A PSO algorithm for Improving Multi-View Classification

Zilton Cordeiro Junior
Department of Computer Science
Universidade Federal de Minas Gerais
Belo Horizonte, Brazil
Email: zilton@dcc.ufmg.br

Gisele L. Pappa
Department of Computer Science
Universidade Federal de Minas Gerais
Belo Horizonte, Brazil
Email: glpappa@dcc.ufmg.br

***Abstract*—The Multi-view or multi-modality learning approach is becoming popular for providing different representations of a problem from which classifiers can learn from. Examples of these representations are, for instance, sound and image for the case of the video classification problem. The main idea behind multi-view learning is that learning from these representations separately can lead to better gains than merging them into a single dataset. In the same way as ensembles combine results from different classifiers, the outputs given by classifiers in different views have to be combined in order to provide a final class for an example. This paper proposes a PSO algorithm to combine the outputs coming from different views. It also considers that some views may be better at classifying specific classes, and provides weighting schemes for both views and classes. Experiments were performed in two datasets with three views each, and compared with all views in a single dataset, a majority voting scheme and a scheme based on the Dempster-Shafer theory. Experimental results show that the PSO obtains statistically better results than the other approaches evaluated.**

I. Introduction

Multiple classifier systems (MCS) are a simple and effective approach for learning [31]. Proof of this is that ensembles are nowadays among the state of the art classifiers [12], thanks to the idea of combining different classifiers and data samples in a variety of ways. In the same direction of MCS, multi-view learning [9] or multi-modality learning [30] is becoming popular by providing different representations of the problem from which classifiers can learn from.

The main idea behind multi-view learning (MVL) is that a problem can be characterized using different representations (views), and that learning from these representations separately can lead to better gains than merging them into a single dataset. For instance, consider the application of video classification illustrated in Figure 1. Classification can be performed using at least three representations: audio, subtitles and image. The multi-view principle states that learning an audio model, a subtitle model, an image model and then combining them is more effective than generating a single model considering all attributes at once [25].

Hence, while many MCS systems divide the dataset horizontally, i.e., they vary the instances in the dataset but preserve the attributes, multi-view learning divides de problem vertically, i.e., the instances remain the same in all views, but the attributes change. Note that, in contrast with MCS, in MVL a single classification model is produced per view, using an adequate classification algorithm.

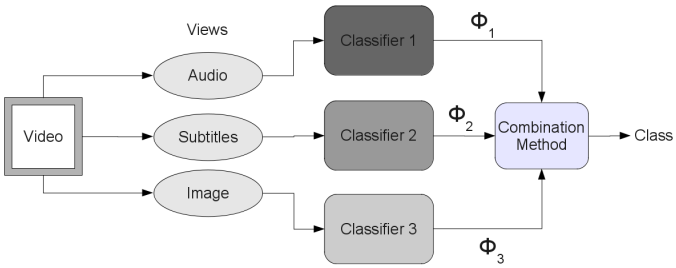

Fig. 1. A multi-view supervised learning approach for classification.

As in MCS, when classifying a new example in MVL, the outputs ϕ given by different classifiers trained in different views have to be combined, and provide a class to the new example. This combination can be made using a variety of methods from MCS [10], including simple approaches such as the majority vote, or more sophisticated ones, such as the Dempster-Shafer theory of evidence [14]. This paper proposes a different direction: to model a particle swarm algorithm (PSO) [7] to weight outputs coming from different views. Furthermore, the method also takes into account another issue: certain views might be better at predicting a specific class than others, and hence should be more trusted. This fact is addressed by weighting classes as well as views.

The PSO algorithm was chosen because it is well-known as a powerful algorithm for optimizing continuous real values [7], [11]. It has been successful applied in a diverse range of "weighting problems", such as estimating the connection weights of artificial neural network [11] or the weight of attributes in a dataset [34].

Using multi-view learning brings clear advantages over traditional single-view learning. First, the data dimensionality classifiers have to deal with is automatically reduced, since n attributes are divided into a set of v views. Second, different classification algorithms may be used to learn from different views, allowing the user to select the one that best suits a view according to its characteristics. Moreover, complex problems in many application areas, such as gene function annotation in bioinformatics and video summarization have shown to greatly

benefit from techniques considering more than one type of data [13]. Finally, different information about the same example can also help solve other machine learning problems, such as transfer learning [1] and semi-supervised learning [4].

Results in two multi-view datasets containing three views each, one regarding text classification and the other spammers detection, showed that the PSO obtains statistically better predictions than a majority voting scheme and a method based on the Dempster-Shafer theory.

The remainder of this paper is organized as follows. Section II describes the theory behind multi-view learning and related work. Section III presents the basic concepts of PSO and explains how it was modeled to provide views and classes combination. Section IV reports computational results. Finally, Section V draws some conclusions and discusses future work.

II. Background

As this work presents a PSO algorithm for combining outputs coming from different classifiers learnt from different views, we first review the theory behind MVL, and then present a few works where the PSO was used in the classification task, including feature weighting and ensemble weighting.

A. Multi-View Learning

The problem of multi-view learning is defined as follows. Given a domain with v views, where each labeled instance represented by a tuple $\langle\ x_1, x_2, ..., x_v, l\ \rangle$, where l is a label and $[x_1], [x_2], ..., [x_v]$ are sets of data in v views. For each view, one classification model is produced, requiring the a posteriori combination of these models to predict the classes of new instances, as showed in Figure 1.

The theoretical foundations of multi-view learning come from the assumption that the defined views are compatible and non-correlated [22]. A problem has compatible views if all examples are labeled identically in each view. The views are non-correlated if, given the label of any example, their descriptions in each view are independent. Hence, for views to be compatible and non-correlated, in any example $[\ x_1, x_2, l\ ]$, x_1 and x_2 are independent, given the label l. Even though these assumptions have been originated from semi-supervised problems, they can be easily extrapolated to the context of supervised learning, as shown in [33].

There have been many research efforts in multi-view learning [19] [24] [30] [35], with most studies concentrating on a semi-supervised approach. We will not go into details about these methods, but instead we will explore the combination strategies they use. The great majority of MVL algorithms work with a few number of views - usually two, and hence their combination methods are usually straightforward, such as the use of the maximum confidence between all views [19]. In cases with a greater number of views, using a simple majority vote scheme, which chooses the class of an instance based on the the class predicted by the majority of the classifiers, is traditional. However, in more sophisticated scenarios, views can be also combined using other combination methods, such as those used to combine ensemble of classifiers, like the Dempster-Shafer theory of evidence [3] or the borda count [26]. Here we briefly described the former, since it will be used as one of the baseline methods for the proposed PSO.

The Dempster-Shafer theory of evidence [29] is a tool for representing uncertain knowledge, and is often viewed as a generalization of the Bayesian probability theory. It provides a coherent representation for ignorance (lack of evidence) and discards the insufficient reasoning principle. It is a method commonly used in decision making processes, and represents evidences in terms of evidential functions (such as mass functions) and ignorance. For more details about this method, the reader is referred to [14].

B. PSO

Several works have used a PSO to solve problems in the context of supervised learning [8] [21] [36]. In [16], for example, a discrete PSO for inducing rules from data was proposed, while in [17] the PSO was used to evolve the weights of a neural network. In the context of neural networks, [27] also proposed a dynamic PSO that outperforms the standard back propagation algorithm. Moreover, the PSO was also explored for weighting attributes in learning problems [34] [37]. In [37], the PSO not to only search weights for features but also to find hyper-parameters of a least-squares support vector machine (LS-SVM).

In the context of weighting classifiers, [23] presented a PSO to compute weights for combining multiple neural network classifiers. The weights were obtained so that they minimize the total classification error rate of the ensemble system. In [15] we found the most similar work to ours. [15] modeled a PSO to weight the majority vote (WMV) of an ensemble of classifiers. Apart from working with MVL instead of ensembles, the strategy adopted here differs from ours in two main points: (i) [15] weights votes, we weight the classifier outputs before performing the voting process, (ii) the weights generated by the PSO can also take into account the classes in each view, while [15] weights only the outputs of the predicted class. Hence, the PSO can enforce or discourage a prediction if that classifier is not the best in that view.

III. A PSO algorithm for Combining Classification Views

Eberhart e *Kennedy* proposed the Particle Swarm Optimization (PSO) [7] method inspired by the social behavior of individuals, which can be summarize with three principles: (i) evaluate, (ii) compare, and (iii) imitate. In other words, the basic idea of the algorithm is that we learn from our own experience, but also by mimicking the behavior of the most successful people.

As its name suggests, the PSO algorithm works with a set of particles moving into the search space for a pre-defined number of iterations. Each particle represents a solution to the problem at hand, and is described by a position $X_i = (X_{i_1}, X_{i_2}, ..., X_{i_d})$ in space and a velocity, represented by $V_i = (V_{i_1}, V_{i_2}, ..., V_{i_d})$. Each X_{i_d} represents a dimension of the solution, and is associated with a V_{i_d}, responsible for

moving the particle in the search space. A particle also has memory, and knows its best position along the iterations and the best position of the best particle in the swarm so far.

Particles are initially randomly placed in the search space and, at each iteration, evaluated according to a pre-defined fitness function, which assesses the quality of the solution represented by the particle. Following evaluation, particles move into the space by changing their velocity and position. Particle movements take memory into account, considering the particles confidence on their best position so far, and their confidence in the swarm's best position up to the current iteration.

Following this basic principle, here we propose a PSO algorithm for weighting classifications coming from different data views, from now on referred as PSO-WV. In a second step, the PSO is modified to also provide weights for classes in specific views, by estimating how good a view is to predict a class, generating PSO-WC. The next sections describe the three main components of the PSO modified to implement PSO-WV and PSO-WC: (i) particle representation, (ii) fitness function, and (iii) the mechanisms for velocity and position update. Finally, we show how the final vector of weights is used to determine the class of a new test example. Algorithm 23 underlines the overall procedure.

Algoritmo 1: PSO for combining multi-view classification

```
1  foreach Train f do
2      Generate random initial population
3      Applies weights to the outputs
4      Calculate the fitness of all particles /* Eq. 1 */
5      foreach Iteration i do
6          Update global information of the swarm (gbest)
7          Update the social factor /* Eq. 5 */
8          Update the individual factor /* Eq. 6 */
9          foreach Particle p do
10             foreach View v do
11                 if PSO-WV then
12                     Update the velocity of the position v
                       /* Eq. 4 */
13                     Update the position v
14                     Applies weights W_v to the outputs φ_v
15                 else if PSO-WC then
16                     foreach Class m do
17                         Update the velocity of the
                           position v_m /* Eq. 4 */
18                         Update the position v_m
19                         Applies weights W_{v_m} to the
                           outputs φ_{v_m}
20             Calculate the fitness of p /* Eq. 1 */
21             Update local information of p (pbest)
22 return W_{gbest_f}
23 Calculate fitness in the test using the weights of gbest
```

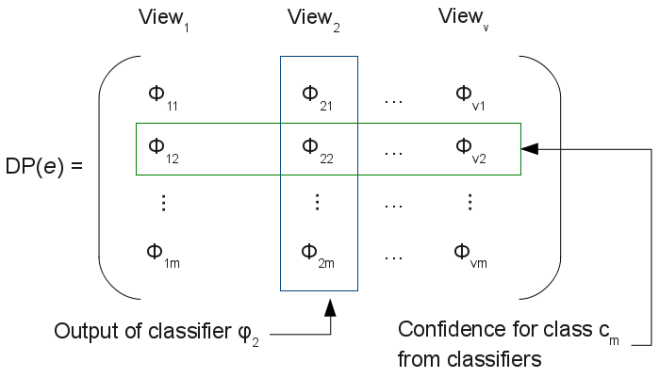

Fig. 2. Decision Profile for a given instance e.

A. Particle Representation

In both versions of the PSO, particles represent a set of weights. PSO-WV has v dimensions, where v corresponds to the number of views in the dataset, and each dimension is a positive real value that represents a weight W_i of a view i. In PSO-WC, in contrast, the number of dimensions is equals to the number of views v $\times$ the number m of classes in the dataset. Hence, each particle's dimension represents a weight W_{ij} to class j in view i. In both versions of the PSO, the sum of the values of all dimensions adds to one.

In order to combine v views (classifiers outputs) in m classes, the outputs of classifier φ_v on instance e are organized into a decision profile (DP) [18], illustrated in Figure 2. In the DP, each column represents a view (classifier), each line represents a class, and $\phi_{v1} > \phi_{v2} > ... > \phi_{vm}$ for all values of v. ϕ_{vm} is the confidence of the classifier φ_v in assigning instance e to class c_m. ϕ values are in the interval $[0, 1]$, that is, $\varphi_v : R^n \rightarrow [0, 1]^c$.

The role of the weights W_i and W_{ij} is to modify the confidence ϕ_v of the classifiers φ_v in predicting the class of an instance e according to this profile. By confidence we mean any metric that assesses the "quality" of a prediction. NaïBayes, for example, returns a list of probabilities of the new example to belong to each class in the database. When weighting views, the weight W_i will modify the probability value of the winner class (the biggest returned probability will correspond to ϕ), while W_{ij} will weight, for each view, each probability value in the returned probability list. More details are given in Section III-D.

B. Fitness function

After the particles are generated, they are evaluated according to their capability of combining classifiers outputs in a way that the predicted class is correct. We evaluate particles using the micro-average f1 ($\mu f1$), as defined in Eq. 1.

$\mu f1$ is a standard metric for evaluating classifiers, and was derived from the f1 metric, which is the harmonic mean of precision and recall. However, it weights the f1 of each class based on the representativeness of the class in the database i.e., according to the number of examples in each class. Eqs. 2 and 3 define the micro-precision (μPr) and micro-recall (μPr), for

a problem with m classes. TP_i corresponds to the number of examples of class i correctly classified, FP_i corresponds to the number of examples in classes different from i but classified as i, and FN_i corresponds to the number of examples in class i wrongly classified.

$$\mu f1 = \frac{2 \times \mu Pr \times \mu Re}{\mu Pr + \mu Re} \tag{1}$$

$$\mu Pr = \frac{\sum_{i=1}^{m} TP_i}{\sum_{i=1}^{m} (TP_i + FP_i)} \tag{2}$$

$$\mu Re = \frac{\sum_{i=1}^{m} TP_i}{\sum_{i=1}^{m} (TP_i + FN_i)} \tag{3}$$

Note that the values of $\mu f1$ are calculated after modifying the output values of the classifiers being combined, as detailed in Section III-D.

C. Velocity and Position Update

Following fitness evaluation, particles are allowed to move into the search space by changing their velocity and position. Particle movements take into account two factors [7], as observed in Eq. 4:

1) The individual factor, represented by the particle's attraction to its best position so far ($pbest$, or particle best).
2) The social factor, represented by the particle's attraction to the best swarm particle position so far ($gbest$, or global best).

According to Eq. 4, the velocity is updated based on the aforementioned factors and the particle's current position. As observed, the individual and social factors are reflected in Eq. 4 by φ_1 and φ_2, which are constants that have their values updated at each iteration according to Eqs. 5 and 6. They represent how much the particle trusts itself and its neighbors, and their values are in the interval $\in$ [0,1].

According to Eqs. 5 and 6, φ_1 decreases and φ_2 increases their values at each iteration, making particles to increasingly trust the swarm more than they trust themselves. In these equations, C_{min} and C_{max} are constants, $maxIt$ is the maximum number of iterations, and it is the current iteration.

$$V_i = V_i + \varphi \times \left[\frac{(\varphi_1 \times pbest) + (\varphi_2 \times gbest)}{\varphi_1 + \varphi_2} - X_i \right] \tag{4}$$

$$\varphi_1 = C_{max} - \left[\frac{(C_{max} - C_{min}) \times it}{maxIt} \right] \tag{5}$$

$$\varphi_2 = C_{min} + \left[\frac{(C_{max} - C_{min}) \times it}{maxIt} \right] \tag{6}$$

After the velocity is updated, the new position X is given by $X_i = X_i + V_i$.

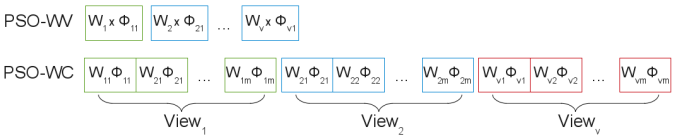

Fig. 3. Particles "Decodification": how they use the decision profile information to classify new examples.

D. Class Decision

After the PSO is run for $maxIt$ iterations, we use the best particle found to set the classes of test examples. Figure 3 shows how the weights of the particles are applied in each version of the PSO. Observe that these figures do not show the particles, which search only for the weights, but the final product of the weight by the classifiers confidence, illustrated in the decision profile showed in Figure 2. The decision making processes of PSO-WV and PSO-WC are formally defined in Eqs. 7 and 8, respectively. Recall that v is the number of views, m is the number of classes, ϕ the classifiers output, and W the PSO weights.

$$cl_j = argmax_v(W_v \times argmax_m(\phi_{vm})) \tag{7}$$

$$cl_j = argmax_v(argmax_m(W_{vm} \times \phi_{vm})) \tag{8}$$

It is interesting to point out that the PSO algorithm can weight outputs coming from distinct classification algorithms or not. In the case of dealing with outputs coming from different classification algorithms, e.g., Naïve Bayes and KNN, the outputs are normalized for each classifier.

IV. Experimental Results

This section presents the results obtained by PSO-WV and PSO-WC in two datasets from different application domains: the ACM-DL[1] (Digital Library of the Association for Computing Machinery), which deals with document classification, and an Online Video Social Network database [2], where each instance represents a YouTube user. Their main characteristics are:

1) ACM-DL: it is a subset of ACM-DL, and contains 6.681 documents divided in 8 classes, distributed as: 702 A, 658 B, 1848 C, 420 D, 197 E, 1554 F, 1065 G, 237 H. From the original ACM-DL were extracted three views: (i) **Terms**, which corresponds to the terms present in the ACM papers abstracts; (ii) **Co-authorship network**, which represents relationships of co-authorship among authors, and (iii) **Citation network**, which shows which papers cite which.
2) YouTube: contains information about 829 YouTube users and is divided in 3 classes, distributed as: 641 legitimate users, 157 spammers and 31 promoters. It also has three views: (i) **Video**, which captures specific properties of the videos uploaded by each user; (ii) **Network**, which

[1] http://portal.acm.org/dl.cfm

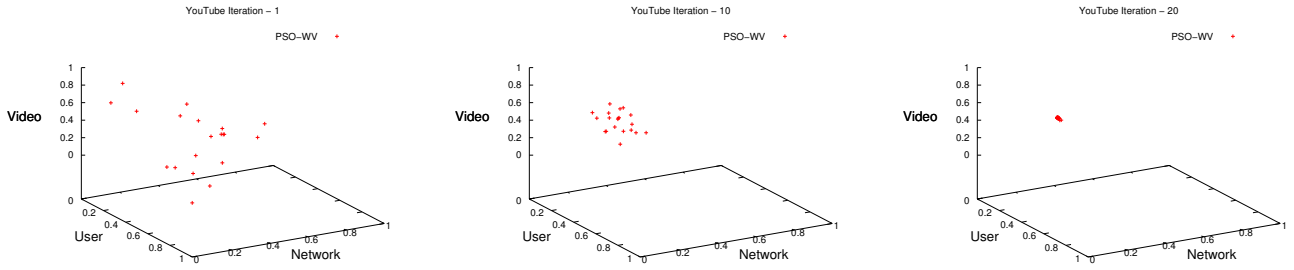

Fig. 4. Swarm Movements for the PSO-WV in the YouTube dataset.

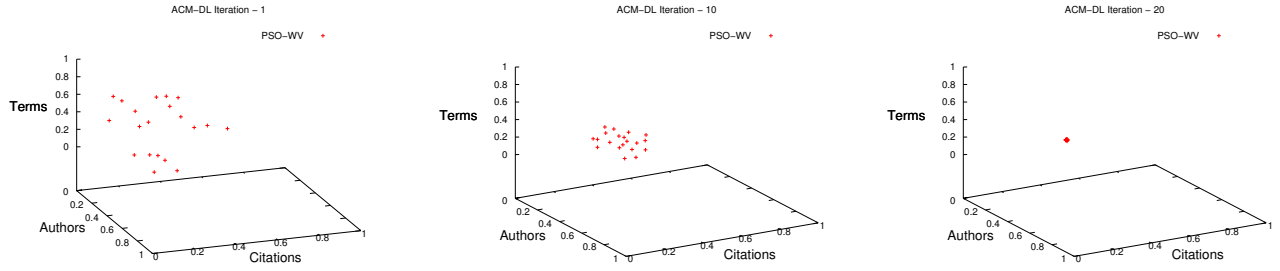

Fig. 5. Swarm Movements for PSO-WV in the ACM-DL dataset.

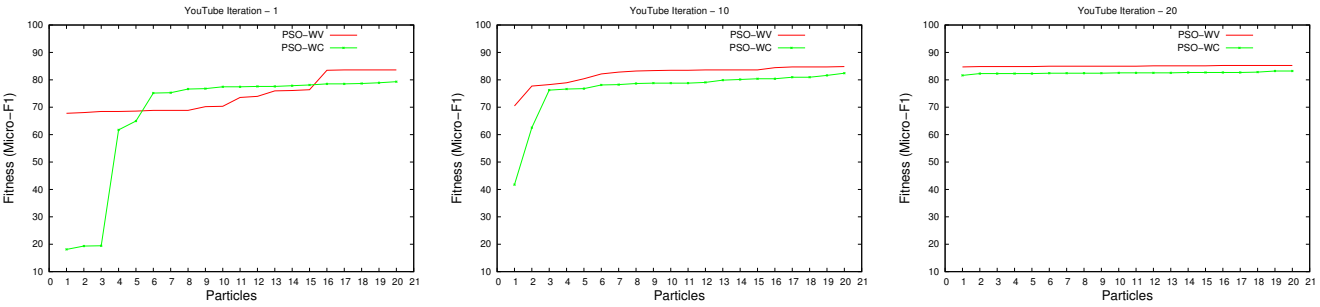

Fig. 6. Fitness evolution of PSO-WV and PSO-WC in the YouTube dataset.

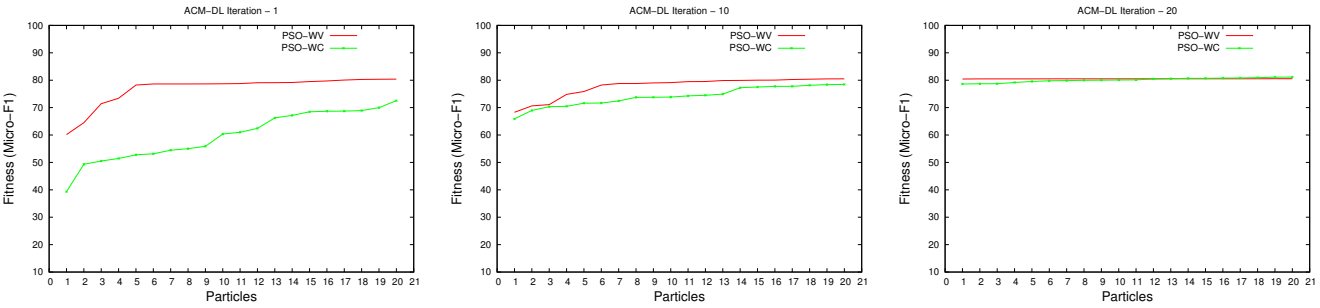

Fig. 7. Fitness evolution of PSO-WV and PSO-WC in the ACM-DL dataset.

models the social relationships established between users from video response interactions; and (iii) **User**, which presents characteristics of user behavior.

Experiments were divided in three phases. In the first phase, four classifiers, namely KNN (K-Nearest Neighbor) [5], Naïve Bayes [20], Rocchio [28] and SVM-Perf [32], were executed independently in the views of the two datasets, and also in a merged version of the three views (i.e., all views in a single file). In a second phase, the proposed PSOs were used to combine the views using the same classifier for all views. For example, SVM was used to classify the three views of the Youtube dataset: video, network and user views In this

TABLE I
RESULTS OF MICRO-F1 FOR ACM-DL.

	Views			Combining Methods				
Algorithms	**Authors**	**Terms**	**Citations**	**A + T + C**	**Maj. Vote**	**DS**	**PSO-WV**	**PSO-WC**
KNN	57.78	**78.79**	33.9	80.77•	79.13▽	78.56▽	81.22△	79.85
Naïve Bayes	**57.30**	78.73	13.44	68.63▽	79.88•	80.19•	80.72•	80.45•
Rocchio	49.69	73.54	18.47	74.63▽	75.48▽	59.19▽	76.62△	75.29
SVM	58.70	78.31	**32.03**	79.09•	60.22▽	66.62▽	79.15△	59.20
Heterogeneous	**Naïve Bayes**	**KNN**	**SVM**	—	59.46▽	77.47▽	80.77	**81.96**△

TABLE II
RESULTS OF MICRO-F1 FOR YOUTUBE.

	Views			Combining Methods				
Algorithms	**Network**	**User**	**Video**	**N + U + V**	**Maj. Vote**	**DS**	**PSO-WV**	**PSO-WC**
KNN	77.57	80.82	**84.08**	84.92▽	81.91▽	81.76▽	83.96	**86.37**△
Naïve Bayes	66.71	58.87	43.9	40.54▽	66.67▽	64.98▽	66.10	74.18△
Rocchio	48.38	**68.17**	75.04	72.26▽	69.13▽	54.11▽	73.35△	70.69
SVM	**69.71**	72.37	72.63	47.75▽	69.12▽	71.50▽	76.12△	74.43
Heterogeneous	**SVM**	**Rocchio**	**KNN**	—	77.93▽	66.79▽	84.56△	83.71

TABLE III
WEIGHTS FOR THE VIEWS IN THE DATASETS ACM-DL AND YOUTUBE.

Classifiers	Views					
	ACM-DL			YouTube		
	Authors	Terms	Citations	Network	User	Video
KNN	0.266029	0.471674	0.262297	0.467758	0.206725	0.325516
Naïve Bayes	0.509747	0.347285	0.142968	0.364758	0.370820	0.264423
Rocchio	0.190283	0.560280	0.249437	0.031204	0.479448	0.489348
SVM	0.356605	0332062	0.311333	0.293970	0.371428	0.334602
Heterogeneous	0.320916	0.298571	0.380513	0.258022	0.514498	0.227480

phase, the best classifiers for each view were identified, and then combined by the PSOs in a third experiment.

As the PSO requires a number of parameters to be set, in preliminary experiments we varied the number of particles and iterations. Although these values are not optimized, the better results found used 20 particles searching the space of solutions for 20 iterations. The values of C_{min} and C_{max}, which appear in Eqs. 5 e 6, were set to 0.5 and 4.0. These values were chosen based on [7], which shows that they make the particles traverse the search space systematically.

Figures 4 and 5 show the swarm movements for PSO-WV in both datasets when using the best classifiers identified for each view. We are not showing the PSO-WC because it has a bigger number of dimensions (3 x 8 for ACM-DL and 3 X 3 for YouTube). Each dimension in the graph corresponds to a view, and each point illustrates the position of the particle (the weights assigned to each view in each dimension) in the first, tenth, and twentieth iterations. For both datasets, we observe that initially the particles are scattered in the search space, and with time they start following the best position, as it should be. Notice that, at the end of 20 iterations, for YouTube the video and network views are considered more important than the users view. For ACM, citations and terms were considered more relevant.

Figures 6 and 7 show the values of fitness of the particles over time, also using the best classifiers identified for each view. Recall that during the process of fitness calculation, the classifiers are executed and their values weighted by the PSO to generate the final classification. First, notice that PSO-WV obtained better results for YouTube while PSO-WC obtained better results for ACM-DL (these results can be also observed in Tables I and II, as discussed later). This can be explained by the fact that, as ACM-DL has eight classes, a change in the ranking of classes due to the PSO weights might have more impact than a change in the ranking of three classes, as in YouTube.

Tables I and II show the results of micro-f1 obtained in the experiments. The first column presents the algorithms executed, followed by the results of the three views independently. The next three columns show the results of executing all views together, followed by other two methods we are going to compare the PSO with: a majority voting scheme and the Dempster-Shafer theory of evidence. The last two columns bring the results of PSO-WV and PSO-WC.

All figures are based on a 10-fold cross-validation [6], and results were compared using a Student t-test with 95% confidence. The results of the tests are indicated in the tables in two ways. We first compare the results of PSO-WV and PSO-WC. PSO cells are then marked either with a (•), indicating that the results are statistically equivalent, or with a (△), indicating that the results in the cell marked is statistically better than the one in the neighbor cell. In a second analysis, we compare the PSO marked in this first analysis with all the other combination methods, marking cells labeled "A+C+T", "Maj. Vote" and "DS". In this case, a third notation is introduced. (▽) indicates that a result is statistically worse than the one obtained by the best PSO.

Let us first analyze the ACM results (Table I). In this case, the results for the PSO-WV are statistically better than PSO-WC in three cases: the homogeneous combinations of KNN, Rocchio and SVM, and statistically equivalent to the results obtained by Naïve Bayes. However, when combining the best methods in each view (showed in bold), the PSO-WC provides the best results overall, with a micro-f1 of 81.96%. This shows a clear advantage of multi-view: the use of different classifiers for different sets of attributes that, given their characteristics, are more suited to one classifier than another. Throughout Table I, we observe that the view of Terms is the best and the Citations the worst when considered independently. However, we have to take into account that the views of Authors and Citations are very sparse.

For YouTube (Table II), PSO-WC is better than PSO-WV in two cases, namely KNN and Naïve Bayes, while in the other three PSO-WV is superior. However, the better results overall are obtained by PSO-WC with KNN. In contrast with ACM-DL, here using only KNN in separate views we obtained better results (86.37%) than using KNN in a single dataset (84.92%). In conclusion, in both datasets the best results were obtained by the PSO-WC, which weights not only the views but also the classes. This approach takes advantage of identifying which methods and views are better at predicting which classes.

A last analysis based on the weights can give some insights on what the PSO has done. As the PSO-WC has too many weights to report here, we discuss the PSO-WV weights. Observing the results in Table III, we notice that, for ACM-DL, the Citations view receives, most of the time, a lower weight than the other views. The terms view was considered by Rocchio twice better than the other two. This is probably due to the way the algorithm works, which will suffer with sparse data. For Naïve Bayes, in contrast, half of the weight should be given to the authors view, followed by the terms. The SVM was the only classifier that gave all views very similar values.

Now let us go to a different analysis. If the views of Terms for ACM-DL and Video for YouTube, when run independently, always obtained the best results, why does the PSO did not assign the greatest weight for them? Because the PSO identifies the contribution of each view to the classification task, and tunes this contribution by the value it gives to each view. Similarly, the PSO-WV with heterogenous classifiers gives the Video view the smallest weight among the three views. When the three views are considered in isolation, this is the classifier/view with the best results.

For YouTube, it is interesting to notice the contrast of weights among different classifiers. For instance, while the network is the most important view for KNN, it is disregarded by Rocchio, which did not perform well in this view. Here, Naïve Bayes is the classifier that best balances the weights in the views, and it represents the worst results for PSO-WV.

These results show us that all views have an important role for the classification task, and the the PSO is an effective way of combining the results provided by each classifier in each view. It also shows that weighting classes can lead to gains when the views that are better at recognizing examples of one class receive more attention than the others.

V. Conclusions and Future Work

This paper presented a particle swarm optimization (PSO) algorithm for weighting views in multi-view classification applications. Two versions of the algorithm were introduced. The first was designed to combine the classifiers outputs coming from different views. The second considered, along with the views, how good they were at predicting examples in each class.

Both approaches (PSO-WV and PSO-WC) were experimentally assessed in two datasets with three views each, namely ACM-DL and YouTube. Four different classification algorithms, with different characteristics, were run in each view separately, and then combined with the PSO. The results were compared with the use of all views in a single dataset, a majority voting scheme, and a method based on the Dempster-Shafer theory, showing that the PSO obtains results statistically superior to these methods. Overall, the PSO-WC obtained the most results, showing the importance of weighting classes.

As future works, we are interested in applying this approach to other datasets, such as those obtained from social networks, where the presence of different views is quite intuitive. We will also explore other natural computing algorithms, such as genetic programming, to solve this same problem, since they seem like a good alternative.

Another interesting subject is how to ensure that a given view is the best for the problem being solved. In other words, what is the best method to generate views? Their intuitive separation in the real world is enough? Finally, we also want to test the current approach under a semi-supervised framework.

Acknowledgments

This work was partially supported by CNPq, CAPES, Fapemig, and InWeb - Brazilian National Institute of Science and Technology for the Web.

References

[1] J. Baxter. Theoretical models of learning to learn. pages 71–94, 1998.

[2] F. Benevenuto, T. Rodrigues, V. Almeida, J. Almeida, and M. Gonalves. Detecting spammers and content promoters in online video social networks. In *Proc. of Int'l ACM SIGIR*, Boston, MA, USA, July 2009.

[3] Y. Bi, S. McClean, and T. Anderson. On combining multiple classifiers using an evidential approach. In *AAAI'06: Proceedings of the 21st national conference on Artificial intelligence*, pages 324–329. AAAI Press, 2006.

[4] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proc. of the 11th Annual Conf. on Computational Learning theory*, pages 92–100, 1998.

[5] D. Bremner, E. Demaine, J. Erickson, J. Iacono, S. Langerman, P. Morin, and G. Toussaint. Output-sensitive algorithms for computing nearest-neighbour decision boundaries. *Discrete Comput. Geom.*, 33:593–604, April 2005.

[6] M. W. Browne. Cross-validation methods. *Journal of Mathematical Psychology*, 44(1):108 – 132, 2000.

[7] E. R. C., S. Yuhui, and K. James. *Swarm Intelligence* . Morgan Kaufmann, 2001.

[8] Q. Cao and Y. Liu. A knn classifier with pso feature weight learning ensemble. In *Intelligent Control and Information Processing (ICICIP), 2010 International Conference on*, pages 110 –114, 2010.

[9] M. Culp and G. Michailidis. A co-training algorithm for multi-view data with applications in data fusion. *Journal of chemometrics*, 23:294–303, 2009.

[10] T. Dietterich. Ensemble methods in machine learning. In *Multiple Classifier Systems*, volume 1857 of *Lecture Notes in Computer Science*, pages 1–15. Springer Berlin / Heidelberg, 2000.

[11] J.-X. Du, D.-S. Huang, and Z.-F. Wang. Pattern classification with a pso optimization based elliptical basis function neural networks. In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, pages 1654 –1661, 2007.

[12] Y. Freund and R. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational Learning Theory*, volume 904 of *Lecture Notes in Computer Science*, pages 23–37. Springer Berlin / Heidelberg, 1995.

[13] W. Fujibuchi and T. Kato. Classification of heterogeneous microarray data by maximum entropy kernel. *BMC Bioinformatics*, 8:267+, July 2007.

[14] V.-N. Huynh, T. T. Nguyen, and C. A. Le. Adaptively entropy-based weighting classifiers in combination using dempster-shafer theory for word sense disambiguation. *Comput. Speech Lang.*, 24:461–473, July 2010.

[15] A. Kausar, M. Ishtiaq, M. A. Jaffar, and A. M. Mirza. Optimization of ensemble based decision using pso. In *Proceedings of the World Congress on Engineering*, WCE '10, 2010.

[16] N. Khan, A. Rauf Baig, and M. Iqbal. A new discrete pso for data classification. In *Information Science and Applications (ICISA), 2010 International Conference on*, pages 1 –6, 2010.

[17] A. Khurshid and A. Gokhale. Classification system for digital signal types using neuro fuzzy system and pso. In *Nature Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on*, pages 373 –378, 2009.

[18] L. Kuncheva. Decision templates for multiple classifier fusion: an experimental comparison. *Pattern Recognition*, 34(2):299–314, February 2001.

[19] S. Li, X. Peng, X. Hou, H. Zhang, and Q. Cheng. Multi-view face pose estimation based on supervised isa learning. In *Automatic Face and Gesture Recognition, 2002. Proceedings. Fifth IEEE International Conference on*, pages 100 –105, May 2002.

[20] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. In *Proc. of the 15th Nat. Conf. on Artificial Intelligence*, pages 41 – 48, 1998.

[21] A. Mohemmed, M. Johnston, and M. Zhang. Particle swarm optimization based multi-prototype ensembles. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, GECCO '09, pages 57–64, New York, NY, USA, 2009. ACM.

[22] I. M. Muslea, S. Minton, and C. A. Knoblock. Active + semi-supervised learning = robust multi-view learning. In *Proc. of 19th Int. Conf. on Machine Learning*, pages 435–442, 2002.

[23] S. Nabavi-Kerizi, M. Abadi, and E. Kabir. A pso-based weighting method for linear combination of neural networks. *Computers & Electrical Engineering*, 36(5):886–894, 2010. Advances in Computing Systems Science and Engineering.

[24] V. Ng and C. Cardie. Weakly supervised natural language learning without redundant views. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 94–101, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[25] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM. *Mach. Learn.*, 39(2-3):103–134, 2000.

[26] C. A. Perez, L. A. Cament, and L. E. Castillo. Methodological improvement on local gabor face recognition based on feature selection and enhanced borda count. *Pattern Recognition*, 44(4):951 – 963, 2011.

[27] A. Rakitianskaia and A. P. Engelbrecht. Training neural networks with pso in dynamic environments. In *Proceedings of the Eleventh conference on Congress on Evolutionary Computation*, CEC'09, pages 667–673, Piscataway, NJ, USA, 2009. IEEE Press.

[28] J. Rocchio. Relevance feedback in information retrieval. *The SMART retrieval system: experiments in automatic document processing*, pages 313–323, 1971.

[29] G. Shafer. *A mathematical theory of evidence*. Princeton university press, 1976.

[30] H. Tong, J. He, M. Li, C. Zhang, and W.-Y. Ma. Graph based multi-modality learning. In *MULTIMEDIA '05: Proc. of the 13th annual ACM international conference on Multimedia*, pages 862–871, New York, NY, USA, 2005. ACM.

[31] R. J. Urbanowicz and J. H. Moore. The application of michigan-style learning classifier systems to address genetic heterogeneity and epistasisin association studies. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, GECCO '10, pages 195–202, New York, NY, USA, 2010. ACM.

[32] V. N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag, 1995.

[33] Z. Wang, S. Chen, H. Xue, and Z. Pan. A novel regularization learning for single-view patterns: Multi-view discriminative regularization. *Neural Processing Letters*, 31:159–175, 2010.

[34] L. Xu, J.-H. Jiang, H.-L. Wu, G.-L. Shen, and R.-Q. Yu. Variable-weighted pls. *Chemometrics and Intelligent Laboratory Systems*, 85(1):140–143, 2007.

[35] S. Yu, B. Krishnapuram, R. Rosales, H. Steck, and B. R. Rao. Bayesian co-training. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1665–1672. MIT Press, Cambridge, MA, 2008.

[36] X. Zhou, J. Yuyu, R. Qi, F. Qian, and Z. Wang. Ipso: An immune based pso supervised learning system for incremental learning. In *Intelligent Control and Automation (WCICA), 2010 8th World Congress on*, pages 2707 –2711, 2010.

[37] H.-Y. Zou, H.-L. Wu, H.-Y. Fu, L.-J. Tang, L. Xu, J.-F. Nie, and R.-Q. Yu. Variable-weighted least-squares support vector machine for multivariate spectral analysis. *Talanta*, 80(5):1698–1701, 2010.