Multiview Learning of Weighted Majority Vote by Bregman Divergence Minimization

Anil Goyal[1,2(✉)], Emilie Morvant[1], and Massih-Reza Amini[2]

[1] Laboratoire Hubert Curien UMR 5516, Université de Lyon, UJM-St-Etienne, CNRS, Institut d'Optique Graduate School, 42023 St-Etienne, France
anil.goyal@univ-st-etienne.fr

[2] Laboratoire d'Informatique de Grenoble, AMA, Université Grenoble Alps, 38058 Grenoble, France

Abstract. We tackle the issue of classifier combinations when observations have multiple views. Our method jointly learns view-specific weighted majority vote classifiers (*i.e.* for each view) over a set of base voters, and a second weighted majority vote classifier over the set of these view-specific weighted majority vote classifiers. We show that the empirical risk minimization of the final majority vote given a multiview training set can be cast as the minimization of Bregman divergences. This allows us to derive a parallel-update optimization algorithm for learning our multiview model. We empirically study our algorithm with a particular focus on the impact of the training set size on the multiview learning results. The experiments show that our approach is able to overcome the lack of labeled information.

Keywords: Multiview learning · Bregman divergence · Majority vote

1 Introduction

In many real-life applications, observations are produced by more than one source and are so-called multiview [22]. For example, in multilingual regions of the world, including many regions of Europe or in Canada, documents are available in more than one language. The aim of multiview learning is to use this multimodal information by combining the predictions of each classifier (or the models themselves) operating over each view (called view-specific classifier) in order to improve the overall performance beyond that of predictors trained on each view separately, or by combining directly the views [21].

Related works. The main idea here follows the conclusion of the seminal work of *Blum and Mitchell* [3] which states that correlated yet not completely redundant views contain valuable information for learning. Based on this idea, many studies on multiview learning have been conducted and they can be grouped in three main categories. These approaches exploit the redundancy in different

W. Duivesteijn et al. (Eds.): IDA 2018, LNCS 11191, pp. 124–136, 2018.
https://doi.org/10.1007/978-3-030-01768-2_11

representations of data, either by projecting the view-specific representations in a common canonical space [10,25,29], or by constraining the classifiers to have *similar* outputs on the same observations; for example by adding a disagreement term in their objective functions [20], or lastly by exploiting diversity in the views in order to learn the final classifier defined as the majority vote over the set of view-specific classifiers [17,18,24]. While the two first families of approaches were designed for learning with labeled and unlabeled training data, the last one, were developed in the context of supervised learning. In this line, most of the supervised multiview learning algorithms dealt with the particular case of two view learning [9,12,28], and some recent works studied the general case of multiview learning with more than two views under the majority vote setting. *Amini et al.* [1] derived a generalization error bound for classifiers learned on multiview examples and identified situations where it is more interesting to use all views to learn a uniformly weighted majority vote classifier instead of single view learning. *Koço et al.* [13] proposed a Boosting-based strategy that maintains a different distribution of examples with respect to each view. For a given view, the corresponding distribution is updated based on view-specific *weak* classifiers from that view and all the other views with the idea of using all the view-specific distributions to weight hard examples for the next iteration. *Peng et al.* [17,18] enhanced this idea by maintaining a single weight distribution among the multiple views in order to ensure consistency between them. *Xiao et al.* [24] proposed a multiview learning algorithm where they boost the performance of view-specific classifiers by combining multiview learning with Adaboost.

Contribution. In this work, we propose a multiview Boosting-based algorithm, called $\mathtt{M}\omega\mathtt{MvC}^2$, for the general case where observations are described by more than two views. Our algorithm combines previously learned view-specific classifiers as in [1] but with the difference that it jointly learns two sets of weights for, first, combining view-specific *weak classifiers*; and then combining the obtained view-specific weighted majority vote classifiers to get a final weighted majority vote classifier. We show that the minimization of the classification error over a multiview training set can be cast as the minimization of Bregman divergences allowing the development of an efficient parallel update scheme to learn the weights. Using a large publicly available corpus of multilingual documents extracted from the `Reuters` RCV1 and RCV2 corpora as well as $\mathtt{MNIST}_1$ and $\mathtt{MNIST}_2$ collections, we show that our approach consistently improves over other methods, in the particular when there are only few training examples available for learning. This is a particularly interesting setting when resources are limited, and corresponds, for example, to the common situation of multilingual data.

Organization of the paper. In the next section, we present the double weighted majority vote classifier for multiview learning. Section 3 shows that the learning problem is equivalent to a Bregman-divergence minimization and describes the Boosting-based algorithm we developed to learn the classifier. In Sect. 4, we present experimental results obtained with our approach. Finally, in

Sect. 5 we discuss the outcomes of this study and give some pointers to further research.

2 Notations and Setting

For any positive integer N, $[N]$ denotes the set $[N]\doteq\{1,\ldots,N\}$. We consider binary classification problems with $V\geq 2$ input spaces $\mathcal{X}_v \subset \mathbb{R}^{d_v}; \forall v \in [V]$, and an output space $\mathcal{Y}=\{-1,+1\}$. Each *multiview observation* $\mathbf{x} \in \mathcal{X}_1\times\cdots\times\mathcal{X}_V$ is a sequence $\mathbf{x}\doteq(x^1,\cdots,x^V)$ where each *view* x^v provides a representation of the same observation in a different vector space $\mathcal{X}_v$ (each vector space are not necessarily of the same dimension). We further assume that we have a finite set of *weak classifiers* $\mathcal{H}_v\doteq\{h_{v,j} : \mathcal{X}_v \to \{-1,+1\} \mid j \in [n_v]\}$ of size n_v. We aim at learning a two-level weighted majority vote classifier where at the first level a weighted majority vote is built for each view $v\in[V]$ over the associated set of weak classifiers $\mathcal{H}_v$, and the final classifier, referred to as the Multiview double ωeighted Majority vote Classifier ($\mathtt{M}\omega\mathtt{Mv}\mathtt{C}^2$), is a weighted majority vote over the previous view-specific majority vote classifiers (see Fig. 1 for an illustration). Given a training set $\mathcal{S}=(\mathbf{x}_i,y_i)_{1\leq i\leq m}$ of size m drawn *i.i.d.* with respect to a fixed, yet unknown, distribution $\mathcal{D}$ over $(\mathcal{X}_1\times\cdots\times\mathcal{X}_V)\times\mathcal{Y}$, the learning objective is to train the weak view-specific classifiers $(\mathcal{H}_v)_{1\leq v\leq V}$ and to choose two sets of weights; $\boldsymbol{\Pi}=(\boldsymbol{\pi}_v)_{1\leq v\leq V}$, where $\forall v\in[V]$, $\boldsymbol{\pi}_v=(\pi_{v,j})_{1\leq j\leq n_v}$, and $\boldsymbol{\rho}=(\rho_v)_{1\leq v\leq V}$, such that the $\boldsymbol{\rho\Pi}$-weighted majority vote classifier $B_{\boldsymbol{\rho\Pi}}$

$$B_{\boldsymbol{\rho\Pi}}(\mathbf{x}) = \sum_{v=1}^{V} \rho_v \sum_{j=1}^{n_v} \pi_{v,j}\, h_{v,j}(x^v) \tag{1}$$

has the smallest possible generalization error on $\mathcal{D}$. We follow the Empirical Risk Minimization principle [23], and aim at minimizing the 0/1-loss over $\mathcal{S}$:

$$\hat{\mathcal{L}}_m^{0/1}(B_{\boldsymbol{\rho\Pi}},\mathcal{S}) = \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}_{y_i B_{\boldsymbol{\rho\Pi}}(\mathbf{x}_i)\leq 0},$$

where $\mathbb{1}_p$ is equal to 1 if the predicate p is true, and 0 otherwise. As this loss function is non-continuous and non-differentiable, it is typically replaced by an appropriate convex and differentiable proxy. Here, we replace $\mathbb{1}_{z\leq 0}$ by the logistic upper bound $a\log(1+e^{-z})$, with $a=(\log 2)^{-1}$. The misclassification cost becomes

$$\hat{\mathcal{L}}_m(B_{\boldsymbol{\rho\Pi}},\mathcal{S}) = \frac{a}{m}\sum_{i=1}^{m} \ln\Big(1+\exp\big(-y_i B_{\boldsymbol{\rho\Pi}}(\mathbf{x}_i)\big)\Big), \tag{2}$$

and the objective would be then to find the optimal combination weights $\boldsymbol{\Pi}^\star$ and $\boldsymbol{\rho}^\star$ that minimize this surrogate logistic loss.

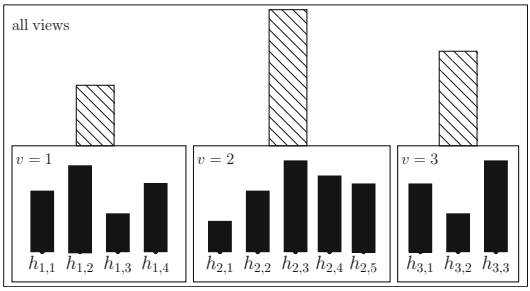


Fig. 1. Illustration of $\mathtt{M}\omega\mathtt{MvC}^2$ with V=3. For all views $v \in \{1,2,3\}$, we have a set of view-specific weak classifiers $(\mathcal{H}_v)_{1\leq v\leq V}$ that are learned over a multiview training set. The objective is then to learn the weights $\boldsymbol{\Pi}$ (black histograms) associated to $(\mathcal{H}_v)_{1\leq v\leq V}$; and the weights $\boldsymbol{\rho}$ (hatched histograms) associated to weighted majority vote classifiers such that the $\boldsymbol{\rho\Pi}$-weighted majority vote classifier $B_{\rho\Pi}$ (Eq. 1) will have the smallest possible generalization error.

3 An Iterative Parallel Update Algorithm to Learn $\mathtt{M}\boldsymbol{\omega}\mathtt{MvC}^{\mathbf{2}}$

In this section, we first show how the minimization of the surrogate loss of Eq. (2) is equivalent to the minimization of a given Bregman divergence. Then, this equivalence allows us to employ a parallel-update optimization algorithm to learn the weights $\boldsymbol{\Pi}=(\boldsymbol{\pi}_v)_{1\leq v\leq V}$ and $\boldsymbol{\rho}$ leading to this minimization.

3.1 Bregman-Divergence Optimization

We first recall the definition of a Bregman divergence [4,14].

Definition 1 (Bregman divergence). *Let $\Omega \subseteq \mathbb{R}^m$ and $F : \Omega \to \mathbb{R}$ be a continuously differentiable and strictly convex real-valued function. The Bregman divergence D_F associated to F is defined for all $(\mathbf{p},\mathbf{q}) \in \Omega \times \Omega$ as*

$$D_F(\mathbf{p}||\mathbf{q}) \doteq F(\mathbf{p}) - F(\mathbf{q}) - \langle \nabla F(\mathbf{q}), (\mathbf{p}-\mathbf{q})\rangle\,, \tag{3}$$

where $\nabla F(\mathbf{q})$ is the gradient of F estimated at $\mathbf{q}$, and the operator $\langle\cdot,\cdot\rangle$ is the dot product function.

The optimization problem arising from this definition that we are interested in, is to find a vector $\mathbf{p}^\star \in \Omega$—that is the closest to a given vector $\mathbf{q}_0 \in \Omega$—under the set $\mathcal{P}$ of V linear constraints

$$\mathcal{P} \doteq \{\mathbf{p} \in \Omega | \forall v \in [V],\ \rho_v \mathbf{p}^\top \mathbf{M}_v = \rho_v \tilde{\mathbf{p}}^\top \mathbf{M}_v\},$$

where $\tilde{\mathbf{p}}\in\Omega$ is a specified vector, and $\mathbf{M}_v$ is a $m\times n_v$ matrix with $n_v=|\mathcal{H}_v|$ the number of weak classifiers for view $v\in[V]$. Defining the Legendre transform as

$$L_F\left(\mathbf{q}, \sum_{v=1}^{V} \rho_v \mathbf{M}_v \boldsymbol{\pi}_v\right) \doteq \underset{\mathbf{p}\in\Omega}{\arg\min} \left\{ D_F(\mathbf{p}||\mathbf{q}) + \sum_{v=1}^{V} \langle \rho_v \mathbf{M}_v \boldsymbol{\pi}_v, \mathbf{p}\rangle \right\}.$$

the dual optimization problem can be stated as finding a vector $\mathbf{q}^\star$ in $\bar{\mathcal{Q}}$, the closure of the set

$$\mathcal{Q} \doteq \left\{ \mathbf{q} = L_F\left(\mathbf{q}_0, \sum_{v=1}^{V} \rho_v \mathbf{M}_v \boldsymbol{\pi}_v\right) \middle| \boldsymbol{\rho} \in \mathbb{R}^V; \forall v, \boldsymbol{\pi}_v \in \mathbb{R}^{n_v} \right\},$$

for which $D_F(\tilde{\mathbf{p}}||\mathbf{q}^\star)$ is the lowest. It can be shown that both of these optimization problems have the same unique solution [8,14], with the advantage of having parallel-update optimization algorithms to find the solution of the dual form in the mono-view case [6–8], making the use of the latter more appealing.

According to our multiview setting and to optimize Eq. (2) through a Bregman divergence, we consider the function F defined for all $\mathbf{p} \in \Omega = [0,1]^m$ as

$$F(\mathbf{p}) \doteq \sum_{i=1}^{m} p_i \ln(p_i) + (1-p_i)\ln(1-p_i),$$

which from Definition 1 and the definition of the Legendre transform, yields that for all $(\mathbf{p}, \mathbf{q}) \in \Omega \times \Omega$ and $\mathbf{r} \in \Omega$

$$D_F(\mathbf{p}||\mathbf{q}) = \sum_{i=1}^{m} p_i \ln\left(\frac{p_i}{q_i}\right) + (1-p_i)\ln\left(\frac{1-p_i}{1-q_i}\right), \tag{4}$$

$$and \forall i \in [m],\ L_F(\mathbf{q}, \mathbf{r})_i = \frac{q_i e^{-r_i}}{1 - q_i + q_i e^{-r_i}}, \tag{5}$$

with a_i the i^{th} characteristic of $\mathbf{a}=(a_i)_{1\leq i\leq m}$ ($\mathbf{a}$ being $\mathbf{p}$, $\mathbf{q}$, $\mathbf{r}$ or $L_F(\mathbf{q},\mathbf{r})$).

Now, let $\mathbf{q}_0 = \frac{1}{2}\mathbf{1}_m$ be the vector with all its components set to $\frac{1}{2}$. For all $i \in [m]$, we define $L_F(\mathbf{q}_0, \mathbf{v})_i = \sigma(v_i)$ with $\sigma(z) = (1+e^z)^{-1}$, $\forall z \in \mathbb{R}$. We set the matrix $\mathbf{M}_v$ as for all $(i,j) \in [m] \times [n_v]$, $(\mathbf{M}_v)_{ij} = y_i h_{v,j}(x_i^v)$. Then using Eqs. (4) and (5), it comes

$$D_F\left(\mathbf{0}\middle|\middle|L_F\left(\mathbf{q}_0, \sum_{v=1}^{V} \rho_v \mathbf{M}_v \boldsymbol{\pi}_v\right)\right) = \sum_{i=1}^{m} \ln\left(1+\exp\left(-y_i \sum_{v=1}^{V} \rho_v \sum_{j=1}^{n_v} \pi_{v,j} h_{v,j}(x_i^v)\right)\right). \tag{6}$$

As a consequence, minimizing Eq. (2) is equivalent to minimizing $D_F(\mathbf{0}||\mathbf{q})$ over $\mathbf{q} \in \bar{\mathcal{Q}}_0$, where for $\Omega = [0,1]^m$

$$\mathcal{Q}_0 = \left\{ \mathbf{q} \in \Omega \middle| q_i = \sigma\left(y_i \sum_{v=1}^{V} \rho_v \sum_{j=1}^{n_v} \pi_{v,j} h_{v,j}(x_i^v)\right); \boldsymbol{\rho}, \boldsymbol{\Pi} \right\}. \tag{7}$$

For a set of weak-classifiers $(\mathcal{H}_v)_{1\leq v\leq V}$ learned over a training set $\mathcal{S}$; this equivalence allows us to adapt the parallel-update optimization algorithm described in [6] to find the optimal weights $\boldsymbol{\Pi}$ and $\boldsymbol{\rho}$ defining $\mathtt{M}\omega\mathtt{MvC}^2$ of Eq. (1), as described in Algorithm 1.

Algorithm 1 Learning $\mathtt{M}\omega\mathtt{Mv}\mathtt{C}^2$

Input: Training set $\mathcal{S} = (\mathbf{x}_i, y_i)_{1\leq i\leq m}$, where $\forall i, \mathbf{x}_i = (x_i^1, \ldots, x_i^V)$ and $y_i \in \{-1, 1\}$; and a maximal number of iterations T.
Initialization: $\boldsymbol{\rho}^{(1)} \leftarrow \frac{1}{V}\mathbf{1}_V$ and $\forall v, \boldsymbol{\pi}_v^{(1)} \leftarrow \frac{1}{n_V}\mathbf{1}_{n_v}$
Train the weak classifiers $(\mathcal{H}_v)_{1\leq v\leq V}$ over $\mathcal{S}$
For $v \in [V]$ set the $m \times n_v$ matrix $\mathbf{M}_v$ such that $\forall i \in [m]$, $\forall j \in [n_v]$, $(\mathbf{M}_v)_{ij} = y_i h_{v,j}(x_i^v)$

for $t = 1, \ldots, T$ **do**
  for $i = 1, \ldots, m$ **do**
    $q_i^{(t)} = \sigma\left(y_i \sum_{v=1}^{V} \rho_v^{(t)} \sum_{j=1}^{n_v} \pi_{v,j}^{(t)}\ h_{v,j}(x_i^v)\right)$
  for $v = 1, \ldots, V$ **do**
    for $j = 1, \ldots, n_v$ **do**
      $W_{v,j}^{(t)+} = \sum_{i:\mathrm{sign}((\mathbf{M}_v)_{ij})=+1}\ q_i^{(t)}|(\mathbf{M}_v)_{ij}|$
      $W_{v,j}^{(t)-} = \sum_{i:\mathrm{sign}((\mathbf{M}_v)_{ij})=-1}\ q_i^{(t)}|(\mathbf{M}_v)_{ij}|$
      $\delta_{v,j}^{(t)} = \frac{1}{2}\ln\left(\frac{W_{v,j}^{(t)+}}{W_{v,j}^{(t)-}}\right)$
    $\boldsymbol{\pi}_v^{(t+1)} = \boldsymbol{\pi}_v^{(t)} + \boldsymbol{\delta}_v^{(t)}$
  Set $\boldsymbol{\rho}^{(t+1)}$, as the solution of :

$$\min_{\boldsymbol{\rho}} \quad -\sum_{v=1}^{V} \rho_v \sum_{j=1}^{n_v} \left(\sqrt{W_{v,j}^{(t)+}} - \sqrt{W_{v,j}^{(t)-}}\right)^2 \tag{8}$$

$$\text{s.t.} \quad \sum_{v=1}^{V} \rho_v = 1, \quad \rho_v \geq 0 \quad \forall v \in [V]$$

Return: Weights $\boldsymbol{\rho}^{(T)}$ and $\boldsymbol{\Pi}^{(T)}$.

3.2 A Multiview Parallel Update Algorithm

Once all view-specific *weak classifiers* $(\mathcal{H}_v)_{1\leq v\leq V}$ have been trained, we start from an initial point $\mathbf{q}^{(1)} \in \mathcal{Q}_0$ (Eq. 7) corresponding to uniform values of weights $\boldsymbol{\rho}^{(1)} = \frac{1}{V}\mathbf{1}_V$ and $\forall v \in [V]$, $\boldsymbol{\pi}_v^{(1)} = \frac{1}{n_v}\mathbf{1}_{n_v}$. Then, we iteratively update the weights such that at each iteration t, using the current parameters $\boldsymbol{\rho}^{(t)}, \boldsymbol{\Pi}^{(t)}$ and $\mathbf{q}^{(t)} \in \mathcal{Q}_0$, we seek new parameters $\boldsymbol{\rho}^{(t+1)}$ and $\boldsymbol{\delta}_v^{(t)}$ such that for

$$\mathbf{q}^{(t+1)} = L_F(\mathbf{q}_0, \sum_{v=1}^{V} \rho_v^{(t+1)} \mathbf{M}_v(\boldsymbol{\pi}_v^{(t)} + \boldsymbol{\delta}_v^{(t)})), \tag{9}$$

we get $D_F(0||\mathbf{q}^{(t+1)}) \leq D_F(0||\mathbf{q}^{(t)})$.

Following [6, Theorem 3], it is straightforward to show that in this case, the following inequality holds:

$$D_F(\mathbf{0}||\mathbf{q}^{(t+1)}) - D_F(\mathbf{0}||\mathbf{q}^{(t)}) \leq A^{(t)}\,, \tag{10}$$

$$\textit{where} \quad A^{(t)} = -\sum_{v=1}^{V} \rho_v^{(t+1)} \sum_{j=1}^{n_v} \left(W_{v,j}^{(t)+}(e^{-\delta_{v,j}^{(t)}} - 1) - W_{v,j}^{(t)-}(e^{\delta_{v,j}^{(t)}} - 1) \right)^2,$$

with $\forall j \in [n_v]; W_{v,j}^{(t)\pm} = \sum_{i:\text{sign}((\mathbf{M}_v)_{ij})=\pm 1} q_i^{(t)} |(\mathbf{M}_v)_{ij}|$.

By fixing the set of parameters $\boldsymbol{\rho}^{(t+1)}$; the parameters $\boldsymbol{\delta}_v^{(t)}$ that minimize $A^{(t)}$ are defined as $\forall v \in [V], \forall j \in [n_v]; \delta_{v,j}^{(t)} = \frac{1}{2} \ln \left(\frac{W_{v,j}^{(t)+}}{W_{v,j}^{(t)-}} \right)$. Plugging back these values into the above equation gives

$$A^{(t)} = -\sum_{v=1}^{V} \rho_v^{(t+1)} \sum_{j=1}^{n_v} \left(\sqrt{W_{v,j}^{(t)+}} - \sqrt{W_{v,j}^{(t)-}} \right)^2. \tag{11}$$

Now by fixing the set of parameters $(W_{v,j}^{(t)\pm})_{v,j}$, the weights $\boldsymbol{\rho}^{(t+1)}$ are found by minimizing Eq. (11) under the linear constraints $\forall v \in [V], \rho_v \geq 0$ and $\sum_{v=1}^{V} \rho_v = 1$. This alternating optimization of $A^{(t)}$ bears similarity with the block-coordinate descent technique [2], where at each iteration, variables are split into two subsets—the set of the active variables, and the set of the inactive ones—and the objective function is minimized along active dimensions while inactive variables are fixed at current values.

Convergence of Algorithm. The sequences of weights $(\boldsymbol{\Pi}^{(t)})_{t\in\mathbb{N}}$ and $(\boldsymbol{\rho}^{(t)})_{t\in\mathbb{N}}$ found by Algorithm 1 converge to the minimizers of the multiview classification loss (Eq. 2), as with the resulting sequence $(\mathbf{q}^{(t)})_{t\in\mathbb{N}}$ (Eq. 9), the sequence $(D_F(\mathbf{0}||\mathbf{q}^{(t)}))_{t\in\mathbb{N}}$ is decreasing and since it is lower-bounded (Eq. 6), it converges to the minimum of Eq. (2).

3.3 A Note on the Complexity of Algorithm

For each view v, the complexity of learning decision tree classifiers is $O(d_v\, mlog(m))$. We learn the weights over the views by optimizing Eq. (11) (Step 10 of our algorithm) using SLSQP method which has time complexity of $O(V^3)$. Therefore, the overall complexity is $O(V\, d_v\, m.log(m) + T\,(V^3 + \sum_{v=1}^{V} m\, n_v))$. Note that it is easy to parallelize our algorithm: by using V different machines, we can learn the view-specific classifiers and weights over them (Steps 4 to 9).

4 Experimental Results

We present below the results of the experiments we have performed to evaluate the efficiency of Algorithm 1 to learn the set of weights $\boldsymbol{\Pi}$ and $\boldsymbol{\rho}$ involved in the definition of the $\boldsymbol{\rho\Pi}$-weighted majority vote classifier $B_{\boldsymbol{\rho\Pi}}$ (Eq. (1)).

4.1 Datasets

`MNIST.` is a publicly available dataset consisting of 70,000 images of handwritten digits distributed over 10 classes [15]. For our experiments, we created 2 multiview collections from the initial dataset. Following [5], the first dataset ($\texttt{MNIST}_1$) was created by extracting 4 no-overlapping quarters of each image considered as its 4 views. The second dataset ($\texttt{MNIST}_2$) was made by extracting 4 overlapping quarters from each image as its 4 views. We randomly split each collection by keeping 10,000 images for testing and the remaining images for training.

`Reuters` RCV1/RCV2. is a multilingual text classification data extracted from Reuters RCV1 and RCV2 corpus[1]. It consists of more than 110,000 documents written in five different languages (English, French, German, Italian and Spanish) distributed over six classes. In this paper we consider each language as a view. We reserved 30% of documents for testing and the remaining for training.

Table 1. Test classification accuracy and F_1-score of different approaches averaged over all the classes and over 20 random sets of $m = 100$ labeled examples per training set. Along each column, the best result is in bold, and second one in italic. $^{\downarrow}$ indicates that a result is statistically significantly worse than the best result, according to a Wilcoxon rank sum test with $p < 0.02$.

Strategy	$\texttt{MNIST}_1$		$\texttt{MNIST}_2$		`Reuters`	
	Accuracy	F_1	Accuracy	F_1	Accuracy	F_1
`Mono`	.7827 ± .008$^{\downarrow}$	.4355 ± .009$^{\downarrow}$	.7896 ± .008$^{\downarrow}$	.4535 ± .011$^{\downarrow}$	.7089 ± .017$^{\downarrow}$	.4439 ± .007$^{\downarrow}$
`Concat`	.7988 ± .011$^{\downarrow}$	.4618 ± .015$^{\downarrow}$	.7982 ± .017$^{\downarrow}$	.4653 ± .021$^{\downarrow}$	.6918 ± .029$^{\downarrow}$	.4378 ± .015$^{\downarrow}$
`Fusion`	.8167 ± .017$^{\downarrow}$	.4769 ± .018$^{\downarrow}$	.8244 ± .019$^{\downarrow}$	.4955 ± .027$^{\downarrow}$	.7086 ± .029$^{\downarrow}$	.4200 ± .021$^{\downarrow}$
`MVMLsp`	.7221 ± .021$^{\downarrow}$	.3646 ± .019$^{\downarrow}$	.7669 ± .032$^{\downarrow}$	.4318 ± .025$^{\downarrow}$	.6037 ± .020$^{\downarrow}$	.3181 ± .022$^{\downarrow}$
`MV-MV`	.8381 ± .009$^{\downarrow}$	.5238 ± .015$^{\downarrow}$	*.8380* ± .010$^{\downarrow}$	.5307 ± .016$^{\downarrow}$	.7453 ± .023$^{\downarrow}$	.4979 ± .012$^{\downarrow}$
`MVWAB`	*.8470* ± .015$^{\downarrow}$	*.5704* ± .012$^{\downarrow}$	.8331 ± .016$^{\downarrow}$	*.5320* ± .011$^{\downarrow}$	.7484 ± .017$^{\downarrow}$	.5034 ± .016$^{\downarrow}$
`rBoost.SH`	.7580 ± .011$^{\downarrow}$	.4067 ± .009$^{\downarrow}$	.8247 ± .009$^{\downarrow}$	.5148 ± .015$^{\downarrow}$	*.7641* ± .014	*.5093* ± .010$^{\downarrow}$
`MωMvC`[2]	**.8659** ± .011	**.5914** ± .015	**.8474** ± .012	**.5523** ± .018	**.7662** ± .010	**.5244** ± .012

4.2 Experimental Protocol

In our experiments, we set up binary classification tasks by using all multiview observations from one class as positive examples and all the others as negative examples. We reduced the imbalance between positive and negative examples by subsampling the latter in the training sets, and used decision trees as view specific weak classifiers. We compare our approach to the following seven algorithms.

- `Mono` is the best performing decision tree model operating on a single view.
- `Concat` is an early fusion approach, where a mono-view decision tree operates over the concatenation of all views of multiview observations.

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

• `Fusion` is a late fusion approach, sometimes referred to as stacking, where view-specific classifiers are trained independently over different views using 60% of the training examples. A final multiview model is then trained over the predictions of the view-specific classifiers using the rest of the training examples.

• `MVMLsp` [11] is a multiview metric learning approach, where multiview kernels are learned to capture the view-specific information and relation between the views. We kept the experimental setup of [11] with Nyström parameter 0.24.[2]

• `MV-MV` [1] is a multiview algorithm where view-specific classifiers are trained over the views using all the training examples. The final model is the uniformly weighted majority vote.

• `MVWAB` [24] is a Multiview Weighted Voting AdaBoost algorithm, where multiview learning and ababoost techniques are combined to learn a weighted majority vote over view-specific classifiers but without any notion of learning weights over views.

• `rBoost.SH` [17,18] is a multiview boosting approach where a single distribution over different views of training examples is maintained and, the distribution over the views are updated using the multiarmed bandit framework. For the tuning of parameters, we followed the experimental setup of [17].

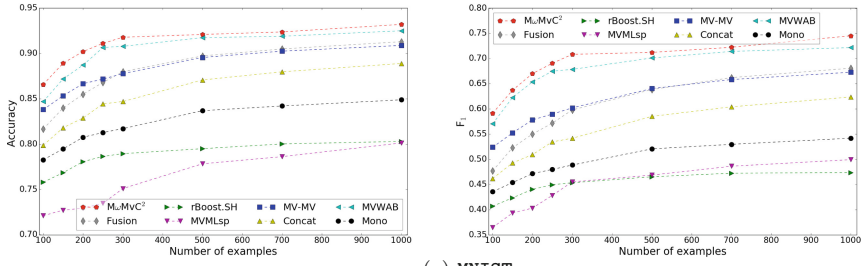
`Fusion`, `MV-MV`, `MVWAB`, and `rBoost.SH` make decision based on some majority vote strategies, as the proposed `MωMvC`^{2} classifier. The difference relies on how the view-specific classifiers are combined. For `MVWAB` and `rBoost.SH`, we used decision tree model to learn view-specific weak classifiers at each iteration of algorithm and fixed the maximum number of iterations to $T = 100$. To learn `MωMvC`^{2}, we generated the matrix $\mathbf{M}_v$ by considering a set of weak decision tree classifiers with different depths (from 1 to $\max_d - 2$, where $\max_d$ is maximum possible depth of a decision tree). We tuned the maximum number of iterations by cross-validation which came out to be $T = 2$ in most of the cases and that we fixed throughout all of the experiments. To solve the optimization problem for finding the weights $\boldsymbol{\rho}$ (Eq. 8), we used the Sequential Least SQuares Programming (SLSQP) implementation of scikit-learn [16], that we also used to learn the decision trees. Results are computed over the test set using the accuracy and the standard F_1-score [19], which is the harmonic average of precision and recall. Experiments are repeated 20 times by each time splitting the training and the test sets at random over the initial datasets.

4.3 Results

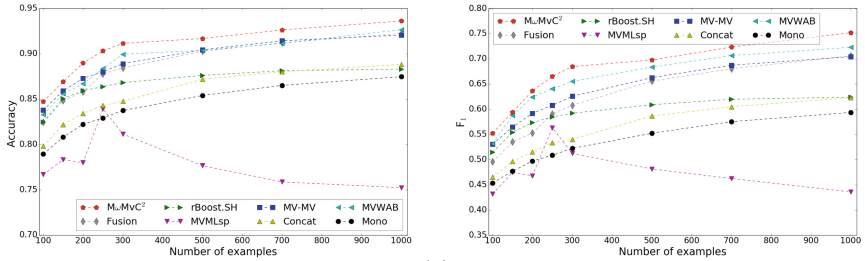
Table 1 reports the results obtained for m=100 training examples by different methods averaged over all classes and the 20 test results obtained over 20 random experiments[3]. From these results it becomes clear that late fusion and other multiview approaches (except `MVMLsp`) provide consistent improvements over training independent mono-view classifiers and with early fusion, when the size of

[2] We used the Python code available from https://lives.lif.univ-mrs.fr/?page_id=12.

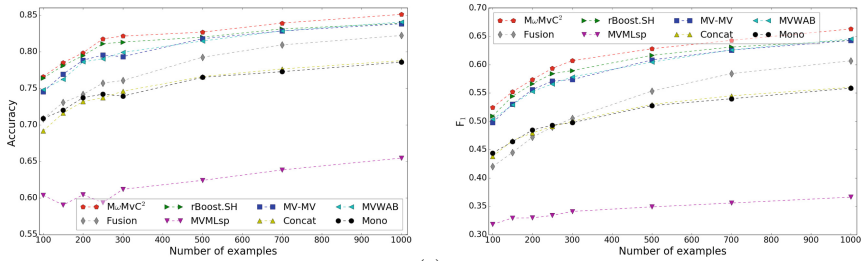
[3] We also did experiments for `Mono`, `Concat`, `Fusion`, `MV-MV` using Adaboost. The performance of Adaboost for these baselines is similar to that of decision trees.



(a) MNIST$_1$



(b) MNIST$_2$



(c) Reuters

Fig. 2. Evolution of accuracy and F_1-score *w.r.t* to the number of labeled examples in the initial labeled training sets on MNIST$_1$, MNIST$_2$ and Reuters datasets.

the training set is small. Furthermore, MωMvC^2 outperforms the other approaches and compared to the second best strategy the gain in accuracy (*resp.* F_1-score) varies between 0.2% and 2.2% (*resp.* 2.2% and 3.8%) across the collections. These results provide evidence that majority voting for multiview learning is an effective way to overcome the lack of labeled information and that all the views do not have the same strength (or do not bring information in the same way) as the learning of weights, as it is done in MωMvC^2, is much more effective than the uniform combination of view-specific classifiers as it is done in MV-MV.

We also analyze the behavior of the algorithms for growing initial amounts of labeled data. Figure 2 illustrates this by showing the evolution of the accuracy and the F_1-score with respect to the number of labeled examples in the initial labeled training sets on MNIST$_1$, MNIST$_2$ and Reuters datasets. As expected, all performance curves increase monotonically (except MVMLsp) *w.r.t* the addi-

tional labeled data. When there are sufficient labeled examples, the performance increase of all algorithms actually begins to slow, suggesting that the labeled data carries sufficient information and that the different views do not bring additional information.

An important point here is that `rBoost.SH`—which takes into account both view-consistency and diversity between views—provides the worst results on $\texttt{MNIST}_1$ where there is no overlapping between the views, while the weighted majority vote as it is performed in $\texttt{M}\omega\texttt{MvC}^2$ still provides an efficient model. Furthermore, `MVMLsp`—which learns multiview kernels to capture views-specific informations and relation between views—performs worst on all the datasets. We believe that the superior performance of our method stands in our two-level framework. Indeed, thanks to this trick, we are able to consider the view-specific information by learning weights over view-specific classifiers, and to capture the importance of each view in the final ensemble by learning weights over the views.

5 Conclusion

In this paper, we tackle the issue of classifier combination when observations have different representations (or have multiple views). Our approach jointly learns weighted majority vote view-specific classifiers (*i.e.* at the view level) over a set of base classifiers, and a second weighted majority vote classifier over the previous set of view specific weighted majority vote classifiers. We show that the minimization of the multiview classification error is equivalent to the minimization of Bregman divergences. This embedding allowed to derive a parallel-update optimization boosting-like algorithm to learn the weights of the double weighted multiview majority vote classifier. Our results show clearly that our method allows to reach high performance in terms of accuracy and F_1-score on three datasets in the situation where few initial labeled training documents are available. It also comes out that compared to the uniform combination of view-specific classifiers, the learning of weights allows to better capture the strengths of different views.

As future work, we would like to extend our algorithm to the *semi-supervised* case, where one has access to an additionally unlabeled set during the training. One possible way is to learn a view-specific classifier using pseudo-labels (for unlabeled data) generated from the classifiers trained from other views, *e.g.* [27]. Moreover, the question of extending our work to the case where all the views are not necessarily available or not complete (*missing views* or *incomplete views*, *e.g.* [1,26]), is very exciting. One solution could be to adapt the definition of the matrix $\mathbf{M}_v$ to allow to deal with incomplete data; this may be done by considering a notion of diversity to complete $\mathbf{M}_v$.

Acknowledgment. This work is partially funded by the French ANR project LIVES ANR-15-CE23-0026-03 and the "Région Rhône-Alpes".

References

1. Amini, M.R., Usunier, N., Goutte, C.: Learning from multiple partially observed views - an application to multilingual text categorization. In: NIPS (2009)
2. Bertsekas, D.P.: Nonlinear Programming. Athena Scientific, Belmont (1999)
3. Blum, A., Mitchell, T.M.: Combining labeled and unlabeled data with co-training. In: COLT, pp. 92–100 (1998)
4. Bregman, L.: The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. USSR Comput. Math. Math. Phys. **7**(3), 200–217 (1967)
5. Chen, M., Denoyer, L.: Multi-view generative adversarial networks. In: ECML-PKDD, pp. 175–188 (2017)
6. Collins, M., Schapire, R.E., Singer, Y.: Logistic regression, adaboost and bregman distances. Mach. Learn. **48**(1–3), 253–285 (2002)
7. Darroch, J.N., Ratcliff, D.: Generalized iterative scaling for log-linear models. Ann. Math. Stat. **43**, 1470–1480 (1972)
8. Della Pietra, S., Della Pietra, V., Lafferty, J.: Inducing features of random fields. IEEE TPAMI **19**(4), 380–393 (1997)
9. Farquhar, J., Hardoon, D., Meng, H., Shawe-taylor, J.S., Szedmák, S.: Two view learning: Svm-2k, theory and practice. In: NIPS, pp. 355–362 (2006)
10. Gönen, M., Alpayd, E.: Multiple kernel learning algorithms. JMLR **12**, 2211–2268 (2011)
11. Huusari, R., Kadri, H., Capponi, C.: Multi-view metric learning in vector-valued kernel spaces. In: AISTATS (2018)
12. Janodet, J.C., Sebban, M., Suchier, H.M.: Boosting classifiers built from different subsets of features. Fundam. Inf. **94**(2009), 1–21 (2009)
13. Koço, S., Capponi, C.: A boosting approach to multiview classification with cooperation. In: Gunopulos, D., Hofmann, T., Malerba, D., Vazirgiannis, M. (eds.) ECML PKDD 2011. LNCS (LNAI), vol. 6912, pp. 209–228. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-23783-6_14
14. Lafferty, J.: Additive models, boosting, and inference for generalized divergences. In: COLT, pp. 125–133 (1999)
15. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. In: Proceedings of the IEEE, pp. 2278–2324 (1998)
16. Pedregosa, F., et al.: Scikit-learn: machine learning in python. JMLR **12**, 2825–2830 (2011)
17. Peng, J., Aved, A.J., Seetharaman, G., Palaniappan, K.: Multiview boosting with information propagation for classification. IEEE Trans. Neural Netw. Learn. Syst. **99**, 1–13 (2017)
18. Peng, J., Barbu, C., Seetharaman, G., Fan, W., Wu, X., Palaniappan, K.: Shareboost: boosting for multi-view learning with performance guarantees. In: ECML-PKDD, pp. 597–612 (2011)
19. Powers, D.M.: Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. J. Mach. Learn. Technol. **1**(2), 37–63 (2011)
20. Sindhwani, V., Rosenberg, D.S.: An RKHS for multi-view learning and manifold co-regularization. In: ICML, pp. 976–983 (2008)
21. Snoek, C., Worring, M., Smeulders, A.W.M.: Early versus late fusion in semantic video analysis. In: ACM Multimedia, pp. 399–402 (2005)
22. Sun, S.: A survey of multi-view machine learning. Neural Comput. Appl. **23**(7–8), 2031–2038 (2013)

23. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer, Berlin (1999)
24. Xiao, M., Guo, Y.: Multi-view Adaboost for multilingual subjectivity analysis. In: COLING, pp. 2851–2866 (2012)
25. Xu, C., Tao, D., Xu, C.: Large-margin multi-viewinformation bottleneck. IEEE TPAMI **36**(8), 1559–1572 (2014)
26. Xu, C., Tao, D., Xu, C.: Multi-view learning with incomplete views. IEEE Trans. Image Process. **24**(12), 5812–5825 (2015)
27. Xu, X., Li, W., Xu, D., Tsang, I.W.: Co-labeling for multi-view weakly labeled learning. IEEE TPAMI **38**(6), 1113–1125 (2016)
28. Xu, Z., Sun, S.: An algorithm on multi-view Adaboost. In: Wong, K.W., Mendis, B.S.U., Bouzerdoum, A. (eds.) ICONIP 2010. LNCS, vol. 6443, pp. 355–362. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17537-4_44
29. Zhang, J., Zhang, D.: A novel ensemble construction method for multi-view data using random cross-view correlation between within-class examples. Pattern. Recogn. **44**(6), 1162–1171 (2011)