}
\tag{C.1}
$$

with $\tau = (1-\varrho)V + \varrho$. Given the definition in Eq. (9) we can rewrite the first condition as:

$$
\begin{aligned}
\tau =& \varrho \langle \Phi_{(k)}, \mathcal{W} \rangle y_k + (1-\varrho) \sum_{v=1}^{V} \varphi^{[v]}(\mathbf{x}_k^{[v]})^T \mathbf{w}^{[v]} y_k + \tau b y_k + \lambda h_k \\
=& \varrho \left(\sum_{i_1=1}^{d_h^{[1]}} \cdots \sum_{i_V=1}^{d_h^{[V]}} \varphi^{[1]}(\mathbf{x}_k^{[1]})_{i_1} \cdots \varphi^{[V]}(\mathbf{x}_k^{[V]})_{i_V} \mathcal{W}_{i_1 \ldots i_V} \right) y_k \\
&+ (1-\varrho) \sum_{v=1}^{V} \varphi^{[v]}(\mathbf{x}_k^{[v]})^T \mathbf{w}^{[v]} y_k + \tau b y_k + \lambda h_k.
\end{aligned}
\tag{C.2}
$$

Since $y_k \in \{-1, 1\}$ and by substituting the weight vectors and weight

tensor by means of the second and third condition it holds that:

$$\begin{aligned}\tau y_k =& \varrho \frac{1}{\eta} \prod_{v=1}^{V} \sum_{i_v=1}^{d_h^{[v]}} \varphi^{[v]}(\mathbf{x}_k^{[v]})_{i_v} \left(\sum_{j=1}^{N} \prod_{v=1}^{V} \varphi^{[v]}(\mathbf{x}_j^{[v]})_{i_v} \right) \\ &+ (1-\varrho) \frac{1}{\eta} \sum_{v=1}^{V} \varphi^{[v]}(\mathbf{x}_k^{[v]})^T \left(\sum_{j=1}^{N} \varphi^{[v]}(\mathbf{x}_j^{[v]}) y_j h_j \right) + \tau b + \lambda h_k y_k \qquad \text{(C.3)} \\ =& \varrho \frac{1}{\eta} \sum_{j=1}^{N} \left(\prod_{v=1}^{V} \Omega_{kj}^{[v]} \right) y_j h_j + (1-\varrho) \frac{1}{\eta} \sum_{j=1}^{N} \left(\sum_{v=1}^{V} \Omega_{kj}^{[v]} \right) y_j h_j + \tau b + \lambda h_k y_k.\end{aligned}$$

Since Eq. (C.3) holds for every $k = 1, \ldots, N$, we can rewrite it as:

$$\frac{1}{\eta} \left((1-\varrho) \sum_{v=1}^{V} \mathbf{\Omega}^{[v]} + \varrho \bigodot_{v=1}^{V} \mathbf{\Omega}^{[v]} \right) (\mathbf{y} \odot \mathbf{h}) + \lambda (\mathbf{y} \odot \mathbf{h}) + \boldsymbol{\tau}_N b = \tau \mathbf{y}. \qquad \text{(C.4)}$$

Together with the last condition in Eq. (C.1), that can be rewritten as $\mathbf{1}_N^T (\mathbf{y} \odot \mathbf{h}) = 0$, the formulation in Eq. (12) is obtained.

Acknowledgments

Research supported by ERC Advanced Grant E-DUALITY (787960), Research Council KUL: CoE PFV/10/002 (OPTEC), PhD/Postdoc grants Flemish Government; FWO: projects: G0A4917N (Deep restricted kernel machines), G.088114N (Tensor based data similarity).

References

[1] Y. Yang, C. Lan, X. Li, J. Huan, B. Luo, Automatic social circle detection using multi-view clustering, ACM Conference on Information and Knowledge Management (CIKM) (2014) 1019–1028.

[2] C. Zhang, E. Adeli, T. Zhou, X. Chen, D. Shen, Multi-Layer Multi-View Classification for Alzheimer's Disease Diagnosis, Proceedings of the AAAI Conference on Artificial Intelligence (2018) 4406–4413.

[3] M. P. Perrone, L. N. Cooper, When networks disagree: Ensemble methods for hybrid neural networks, in: Artificial Neural Networks for Speech and Vision, Chapman and Hall, 1993, pp. 126–142.

[4] S. Wang, E. Zhu, J. Hu, M. Li, K. Zhao, N. Hu, X. Liu, Efficient multiple kernel K-means clustering with late fusion, IEEE Access 7 (2019) 61109–61120.

[5] Z. Karevan, S. Mehrkanoon, J. A. K. Suykens, Black-box modeling for temperature prediction in weather forecasting, Proceedings of the International Joint Conference on Neural Networks (IJCNN) (2015) 1–8.

[6] S. Sun, X. Xie, C. Dong, Multiview Learning With Generalized Eigenvalue Proximal Support Vector Machines, IEEE Transactions on Cybernetics 49 (2) (2019) 688–697.

[7] L. Houthuys, R. Langone, J. A. K. Suykens, Multi-view least squares support vector machines classification, Neurocomputing 282 (2018) 78 – 88. doi:https://doi.org/10.1016/j.neucom.2017.12.029.

[8] H. Q. Minh, L. Bazzani, V. Murino, A unifying framework in vector-valued reproducing kernel hilbert spaces for manifold regularization and co-regularized multi-view learning, Journal of Machine Learning Research 17 (2016) 769–840.

[9] Y. Luo, D. Tao, K. Ramamohanarao, C. Xu, Y. Wen, Tensor canonical correlation analysis for multi-view dimension reduction, IEEE Transactions on Knowledge and Data Engineering 27 (11) (2015) 3111–3124.

[10] H. Hotelling, Relations between two sets of variates, Biometrica 28 (1936) 321–377.

[11] J. A. K. Suykens, Deep restricted kernel machines using conjugate feature duality, Neural Computation (2017) 2123–2163.

[12] R. K. Vinayak, T. Zrnic, B. Hassibi, Tensor-based crowdsourced clustering via triangle queries, in: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2017, pp. 2322–2326.

[13] A. Cichochi, D. P. Mandic, A. H. Phan, C. F. Caiafa, G. Zhou, Q. Zhao, L. De Lathauwer, Tensor Decompositions for Signal Processing Applications, IEEE Signal Processing Magazin (2015) 145–163.

[14] N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, C. Faloutsos, Tensor Decomposition for Signal Processing and Machine Learning, IEEE Transactions on Signal Processing (2017) 3551–3582.

[15] M. Signoretto, Q. Tran Dinh, L. De Lathauwer, J. A. K. Suykens, Learning with tensors: A framework based on convex optimization and spectral regularization, Machine Learning 94 (2014) 303–351. doi:10.1007/s10994-013-5366-3.

[16] K. Wimalawarne, M. Sugiyama, R. Tomioka, Multitask learning meets tensor factorization: Task imputation via convex optimization, Proceedings of Neural Information Processing Systems (NIPS) 4 (2014) 2825–2833.

[17] E. Adeli, Y. Meng, G. Li, W. Lin, D. Shen, Joint Sparse and Low-Rank Regularized MultiTask Multi-Linear Regression for Prediction of Infant Brain Development with Incomplete Data., Medical Image Computing and Computer Assisted Intervention (MICCAI) (2017) 40–48.

[18] J. Liu, C. Wang, J. Gao, J. Han, Multi-View Clustering via Joint Nonnegative Matrix Factorization, Proceedings of SIAM Data Mining Conference (SDM) (2013) 252–260.

[19] J. Wu, Z. Lin, H. Zha, Essential Tensor Learning for Multi-view Spectral Clustering, Proceedings of IEEE Transactions on Image Processing 28 (2019) 5910–5922.

[20] C. Zhang, H. Fu, S. Liu, G. Liu, X. Cao, Low-rank tensor constrained multiview subspace clustering, in: 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1582–1590.

[21] Y. Xie, D. Tao, W. Zhang, Y. Liu, L. Zhang, Y. Qu, On Unifying Multi-view Self-Representations for Clustering by Tensor Multi-rank Minimization, International Journal of Computer Vision (2018) 1157–1179.

[22] C.-T. Lu, L. He, W. Shao, B. Cao, P. S. Yu, Multilinear Factorization Machines for Multi-Task Multi-View Learning, Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM) (2017) 701–709.

[23] J. Yin, S. Sun, Multiview Uncorrelated Locality Preserving Projection, IEEE Transactions on Neural Networks and Learning Systems (2019) 1–14.

[24] L. Houthuys, J. A. K. Suykens, Tensor Learning in Multi-View Kernel PCA, Proc. of the 27th International Conference on Artificial Neural Networks (ICANN 2018) (2018) 205–215.

[25] M. Blondel, A. Fujino, N. Ueda, M. Ishihata, Higher-Order Factorization Machines, Proceedings of Neural Information Processing Systems (NIPS) (2016) 3351–3359.

[26] M. Blondel, V. Niculae, T. Otsuka, N. Ueda, Multi-output Polynomial Networks and Factorization Machines, Proceedings of Neural Information Processing Systems (NIPS) (2017) 3349–3359.

[27] B. Cao, H. Zhou, G. Li, P. S. Yu, Multi-view machines, Proceedings of the Ninth ACM International Conference on Web Search and Data Mining (WSDM) (2016) 427–436.

[28] S. Zheng, X. Cai, C. Ding, F. Nie, H. Huang, A closed form solution to multi-view low-rank regression, Proceedings AAAI Conference on Artificial Intelligence (2016) 1973–1979.

[29] M. Yang, C. Deng, F. Nie, Adaptive-weighting discriminative regression for multi-view classification, Pattern Recognition 88 (2019) 236–245.

[30] X. Xue, F. Nie, Z. Li, S. Wang, X. Li, M. Yao, A multiview learning framework with a linear computational cost, IEEE Transactions on Cybernetics (2018) 2416–2425.

[31] J. Xu, J. Han, F. Nie, Multi-view feature learning with discriminative regularization, in: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2017, pp. 3161–3167.

[32] X. Xie, S. Sun, Multi-view Support Vector Machines with the Consensus and Complementarity Information, IEEE Transactions on Knowledge and Data Engineering 4347 (2019) 1–14.

[33] M. Gönen, E. Alpaydm, Multiple Kernel Learning Algorithms, Journal of Machine Learning Research 12 (2011) 2211–2268.

[34] F. Aiolli, M. Donini, EasyMKL: A scalable multiple kernel learning algorithm, Neurocomputing 169 (2015) 215–224.

[35] C. Cortes, M. Mohri, A. Rostamizadeh, Multi-class classification with maximum margin multiple kernel, Proceedings of the International Conference on Machine Learning (ICML) 28 (2013) 46–54.

[36] J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, J. Vandewalle, Least Squares Support Vector Machines, World Scientific, (2002).

[37] G. E. Hinton, What kind of a graphical model is the brain?, in: Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI), IJCAI'05, 2005, pp. 1765–1775.

[38] J. Mercer, Functions of positive and negative type, and their connection with the theory of integral equations, Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character 209 (1909) 415–446.

[39] N. Kushmerick, Learning to remove internet advertisements, in: Proceedings of the Third Annual Conference on Autonomous Agents, 1999, pp. 175–181.

[40] M.-E. Nilsback, A. Zisserman, A visual vocabulary for flower classification, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2 (2006) 1447–1454.

[41] M.-E. Nilsback, A. Zisserman, Automated flower classification over a large number of classes, Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP) (2008) 722–729.

[42] T. Kolenda, L. K. Hansen, J. Larsen, O. Winther, Independent component analysis for understanding multimedia content, IEEE Workshop on Neural Networks for Signal Processing 12 (2002) 757–766.

[43] O. Madani, M. Georg, D. A. Ross, On using nearly-independent feature families for high precision and confidence, Machine Learning 92 (2013) 457–477.

[44] W. Yang, G. Toderici, Discriminative tag learning on youtube videos with latent subtags, in: Computer Vision and Pattern Recognition (CVPR), (2011), pp. 3217–3224.

[45] R. F. Lyon, M. Rehn, S. Bengio, T. C. Walters, G. Chechik, Sound retrieval and ranking using sparse auditory representations, Neural computation 22 (9) (2010) 2390–2416.

[46] M. van Breukelen, R. Duin, D. Tax, J. den Hartog, Handwritten digit recognition by combined classifiers, Kybernetika 34 (1998) 381 – 386.

[47] M. Lichman, UCI machine learning repository (2013).

[48] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, Y. Zheng, NUS-WIDE: a real-world web image database from national university of Singapore, International Conference on Image and Video Retrieval 34 (2009) 48:1–48:9.

[49] P. Li, G. Samorodnitsky, J. Hopcroft, Sign Cauchy projections and Chi-square kernel, Proceedings of Neural Information Processing Systems (NIPS) 26 (2013) 2571–2579.

[50] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, C. Watkins, Text Classification using String Kernels, Journal of Machine Learning Research 2 (2002) 419–444.

[51] T. Diethe, D. Hardoon, J. Shawe-Taylor, Constructing nonlinear discriminants from multiple data views, in: Machine Learning and Knowledge Discovery in Databases (ECML PKDD), 2010, pp. 328–343.

[52] A. Rakotomamonjy, F. R. Bach, S. Canu, Y. Grandvalet, SimpleMKL, Journal of Machine Learning Research 9 (2008) 2491–2521.

[53] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research 7 (2006) 1–30.

[54] E. B. Wilson, Probable inference, the law of succession, and statistical inference, Journal of the American Statistical Association 22 (1927) 209–212.

[55] M.-R. Amini, N. Usunier, C. Goutte, Learning from multiple partially observed views - an application to multilingual text categorization, Advances in Neural Information Processing Systems (2009) 28–36.

[56] L. Houthuys, R. Langone, J. A. K. Suykens, Multi-View Kernel Spectral Clustering, Information Fusion 44 (2018) 46–56.

[57] M. Girolami, Orthogonal series density estimation and the kernel eigenvalue problem, Neural Computation (2002) 669 – 688.

[58] S. Mehrkanoon, J. A. K. Suykens, Large scale semi-supervised learning using KSC based model, Proceedings of the International Joint Conference on Neural Networks (IJCNN) (2014) 4152 – 4159.

[59] C. Alzate, J. A. K. Suykens, Multiway spectral clustering with out-of-sample extensions through weighted kernel PCA, IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (2) (2010) 335–347.

[60] R. Mall, R. Langone, J. A. K. Suykens, Kernel spectral clustering for big data networks, Entropy 15 (2013) 1567–1586.

[61] V. Tresp, Scaling kernel-based systems to large data sets, Data Mining and Knowledge Discovery 5 (3) (2001) 197–211.