MI-Winnow: A New Multiple-Instance Learning Algorithm

Sharath R. Cholleti, Sally A. Goldman, and Rouhollah Rahmani
Department of Computer Science and Engineering
Washington University in St. Louis
cholleti, sg, rahmani@wustl.edu

Abstract

We present MI-Winnow, a new multiple-instance learning (MIL) algorithm that provides a new technique to convert MIL data into standard supervised data. In MIL each example is a collection (or bag*) of d-dimensional points where each dimension corresponds to a feature. A label is provided for the bag, but not for the individual points within the bag. MI-Winnow is different from existing multiple-instance learning algorithms in several key ways. First, MI-Winnow allows each image to be converted into a bag in multiple ways to create training (and test) data that varies in both the number of dimensions per point, and in the kind of features used. Second, instead of learning a concept defined by a single point-and-scaling hypothesis, MI-Winnow allows the underlying concept to be described by combining a set of separators learned by Winnow. For content-based image retrieval applications, such a generalized hypothesis is important since there may be different ways to recognize which images are of interest.*

1. Introduction

Content-Based Image Retrieval (CBIR) is the problem of retrieving semantically relevant images from an image database. One mechanism to get additional labeled images is through *relevance feedback*, where the user labels images from the results of an initial query as either desirable (positive) or not desirable (negative). We define the *training data* as a set of images labeled by the user.

The multiple-instance learning (MIL) model is a generalization of the standard supervised learning model that was originally proposed for the problem of drug discovery [5]. In MIL each example is a collection (or *bag*) of d-dimensional points where each dimension corresponds to a feature. The MIL model is well suited for content-based image retrieval(CBIR) since there is natural ambiguity as to what portion of each image is important to the user.

In the CBIR application, each bag corresponds to an image, and each point in the bag corresponds to a region of the image. Each region is represented by a feature vector. For example, if each region is represented by the average of each of the 3 color values then each point would be 3-dimensional. However, typically much richer feature representations are used. There has been significant research applying MIL to CBIR [12, 13, 21, 1, 3, 14, 2]. We are particularly interested in developing an MIL algorithm to use within an interactive CBIR system in which the user would only have to label a very small number of images ($\leq$20) in order to obtain good performance. This is very different from image categorization and other machine learning problems in which very large labeled data sets are available.

Many multiple-instance learning algorithms output a hypothesis that is a d-dimensional point and a d-dimensional scale vector defining a weighted Euclidean metric. In some cases a L_1 norm is used in which case the final hypothesis is a d-dimensional axis-parallel box. We refer to such a hypothesis as a point-and-scaling hypothesis (regardless if the L_1 or L_2 norm is used). For some application areas, a single point-and-scaling hypothesis has sufficient expressive power. However, in many CBIR tasks, one would expect several points (regions) in the bag (image) to be needed to explain why the image is desirable. These points may represent parts of a single complex object or several objects comprising a scene. Thus, it is important to study MIL when using a more complex hypothesis space than a single point-and-scaling hypothesis in order to have the expressive power to capture what makes an image desirable to a user.

We present a new MIL algorithm, MI-Winnow based upon Winnow [10], an on-line learning algorithm with provable bounds upon its performance. Winnow was designed to separate the relevant from irrelevant attributes. MI-Winnow proposes a new way to convert a MIL problem to a set of problems each of which can be solved using a standard supervised learning algorithm. There are several advantages of MI-Winnow over the EM-DD algorithm [20] and others like it that are commonly used for CBIR. First, the hypothesis returned by MI-Winnow is a set of d-dimensional boxes as opposed to returning a feature point and scale vector. By taking this approach, the difficult problem of finding a good

starting value for the scale factors is not needed. Second, by combining the predictions of a set of boxes, MI-Winnow is capable of capturing more complex concepts.

Another drawback of EM-DD is that given a test bag it cannot directly classify whether or not the bag is positive or negative. In order to perform classification of a bag, it is necessary to learn a threshold on the label between positive and negative. MI-Winnow can directly predict if a bag is positive or negative. We define the *bag generator* as the procedure used to convert the image to a bag. There are many different methods to generate a MIL representation for an image, and depending on the type of images, some can capture the target concept better than others. Unlike existing MIL algorithms, MI-Winnow allows any number of different bag generators to be applied to each image.

The remainder of this paper is organized as follows. Section 2 discusses related work. Section 3 discusses how each image is converted into a bag to create the training data for MI-Winnow. We review Winnow in Section 4. MI-Winnow is presented in Section 5. Experimental results are presented in Section 6. Finally, we close with concluding remarks and a discussion of future work.

2. Related Work

We briefly overview prior work on MIL with an emphasis on applications to CBIR. Maron and Lozano-Pérez [12] were the first to apply MIL to a CBIR task. Their algorithm returns a *point-and-scaling hypothesis* as $h = \{t_1, \ldots, t_d, s_1, \ldots, s_d\}$ where t_k is the feature value for dimension k and s_k is a scale factor indicating the importance of feature k. They introduced the diverse density (DD) algorithm that uses a two-step gradient descent search with multiple starting points to find a hypothesis that minimizes NLDD, the negative logarithm of $DD(h, D)$ which is a measure of the likelihood of h given the training data D. One step of the gradient search modifies $\vec{t}$ and the other modifies $\vec{s}$. Intuitively, the point in this $2d$-dimensional search space where the NLDD is the greatest is where the greatest number of positive bags have points nearby and the all points from the negative bags are far.

Yang et al. [18] built upon the work of Maron and Lozano-Pérez, by using a different fixed partitioning of the image and evaluating the quality of a hypothesis with a weighted correlation similarity measure instead of the diverse density measure.

Zhang and Goldman [20] introduced the EM-DD algorithm which treats the knowledge of which instance corresponds to the label of the bag as a missing attribute and applies a variation of the EM algorithm [4] where it selects the value for the hidden variable with the best (versus expected) correspondence to convert the multiple-instance learning problem to a standard supervised learning problem[1]. EM-DD starts with some initial guess of a point-and-scaling hypothesis h and then repeatedly performs the following two steps. In the first step (E-step), the current hypothesis h is used to pick one instance from each bag which is most likely (given the generative model) to be the one responsible for the label. In the second step (M-step), the two-step gradient search of the DD algorithm is used to find a new h that minimizes $NLDD(h, D)$. From among the multiple-starts of EM, the point-and-scaling concept with the minimum NLDD value is returned. Finally, a run of EM-DD is started from every positive point from five randomly selected positive bags (or from all positive bags if there are less than five) with all scale factors set to 0.1.

Zhang et al. [21] applied EM-DD to CBIR using a segmentation-based approach to represent each image as a bag. Huang et al. [8] presented a variation of their work that incorporated a different segmentation algorithm, and a neural network based MIL algorithm.

Andrews and Hofmann [1] introduced two new MIL algorithms, MI-SVM and mi-SVM, that like EM-DD used an EM-based approach. However, instead of using the DD algorithm for the maximization phase, they used a support vector machine. The difference between these two approaches is the choice of the hidden variables. In mi-SVM, the hidden variables are the labels of the individual points, whereas in MI-SVM, the hidden variable is a selector variable indicating which point in the bag is positive.

Chen and Wang [3] consider the problem of image categorization which is the multi-class problem of determining which of a set of pre-defined categories an image belongs to. For this problem, one would typically expect to have fairly large training sets and the image categories are generally quite broad. DD-SVM uses a one-against-the-rest strategy to handle the multi-class setting. To create the binary classifier for each category, DD-SVM converts the multiple-instance problem into a single instance problem by introducing a dimension for each point from a positive bag and then applies a SVM with examples mapped to this new feature space.

Bi et al. [2] present 1-norm SVM which modifies DD-SVM in several key ways. First, EM-DD is not used, but rather each point in each positive bag is used as a positive prototype and no negative prototypes are used. Second, the feature value in each new dimension uses an unweighted Euclidean metric. Finally, a 1-norm SVM is used since it reduces to solving a linear program which is more efficient. The advantage of this approach is that when the training data size is large (as common for image categorization), comparable results can be obtained with significantly lower computation costs. However, this approach is

[1] More accurately, the method used by EM-DD is MLESAC [16], which is a generalization of RANSAC [6], in which the maximum likelihood value for the hidden variables is selected in the estimation phase.

not good when there are small data sets as one would expect for CBIR. Also, it does not perform any scaling, so all image features are treated equally. Finally, while their method removes the need to tune the parameters used to select the instance prototypes, there are two parameters that must be tuned for the SVM.

Rahmani et al. [14] introduced a new method to represent the image as a bag that is segmentation-based but also introduces features to capture information about the relationship of a segment to its four neighbors in the cardinal directions. They use a variant of EM-DD in which the average label from all hypotheses returned from the multiple runs of EM are used to rank the test images. We have chosen to use their image representation and also perform experiments using the SIVAL benchmark that they introduced.

A common alternative to segmenting images in CBIR is to use salient points which are locations in an image where there is significant variation in a feature. Recently, Hui et al. [19] developed a MIL approach where bags represent the salient points in an image, and an image segmentation is used to form a mask to limit the number of salient points and hence the size of the bag.

There has also been research on building ensembles of multiple-instance learners and applying MIL to CBIR such as the work of Zhou and Zhang [22] and Xu and Frank [17]. Zhou and Zhang applied bagging to a set of different MIL algorithms. For several data sets the bagged version of EM-DD performed best. Xu and Frank applied boosting to a MIL algorithm. One drawback of applying bagging or boosting is that the MIL algorithm must be run multiple times using different training data.

3. Converting an Image into a MIL Bag

Here we briefly describe the two bag generators used in our experimental work. These are both based upon the image representation of Accio [14] in which all images are first transformed into the YCrCb color space and then preprocessed using a wavelet texture filter so that each pixel in the image has three color features and three texture features. Next, a segmentation algorithm is used to segment the image into 32 segments. Since the context in which a segment occurs can be important in defining what the user wants, they augment the feature information for each segment with the difference between its neighbors' value and its value for each of the six features for its neighbor to the right, bottom, left, and top.

In summary, let I be the segmented image. In the *neighbor bag generator*, each segment $x \in I$ is a point in a 30-dimensional feature space where the first 6 features hold the average color and texture values for x, the next 6 features hold the difference between the average color and texture values of x and the northern neighbor. Similarly there are 6 features for the difference information between x and its other 3 cardinal neighbors. So each image is represented as a bag of 32 such points. In the *no-neighbor bag generator*, each segment $x \in I$ is a point in a 6-dimensional feature space with just the color and texture values for the segment itself. Even though this feature space is a subset of the feature space of neighbor bag generator some results are better for this generator (more details in Section 6).

4. Winnow

A seminal result in the on-line model is Littlestone's noise-tolerant algorithm Winnow [10] for learning disjunctions among a set of N attributes in which only k (for $N \gg k$) of the attributes are relevant. Winnow makes predictions based on a linear threshold function

$$\hat{y}_t = \begin{cases} 1 & \text{if } \sum_{i=1}^{N} w_i x_i \geq \theta \\ 0 & \text{otherwise} \end{cases}$$

where w_i is the weight associated with the boolean attribute x_i, and θ is generally set to $N/2$ (unless additional prior information is known about the target concept). If the prediction is wrong then the weights are updated as follows. On a false negative prediction, for each attribute x_i, where x_i = 1, Winnow *promotes* the weight w_i by multiplying w_i by some constant update factor $\alpha > 1$. On a false positive prediction, for each attribute x_i, where x_i = 1, Winnow *demotes* the weight w_i by dividing it by α.

Winnow is similar to the classical Perceptron algorithm [15], except that the Perceptron algorithm updates its weights *additively* while Winnow uses *multiplicative* weight updates. Another major difference between these algorithms is that when learning disjunctions over n attributes with k of them relevant, Winnow's mistake bound is logarithmic in N whereas the Perceptron algorithm's mistake bound can be linear in N in the worst case [9]. Thus Winnow is a good choice when there are many irrelevant features.

We learn a box in the MIL model using the same basic approach of Maass and Warmuth [11] to learn a box in a discretized d-dimensional space $\{0, 1, \ldots, n-1\}^d$. For each of d dimensions they introduce $2n$ halfspaces – n directed in each direction for each discrete value. Winnow can be used to efficiently determine which of the $2d$ halfspaces from among the $2dn$ possible halfspaces are relevant.

5. Multiple-Instance Winnow

In this section we describe multiple-instance winnow (MI-Winnow). A new contribution included within MI-Winnow is an alternative to using EM to handle the ambiguity inherent in not knowing which point in a positive bag is responsible for the positive label. Instead we use the diverse density (DD) measure, over the points in all positive bags, to select a set of candidates for truly positive points. We then create negative training data from all the points in

negative bags, and train Winnow to generate a box. Then we combine the hypotheses obtained from multiple runs of Winnow (possibly with data obtained using different bag generators) to create a final hypothesis.

We have performed experiments in which we have combined EM with MI-Winnow (to reselect the most likely points to be positive and re-running Winnow), but have found that using EM does not improve the accuracy of the resulting hypothesis (and increases the computation cost).

5.1. Reducing to Winnow

Now, we describe how a bag can be converted to a standard example (with one point) for Winnow. Recall that a bag is positive iff at least one point in the bag is positive. However, the learning algorithm can only see the label for the bag. We define a *true positive* point to be a point from a positive bag that has the same label as the bag. What makes MIL hard is that a positive bag might only contain one true positive point – the other points in the bag could be false positives. In contrast, all points in a negative bag are truly negative.

For the moment, suppose MI-Winnow could locate a truly positive point p from a positive bag. Then for any other positive bag, the closest point in that bag to p is very likely to also be a positive point. That is, the set S^+ of the closest point to p from all other positive bags should contain mostly positive instances if p is truly positive. Let S^- contain all points from the negative bags. If p is truly positive then S^+ and S^- will be consistent with a d-dimensional box which can be learned with Winnow.

Many existing algorithms address the difficulty of finding a truly positive point by trying each point from a set of positive bags as the candidate for p. However, such an approach is computationally expensive. MI-Winnow proposes the alternate approach of using the the training data to filter the points from the positive bags to determine which ones are most likely to be truly positive since those are the points that will produce the best results. To achieve this goal, we use the diverse density (DD) measure as first defined by Maron and Lozano-Pérez [12].

We use the following notation and conventions. Let $dist(p_i, p_j)$ denote the Euclidean distance between points p_i and p_j. Let, $d_h(p) = exp\left(-dist^2(h,p)\right)$. Intuitively, $d_h(p)$ can be viewed as the likelihood that p is positive according to h. We use a Gaussian centered around h to convert the distance to this likelihood measure that we then treat as a real-valued label for p. Finally, the diverse density of hypothesis h is $DD(h) = \Pi_{i=1}^{n} \max_j \left\{1 - |y_i - d_h(x_{ij})|^2\right\}$. Observe that the term $|y_i - d_h(x_{ij})|^2$ is the absolute loss between the true label y_i for bag x_i and the label that would be given to the jth point x_{ij} in x_i. Thus the term inside the maximum of the $DD(h)$ definition is a measure of the likelihood that label y_i being given to bag x_i is correct based on point x_{ij}. The label for each bag, is the likelihood for the best point in the bag. Finally, under the assumption (as typically made in defining the DD measure) of independence between the points in a bag, of a uniform prior of the hypothesis space, $DD(h)$ is the likelihood of h being the target based on the labeled data L.

An observation used in creating MI-Winnow is that the use of EM to determine which point in each bag is important as EM-DD does, can be replaced by the much more direct approach of using the DD of each point as a measure of its importance. MI-Winnow calculates the DD of all points in each positive bag (treating all features as equally important) to determine which points are most likely to be truly positive. Only these points are used as a candidate for a truly positive point p. This method generates fewer candidate points for Winnow, and the resulting hypotheses are more likely to be accurate.

An overview of MI-Winnow is shown in Figure 1. We partition the training bags $\mathcal{B}$ into $\{(B_1^+, B_2^-), \cdots (B_b^+, B_b^-)\}$ where each pair is generated by a different bag generator. While all points created by the same bag generator must have the same dimensionality, different bag generators need not generate points of the same dimensionality. MI-Winnow takes two parameters, τ which controls how many different candidates are considered as the truly positive point, and s which controls the number of variables used by Winnow. As shown in the experimental results, the performance of MI-Winnow is not very sensitive to either of these parameter values.

MI-Winnow($\mathcal{B} = \{(B_1^+, B_2^-), \cdots, (B_b^+, B_b^-)\}, \tau, s$)
  Let d_i be the number of dimension for points in (B_i^+, B_i^-)
  Let $B_{i1}^+, \ldots, B_{im_i}^+$ be the bags in B_i^+
  For i = 1 to b
    Let $\mathcal{H} = \{\}$
    Let $P_- = \{p \in B_i^-\}$
    For all $p \in B_i^+$
      Compute $DD(p)$
    For k = 1 to m_i
      $P_\tau = \{p \in B_{ik}^+ \mid DD(p)$ in top $\tau\%\}$
      For all $p \in P_\tau$
        Let $P_+ = \{\}$
        For j = 1 to m_i
          Let $P_q = \{p' \in B_{ij}^+ \mid DD(p')$ in top 25%$\}$
          $p' \in B_{ij}^+$ be the closest to p
          If $p' \in P_q$ then $P_+ = P_+ \cup \{p'\}$
        If $|P_+| > |B_i^+|/2$
          $\mathcal{H} = \mathcal{H} \cup \{$GenHypotheses$(P_+, P_-, d_i, s)\}$
  Return $\mathcal{H}$

Figure 1. Overview of the MI-WINNOW.

The set P_τ contains all points from a positive bag that

GenHypotheses(P_+, P_-, d, s)
Let $[v_i^{min}, v_i^{max}]$ be range of the training data for dimension i
Let $(x_1, \cdots x_d)$ be a point
For i = 1 to d
  $t = (\lceil v_i^{max} \rceil - \lfloor v_i^{min} \rfloor)/(s-1)$
  $h = \lfloor v_i^{min} \rfloor$
  For j = 1 to s
    Add var. that is true iff $x_i \geq h$
    Add var. that is true iff $x_i < h$
    $h = h + t$
Initialize corresponding $2ds$-dim weight vector $\vec{w} = \vec{1}$
Let $\theta = 2ds/2 = ds$
Repeat until training error does not change in an iteration
  Perform Winnow weight update for each pt. $p \in P_-$
  Perform Winnow weight update for each pt. $p \in P_+$
Return $\{w_1/\theta, \cdots, w_{2ds}/\theta\}$

Figure 2. Hypothesis Generation Method.

have a DD value in the top $\tau\%$ among all points in that bag. For each p in P_τ we pick the closest points to p from each positive bag to place in P_+. However, we do not use any point from a bag if the closest point to p is not within the top quarter of DD values among the points in that bag (stored in P_q). The motivation for this choice is that there could be multiple target boxes, each defined by a different subset of positive bags. In such cases, not all positive bags have points in every target box. If at least half the positive bags do not contribute a point to P_+, then this P_+ is not used with Winnow. This optimization further reduces the number of hypotheses generated.

5.2. Generating Hypotheses

Figure 2 describes how the hypothesis are generated from the single-instance data created by MI-Winnow. We use Winnow as our learning algorithm since it can provably learn a box (polynomially in the number of dimensions) and is robust to noise. Having multiple data sets helps Winnow to learn multiple hypotheses (boxes). If multiple boxes are required to define when an image is positive, using a hypothesis space that combines multiple boxes is essential. Once Winnow has been applied to all the training sets (one for each candidate for a truly positive point), the hypotheses created must be combined to create a single overall ranking of the test bags.

5.3. Ranking Images

In this section, we describe how the hypothesis are combined. First, hypotheses with low accuracy on the training set are removed. The remaining hypotheses are combined as described in Figure 3. In summary, Winnow computes a linear threshold function (the threshold for all hypotheses is 1 after the normalization by dividing by θ). Then the label for each bag can be defined according to the point that maximizes the dot product $\vec{w} \cdot \vec{p}$, where $\vec{w}$ are the weights defining the linear function and $\vec{p}$ is a point in the bag representation of the image that was used when training Winnow to obtain the given weight vector. Finally, the sum of these dot products are used to rank. If the goal is instead to classify an image, then a positive label is returned if the average value for the dot product is at least 1. Another alternative would be to use a majority vote of the predictions of the individual hypotheses. We found that these two methods gave similar performance. All results here use the method shown in Figure 3.

Rank($\mathcal{H}$, test image set $\mathcal{T}$)
For image $t \in \mathcal{T}$
  Let $v_t = 0$
  For each hypothesis $\vec{w} \in \mathcal{H}$
    Let A be bag generator used for training data for $\vec{w}$
    Let $B = \{\vec{p}_1, \ldots, \vec{p}_r\}$ be bag output by $A(t)$
    Let $v_t = v_t + \max_{\vec{p}_i \in B} \vec{w} \cdot \vec{p}_i$
Rank images in non-increasing order using $\{v_1, \ldots, v_{|\mathcal{T}|}\}$

Figure 3. Ranking Method.

6. Experimental Results

As a measure of performance we have chosen to use the area under the ROC curve [7]. The ROC (Receiver Operating Characteristic) curve plots the true positive rate (i.e. the recall) as a function of the false positive rate. The area under the ROC curve (AUC) is equivalent to the probability that a randomly chosen positive image will be ranked higher than a randomly chosen negative image. Unlike the precision-recall curve, the ROC curve is insensitive to ratio of positive to negative examples in the image repository. Regardless of the fraction of the images that are positive, for a random permutation the AUC is 0.5.

We consider two different CBIR data sets. MI-Winnow works well in both settings without adjusting any parameters – the identical algorithm is run for both. The SIVAL benchmark (`www.cse.wustl.edu/~sg/accio`) includes 25 image categories containing 1500 images. The categories consist of complex objects photographed against 10 different highly diverse backgrounds. SIVAL emphasizes the task of Localized CBIR through nearly identical scenes which only vary by the localized target objects. For each object class the same complex physical object was used in all scenes. It is also a difficult data set in that the scenes are highly diverse and often complex. Furthermore, the objects may occur anywhere spatially in the image and also may be photographed at a wide-angle or close up or with different orientations. In most images, the target object occupies less than 10%-15% of the image area but may occupy as much as 70%.

Table 1. Combining with and without neighbors (8 positive and 8 negative examples)

Category	Combined	No Neighbors	Neighbors	Accio!
AjaxOrange	.872±.026	.852±.029	.830±.036	.770±.034
Apple	.698±.065	.719±.060	.585±.059	.634±.034
Banana	.737±.029	.758±.025	.598±.031	.659±.033
BlueScrunge	.793±.044	.819±.038	.586±.051	.695±.034
CandleWithHolder	.811±.015	.694±.023	.861±.015	.688±.023
CardboardBox	.665±.028	.574±.030	.725±.038	.679±.022
CheckeredScarf	.924±.010	.886±.011	.932±.012	.908±.016
CokeCan	.918±.010	.878±.013	.919±.024	.815±.035
DataMiningBook	.812±.034	.803±.039	.745±.045	.747±.034
DirtyRunningShoe	.812±.017	.729±.020	.844±.017	.837±.019
DirtyWorkGloves	.652±.024	.548±.021	.720±.031	.653±.015
FabricSoftenerBox	.944±.011	.905±.013	.956±.011	.866±.030
FeltFlowerRug	.877±.015	.839±.013	.887±.015	.869±.017
GlazedWoodPot	.675±.051	.677±.055	.585±.030	.727±.023
GoldMedal	.890±.039	.892±.035	.741±.049	.777±.026
GreenTeaBox	.908±.022	.890±.024	.864±.030	.873±3.0
JuliesPot	.743±.053	.720±.061	.721±.058	.792±.026
LargeSpoon	.518±.018	.508±.015	.529±.025	.576±.023
RapBook	.554±.035	.527±.037	.583±.031	.628±.017
SmileyFaceDoll	.869±.029	.875±.024	.724±.040	.774±.033
SpriteCan	.847±.021	.777±.019	.856±.018	.719±.025
StripedNotebook	.763±.030	.760±.029	.724±.038	.702±.032
TranslucentBowl	.812±.046	.822±.040	.704±.053	.775±.023
WD40Can	.909±.014	.879±.017	.907±.014	.820±.024
WoodRollingPin	.571±.036	.548±.041	.570±.029	.667±.017
Average	**.783**	**.755**	**.748**	**.746**

A very different data set is obtained from `www.cs.uno.edu/~yixin/ddsvm.html`. Since these images are from the Corel corporation, we will refer to it as the COREL data set[2]. In this data set there are 20 very broad categories. Unlike SIVAL in which image contains many objects other than the one of interest, in the COREL benchmark each image just contains an object from particular category. Thus in many ways, the SIVAL and COREL benchmark are very different in characteristics. For both data sets, we segment each image into 32 segments. Each segment has either 6 features (3 color and 3 texture) with the no-neighbor bag generator, or 30 features when using the neighbor bag generator. Unless otherwise mentioned the experimental results are 30 fold with the mean AUC and 95% confidence interval.

We first evaluated the sensitivity of MI-Winnow to the choice for τ. Recall that the computation time increases as τ increases. Also, if τ is too large, then incorrect hypotheses are likely to be created since the selected point is not likely to be truly positive. We have performed experiments with the SIVAL dataset with $\tau = 10\%$, 25%, 50%, and 100% and found that all worked quite well achieving mean AUC values of 0.748, 0.747, 0.747, and 0.746, respectively. In all of these tests we set s, the number of halfspaces to 15 and used the neighbor bag generator. The performance is same in all the cases showing that DD measure is picking good points as starting points. For the remainder of the experiments reported let $\tau = 10\%$.

Next we evaluated the sensitivity of MI-Winnow to the choice for s, the number of halfspaces used by Winnow for each dimension. We have performed experiments with the SIVAL dataset with values for s of 15, 30 and 100. The average AUC values were 0.748, 0.747, and 0.752, respectively. Observe that even with only 15 halfspaces MI-Winnow performs almost as good as 100 halfspaces. By reducing the number of halfspaces the time complexity of MI-Winnow is reduced without any significant reduction in the performance. For the remainder of these experiments we fix the number of halfspaces to 15.

For some categories the results are better when using the neighbor bag generator, and for other categories the results are better when using the no-neighbors bag generator. To benefit from both representations, in the next set of experiments we compared the performance when both bag generators are combined as compared to just using them individually. (See Table 1.) For most categories the results when

[2]There are many other different data sets that have been created from Corel CDs.

Table 2. Varying the number of training examples

Category	(# positive examples, # negative examples) (4,4)	(8,8)	(12,12)	(16,16)
AjaxOrange	0.780 ± 0.039	0.830 ± 0.036	0.904 ± 0.029	0.911 ± 0.026
Apple	0.516 ± 0.043	0.585 ± 0.059	0.671 ± 0.050	0.752 ± 0.040
Banana	0.501 ± 0.037	0.598 ± 0.031	0.646 ± 0.042	0.720 ± 0.035
BlueScrunge	0.481 ± 0.042	0.586 ± 0.051	0.632 ± 0.051	0.770 ± 0.044
CandleWithHolder	0.814 ± 0.035	0.861 ± 0.015	0.873 ± 0.011	0.884 ± 0.009
CardboardBox	0.651 ± 0.033	0.725 ± 0.038	0.771 ± 0.028	0.829 ± 0.025
CheckeredScarf	0.903 ± 0.014	0.932 ± 0.012	0.948 ± 0.007	0.961 ± 0.008
CokeCan	0.900 ± 0.017	0.919 ± 0.024	0.948 ± 0.007	0.962 ± 0.006
DataMiningBook	0.584 ± 0.034	0.745 ± 0.045	0.795 ± 0.040	0.854 ± 0.027
DirtyRunningShoe	0.784 ± 0.026	0.844 ± 0.017	0.883 ± 0.010	0.875 ± 0.012
DirtyWorkGloves	0.683 ± 0.030	0.720 ± 0.031	0.787 ± 0.021	0.811 ± 0.018
FabricSoftenerBox	0.915 ± 0.032	0.956 ± 0.011	0.970 ± 0.007	0.975 ± 0.005
FeltFlowerRug	0.867 ± 0.019	0.887 ± 0.015	0.922 ± 0.012	0.935 ± 0.011
GlazedWoodPot	0.475 ± 0.035	0.585 ± 0.030	0.691 ± 0.031	0.694 ± 0.033
GoldMedal	0.595 ± 0.041	0.741 ± 0.049	0.830 ± 0.044	0.901 ± 0.025
GreenTeaBox	0.755 ± 0.045	0.864 ± 0.030	0.918 ± 0.023	0.946 ± 0.021
JuliesPot	0.534 ± 0.038	0.721 ± 0.058	0.834 ± 0.030	0.893 ± 0.013
LargeSpoon	0.500 ± 0.019	0.529 ± 0.025	0.531 ± 0.015	0.525 ± 0.020
RapBook	0.505 ± 0.025	0.583 ± 0.031	0.634 ± 0.025	0.683 ± 0.024
SmileyFaceDoll	0.676 ± 0.038	0.724 ± 0.040	0.812 ± 0.028	0.820 ± 0.033
SpriteCan	0.787 ± 0.036	0.856 ± 0.018	0.888 ± 0.011	0.907 ± 0.010
StripedNotebook	0.590 ± 0.040	0.724 ± 0.038	0.780 ± 0.037	0.777 ± 0.047
TranslucentBowl	0.578 ± 0.041	0.704 ± 0.053	0.694 ± 0.051	0.806 ± 0.037
WD40Can	0.844 ± 0.024	0.907 ± 0.014	0.930 ± 0.010	0.941 ± 0.007
WoodRollingPin	0.489 ± 0.041	0.570 ± 0.029	0.569 ± 0.028	0.594 ± 0.031
Average	**0.668**	**0.748**	**0.794**	**0.829**

using both bag generators are almost as good (or better) as the best one when only using one bag generator. Only in a few cases is the performance significantly worse (at the 95% confidence interval) than the best of the two options. By combining both bag generators, we obtain better performance than Accio! when it uses the best parameter choices. MI-Winnow can be viewed as just using two of these five starting values.

Table 2 shows another set of experiments in which we vary the size of the training data. We label each column by (p, n) where p is the number of positive images in the training data, and n is the number of negative images in the training data. The rest of the available images are used as test data. For these experiments we only used the neighbor bag generator. We believe that better performance can be obtained by combining the results with both bag generators. As expected, we find that the performance improves as more training data is given.

Table 3 gives results with combined method for the Corel data set which is for a broad image category. Localized Content-Based Image Retrieval has been defined as a CBIR task where the user is only interested in a portion of the image, and the majority of the image is irrelevant. Though MI-Winnow is designed for localized CBIR task these results show that it performs well with the global CBIR task. Here again the performance improves when there are more training examples. To best of our knowledge we are the first to report AUC results for this data set, so we are unable to make comparisons with other work.

7. Concluding Remarks

We described a new algorithm, MI-Winnow, for CBIR task which combines the multiple-instance problem with Winnow. It outperforms the current best method Accio! on SIVAL data set and performs well on the COREL data set. Using various experiments we showed that MI-Winnow works well under various parameter settings indicating that there is no need to tune the algorithm too much based on the data set. We plan to perform experiments on additional data sets and also compare the AUC values from MI-Winnow with other existing algorithms on the Corel data set. We also plan to perform some experiments in which we consider classification tasks as opposed to ranking tasks.

Unlike Accio!, MI-Winnow can not only rank images but also predict whether a given image is positive or not. Given the training data consisting of images from multiple cate-

Table 3. Results on Corel Data Set

Category	(# positive exs, # negative exs) (8,8)	(16,16)
African	0.794 ± 0.019	0.844 ± 0.012
beach	0.704 ± 0.032	0.789 ± 0.015
historical	0.729 ± 0.012	0.730 ± 0.010
buses	0.888 ± 0.013	0.911 ± 0.008
dinosaurs	0.995 ± 0.001	0.997 ± 0.001
elephants	0.682 ± 0.028	0.725 ± 0.021
flowers	0.927 ± 0.010	0.942 ± 0.009
horses	0.833 ± 0.033	0.881 ± 0.022
mountains	0.758 ± 0.026	0.808 ± 0.016
food	0.848 ± 0.019	0.876 ± 0.015
dogs	0.667 ± 0.032	0.714 ± 0.025
lizards	0.779 ± 0.018	0.817 ± 0.013
fashion	0.698 ± 0.039	0.754 ± 0.029
sunsets	0.931 ± 0.010	0.931 ± 0.005
cars	0.763 ± 0.030	0.816 ± 0.014
waterfalls	0.738 ± 0.028	0.800 ± 0.023
antiques	0.908 ± 0.014	0.933 ± 0.008
battleships	0.866 ± 0.019	0.884 ± 0.018
skiing	0.862 ± 0.018	0.890 ± 0.011
desserts	0.504 ± 0.025	0.564 ± 0.022
Average	**0.794**	**0.830**

gories MI-Winnow can be used to learn hypotheses with each category as positive. The hypotheses predict if a given image is positive or not. These predictions can be combined to predict the category of the image. Work is under progress for this categorization tasks and looks very promising.

8. Acknowledgments

We would like to thank Hui Zhang and Jason Fritts for helpful discussions. We also thank the anonymous reviewers for their valuable feedback. This material is based upon the work supported by the National Science Foundation under Grant No. 0329241.

References

[1] S. Andrews, T. Hofmann, and I. Tsochantaridis. Multiple instance learning with generalized support vector machines. In *18th National Conference on Artificial Intelligence*, pages 943–944. American Association for Artificial Intelligence, 2002.

[2] J. Bi, Y. Chen, and J. Wang. A sparse support vector machine approach to region-based image categorization. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1121–1128, 2005.

[3] Y. Chen and J. Wang. Image categorization by learning and reasoning with regions. *Journal of Machine Learning Research*, 5:913–939, 2004.

[4] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistics Society*, 39:1–38, 1977.

[5] T. Dietterich, R. Lathrop, and T. Lozano-Pérez. Solving the multiple-instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2):31–71, 1997.

[6] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartograpy. *Comm. of the ACM*, 24:381–395, 1981.

[7] J. Hanley and B. McNeil. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143:29–36, 1982.

[8] X. Huang, S.-C. Chen, M.-L. Shyu, and C. Zhang. User concept pattern discovery using relevance feedback and multiple instance learning for content-based image retrieval. *8th Int. Conf. on Knowledge Discovery and Data Mining*, pages 100–108, 2002.

[9] J. Kivinen, M. K. Warmuth, and P. Auer. The perceptron algorithm vs. Winnow: Linear vs. logarithmic mistake bounds when few input variables are relevant. *Artificial Intelligence*, 97(1-2):325–343, 1997.

[10] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4):285–318, 1988.

[11] W. Maass and M. K. Warmuth. Efficient learning with virtual threshold gates. *Inf. Comput.*, 141(1):66–83, 1998.

[12] O. Maron and T. Lozano-Pérez. A framework for multiple-instance learning. *NIPS*, 1997.

[13] O. Maron and A. Ratan. Multiple-instance learning for natural scene classification. *Proc. 15th Int. Conf. on Machine Learning*, pages 341–349, 1998.

[14] R. Rahmani, S. Goldman, H. Zhang, J. Krettek, and J. Fritts. Localized content-based image retrieval. *Proc. of ACM Workshop on Multimedia Image Retrieval*, pages 227–236, 2005.

[15] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psych. Rev.*, 65:386–407, 1958. (Reprinted in *Neurocomputing* (MIT Press, 1988).).

[16] P. Torr and A. Zisserman. Mlesac: a new robust estimator with application to estimating image geometry. *Comput. Vision Image Understand.*, 78:138–156, 2000.

[17] X. Xu and E. Frank. Logistic regression and boosting for labeled bags of instances. In H. Dai, R. Srikant, and C. Zhang, editors, *Advances in Knowledge Discovery and Data Mining*, volume 3056 of *LNAI*, pages 272–281, 2004.

[18] C. Yang and T. Lozano-Pérez. Image database retrieval with multiple instance techniques. *Proc. of the 16th Int. Conf. on Data Engineering*, pages 233–243, 2000.

[19] H. Zhang, R. Rahmani, S. Cholleti, and S. Goldman. Local image representations using pruned salient points with applications to CBIR. *Proc. 14th Annual ACM Int. Conf. on Multimedia*, 2006.

[20] Q. Zhang and S. Goldman. EM-DD: an improved multiple-instance learning technique. *NIPS*, 2001.

[21] Q. Zhang, S. Goldman, W. Yu, and J. Fritts. Content-based image retrieval using multiple instance learning. *Proc. 19th Int. Conf. on Machine Learning*, pages 682–689, 2002.

[22] Z.-H. Zhou and M.-L. Zhang. Ensembles of multi-instance learners. *Proc. 14th European Conf. on Machine Learning*, 2003.