Automatic image annotation using a semi-supervised ensemble of classifiers

Heidy-Marisol Marin-Castro, L. Enrique Sucar, and Eduardo F. Morales

National Institute of Astrophysics, Optics and Electronics,
Computer Science Department,
Luis Enrique Erro 1, 72840 Tonantzintla, México
{hmmarinc,esucar,emorales}@inaoep.mx
http://ccc.inaoep.mx

Abstract. Automatic image annotation consists on automatically labeling images, or image regions, with a pre-defined set of keywords, which are regarded as descriptors of the high-level semantics of the image. In supervised learning, a set of previously annotated images is required to train a classifier. Annotating a large quantity of images by hand is a tedious and time consuming process; so an alternative approach is to label manually a small subset of images, using the other ones under a semi-supervised approach. In this paper, a new semi-supervised ensemble of classifiers, called WSA, for automatic image annotation is proposed. WSA uses naive Bayes as its base classifier. A set of these is combined in a cascade based on the AdaBoost technique. However, when training the ensemble of Bayesian classifiers, it also considers the unlabeled images on each stage. These are annotated based on the classifier from the previous stage, and then used to train the next classifier. The unlabeled instances are weighted according to a confidence measure based on their predicted probability value; while the labeled instances are weighted according to the classifier error, as in standard AdaBoost. WSA has been evaluated with benchmark data sets, and 2 sets of images, with promising results.

1 Introduction

In recent years the amount of digital images in databases has grown impressively. This situation demands efficient search methods to extract images in huge collections according to the user requirements, in what is known as *content–based image retrieval*. To solve this problem, one alternative is to include with each image a list of keywords that describe the semantics of the image. However, this is not practical, because many images do not have an associated caption, and it is too costly to label a huge collection manually. Another alternative is automatic image annotation. Automatic image annotation consists on automatically labeling images, or image regions, with a pre-defined set of keywords, which are regarded as descriptors of the high-level semantics of the image. Once annotated, the set of keywords obtained are associated to the image for future queries.

Recently, there has been an increasing interest on automatic image annotation [3, 8, 9, 11]. Most methods are based on machine learning techniques, where

a set of manually labeled images is used to train a classifier, and then the classifier is used to label the rest of the images. In some cases, the labels are assigned to an specific image region; and in others, labels are globally assigned to each image [9]. A third approach considers salient features in the images, to avoid segmentation [11]. In these approaches the performance of the annotation systems depends on the quantity and quality of the training set, which was manually labeled. However, there is usually a larger set of images that has not been manually labeled, and which in principle could be used to improve the annotation system using a semi-supervised approach.

Semi-supervised methods exploit unlabeled data in addition to labeled data to improve classification performance. This approach could be used with different classification methods, such as neural networks, support vector machines and statistical models. In this work we are interested in improving ensemble methods, in particular AdaBoost [6], using unlabeled data. Ensemble methods work by combining a set of base or *weak* classifiers (usually a simple classifier, such as Naive Bayes) in some way, such as a voting scheme, producing a combined classifier which usually outperforms a single classifier, even a more complex one.

In this paper, a new semi-supervised ensemble of classifiers, called WSA (Weighted Semi-supervised AdaBoost), for automatic image annotation is proposed. It is based on AdaBoost and uses naive Bayes as its base classifier. When training the ensemble of Bayesian classifiers, WSA also considers the unlabeled images on each stage. These images are annotated based on the classifier from the previous stage, and then used to train the next classifier. The unlabeled instances are weighted according to a confidence measure based on their probability; while the labeled instances are weighted according to the classifier error, as in standard AdaBoost. Although there is some previous work in using unlabeled data in ensemble methods [2], they assign a smaller initial weight to the unlabeled data and from then on, the weights are changed as in AdaBoost. In our approach, the weights of unlabeled instances are dynamic, proportional to the probability given by the previous stage. Also, this approach has not been applied to automatic image annotation.

WSA was experimentally evaluated on two set of experiments. In the first one we used two standard data sets from the UCI repository [5], using different percentages of data as labeled and unlabeled. We compared our approach against: (i) supervised AdaBoost and (ii) a semi-supervised version without giving weights to the unlabeled data; using 10 fold cross-validation. In the second experiments we evaluated the performance of image annotation using two subsets of the Corel image data set [4].

The rest of the paper is organized as follows. Section 2 describes the AdaBoost algorithm using Naive Bayes as the base classifier, while section 3 introduces our semi-supervised AdaBoost algorithm with variable weights (WSA). In the next section we briefly describe how images are segmented, and the visual features obtained per region. In section 5 the experimental results are described, and we conclude with a summary and directions for future work.

2 The AdaBoost Algorithm

We start by describing the base classifier used with AdaBoost.

2.1 Base classifier

As base classifier we use the naive Bayes classifier, which is a simple method that has shown good performance in many domains. It is also very efficient to train and for classification, which is important when a large number of classifiers is combined. A Bayesian classifier obtains the posterior probability of each class, C_i, using Bayes rule. The naive Bayes classifier (NBC) makes the simplifying assumption that the attributes, A, are conditionally independent between each other given the class, so the likelihood can be obtained by the product of the individual conditional probabilities of each attribute given the class. Thus, the posterior probability, $P(C_i|A_1, \ldots, A_n)$, is given by:

$$P(Ci \mid A_1, \ldots, A_n) = P(C_i)P(A_1 \mid C_i) \ldots P(A_n \mid C_i)/P(A) \tag{1}$$

In this work we consider the discrete version of the NBC, so the continuous attributes are previously discretized.

2.2 AdaBoost

Our method is based on the supervised multi-class AdaBoost ensemble, which has shown to be an efficient scheme to reduce the rate error of different classifiers, such as trees or neural networks. The main idea of AdaBoost [6] is to combine a series of base classifiers using a a weighted linear combination. Each time a new classifier is generated it tries to minimize the expected error by assigning a higher weight to the samples that were wrongly classified in the previous stages. Ensembles tend to improve the limitations of using a single classifier (e.g., [7]). When the training samples can not provide enough information to generate a "good" classifier; however, the correlated errors of the single classifiers can be eliminated when the decisions of the other classifiers are considered.

Formally, the AdaBoost algorithm starts from a set S of labeled instances, where each instance, x_i, is assigned a weight, $W(x_i)$. It considers N classes, where the known class of instance x_i is y_i. The base classifier is h, and h_t is one of the T classifiers in the ensemble. AdaBoost produces a linear combination of the H base classifiers, $F(x) = \sum_t \alpha_t h_t$, where α_t is the weight of each classifier. The weight is proportional to the error of each classifier on the training data. Initially the weights are equal for all the instances, and these are used to generate the first base classifier, h_1 (using the training algorithm for the base classifier, which should consider the weight of each instance). Then the error, e_1, of h_1 is obtained by summing the weights of the incorrectly classified instances. The weight of each correctly classified instance is increased by the factor $\beta_t = e_t/(1 - e_t)$, and these weights are used to train the next base classifier. The cycle is repeated until $e_t > 0.5$ or when a predefined maximum number of iterations is reached. Supervised AdaBoost is shown in algorithm 1.

Algorithm 1 AdaBoost algorithm.

Input: S: Labeled instances, T: Iterations, W: weighted vector

Output: Final Hypothesis: $H_f = \text{argmax} \sum_{t=1}^{T} log \frac{1}{B_t}$

1: Initialize W. $W(x_i)^0 = \frac{1}{NumInst(S)}$
2: **for** t from 1 to T **do**
3: Normalize W. $W(x_i)^t = \frac{W(x_i)}{\sum_{i=1}^{N} W(x_i)}$
4: Call weak algorithm. $h_t = C(S, W(x_i)^t))$
5: Compute the error. $e_t = \sum_{i=1}^{N} W(x_i)^t$ if $h_t(x_i) \neq y_i$
6: **if** $e_t \geq 0.5$ **then**
7: exit
8: **end if**
9: $B_t = \frac{e_t}{(1-e_t)}$
10: Re-compute W. $W(x_i)^{(t+1)} = W(x_i)^t * B_t$ if $h_t(x_i) = y_i$
11: **end for**

3 Variable weight semi-supervised AdaBoost

Labeling large sets of instances is a tedious process, so we will like to label only a small fraction of the training set, combining the labeled instances with the unlabeled ones to generate a classifier. This paper introduces a new semi-supervised learning algorithm, called WSA, for image annotation. WSA is based on AdaBoost and uses a naive Bayes classifier as its base classifier. WSA receives a set of labeled data (L) and a set of unlabeled data (U). An initial classifier NB_1 is build using L. The labels in L are used to evaluate the error of NB_1. As in AdaBoost the error is used to weight the examples, increasing the weight of the misclassified examples and keeping the same weight of the correctly classified examples. The classifier is used to predict a class for U with certain probability. In the case of U, the weights are multiplied by the predicted probability of the majority class. Unlabeled examples with high probability of their predicted class will have more influence in the construction of the next classifier than examples with lower probabilities. The next classifier NB_2 is build using the weights and predicted class of $L \cup U$. NB_2 makes new predictions on U and the error of NB_2 on all the examples is used to re–weight the examples. This process continues, as in AdaBoost, for a predefined number of cycles or when a classifier has a weighted error greater or equal to 0.5. The main differences with AdaBoost are: (i) WSA uses labeled and unlabeled data, (ii) the base classifiers create new class labels for the unlabeled instances, and (iii) the weights assigned to the original unlabeled data depends on its predicted probability class. As in AdaBoost, new instances are classified using a weighted sum of the predicted class of all the constructed base classifiers. WSA is described in algorithm 2.

Algorithm 2 Semi-supervised Weighted AdaBoost (WSA) algorithm.

Input: L: labeled instances, U: unlabeled instances, P: training instances, T: Iterations

Output: Final Hypotesis and probabilities: $H_f = \text{argmax}\sum_{t=1}^{T} log\frac{1}{B_t}$, $P(x_i)$

$W(x_i)^0 = \frac{1}{NumInst(L)}, \forall x_i \in L$
for t from 1 to T **do**
  $W(x_i)^t = \frac{W(x_i)}{\sum_{i=1}^{N} W(x_i)} \forall x_i \in L$
  $h_t = C(L, W(x_i)^t)$
  $e_t = \sum_{i=1}^{N} W(x_i)^t$ if $h_t(x_i) \neq y_i$
  if $e_t \geq 0.5$ **then**
    exit
  end if
  if $e_t = 0.0$ **then**
    $e_t = 0.01$
  end if
  $B_t = \frac{e_t}{(1-e_t)}$
  $W(x_i)^{(t+1)} = W(x_i)^t * B_t$ if $h_t(x_i) = y_i \ \forall x_i \in L$
  $P(x_i) = C(L, U, W(x_i)^t)$
  $W(x_i) = P(x_i) * B_t \ \forall x_i \in U$
end for

4 Image segmentation and visual features

Before applying our semi-supervised method for image annotation, we perform two operations on the images: (i) segmentation and (ii) feature extraction.

The Normalized Cuts algorithm [10] was used for image segmentation. This algorithm considers the image as a non–directed complete graph. Each pixel in the image represents a node in the graph. There is an edge between each pair of pixels a y b, with a cost C_{ab} that measures the similarity of a and b. The algorithm finds those edges with a small cost and eliminates them making a cut in the graph. Each one of the edges must keep similar pixels in the same segments. The minimum cut is performed by using equation (2), where $cut(A, B)$ is a cut between the segments A and B, $volume(A)$ is the sum of each border that touches A and $volume(B)$ is the sum of the borders that touch B.

$$Ncut(A, B) = \frac{cut(A, B)}{Volume(A)} + \frac{cut(A, B)}{Volume(B)} \tag{2}$$

Once the image is segmented, each segment or region is characterized by a set of visual features that describe the region, and which will constitute the attribute vector for WSA. We consider a set of features for color, texture and shape per region:

Color: This feature is the most common in image retrieval. Several representations for this feature have been considered such as the histogram, momentum, sets and color distributions. We use the color histogram in the RGB color space (8 values per band), as well as the mean and variance for each band.

Texture: The perception of textures also plays an important role in content-based image retrieval. Texture is defined as the statistical distribution of spatial dependences for the gray level properties [1]. One of the most powerful tools for texture analysis are the Gabor filters [3], which can be viewed as the product of a low pass (Gaussian) filter at different orientations and scales. A Gabor filter in 2D is given by equation (3).

$$g(x,y) = \left(\frac{1}{2\pi\sigma_x\sigma_y}\right) exp\left[-\frac{1}{2}\left(\left(\frac{x}{\sigma_x}\right)^2 + \left(\frac{y}{\sigma_y}\right)^2\right) + jw\,(xcos\theta)\right] \quad (3)$$

where θ represents the orientation of the filter in the range. The constants σ_x and σ_y determine the fall of the Gaussian in the x-axis and the y-axis, respectively. jw represents the frequency along x-axis. To characterize texture we used 4 filters with orientations of 0, 45, 90 and 135 grades, with one scale.

Shape: The following set of features were used to characterize the shape of the region:

1. Area: is an scalar that measures the actual number of pixels the image region.
2. Convex Area: represents the number of pixels in the convex hull of the region.
3. Perimeter: computed as $\sqrt{\frac{4*Area}{\pi}}$.
4. Major and minor axis: measure the longitude in pixels of the major and minor axis of the region.
5. Solidity: computed as $\frac{Area}{Convex_area}$.

5 Experiments and results

WSA was tested on datasets from the UCI Machine Learning Repository [5] and on the Corel image database [4]. For the image collection a user interface, shown in figure 1, was designed to easily label a small set of examples. This tool allows to segment images and to extract their features such as color, texture, and form. Only regions with a *large proportion* of a single object were manually labeled.

WSA was compared against AdaBoost and against a version of WSA without changing the weights of the unlabeled instances using the predicted probability value, which we will call SA. Real valued attributes were discretized in 10 bins using WEKA [12].

The algorithms were evaluated by their predicted precision using 10-fold cross validation for different percentages of unlabeled data on the training sets.

Two datasets were used from UCI repository: *Iris* and *Balance-Scale*, whose characteristics are given in table 1. Figure 2 shows the performance of WSA, SA, and (supervised) AdaBoost on both datasets. As can be seen from the figure,

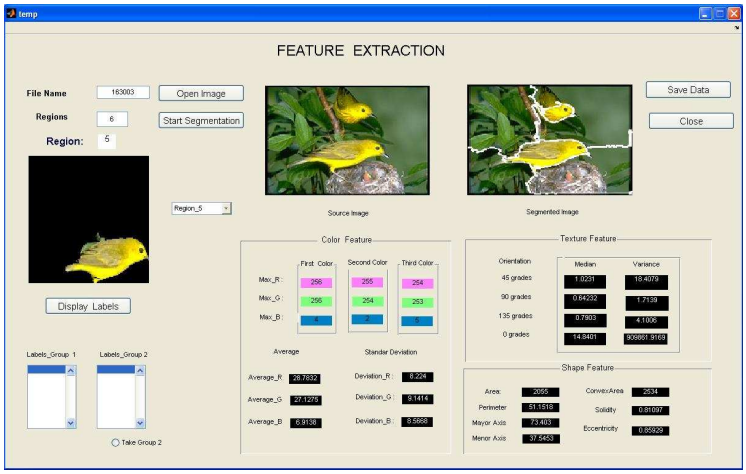

Fig. 1. Graphical user interface for image segmentation, feature extraction and region labeling.

using unlabeled data can improve the performance of AdaBoost, in particular, when there is a large number of unlabeled instances. Also, WSA has a better performance than SA which shows that using the probability class value on the unlabeled instances can have a positive effect as it reduces the unwanted bias that the unlabeled data can produce in the classifier.

Table 1. Characteristics of the Iris and Balance-Scale datasets.

Datasets	**N**um-Instances	**N**um-Attributes	**N**um-Classes
Iris	150	4	3
Balance-Scale	625	4	3

WSA was also tested on the Corel images, that are grouped according to different topics, such as, scenes, animals, buildings, airplanes, cars, and persons, among others. The size of these color images is: 192x128 pixels. The images were segmented with normalized cuts (5 regions) and a set of visual features was obtained per region, as describe in section 4. We performed tests in two topics: airplanes and birds. 100 images were randomly selected from the airplane topic; from these images 127 regions were used as the training set. In this test we considered 6 classes. We also used images of bird topic, also with 6 classes, from which 225 regions were considered for training. Table 2 shows the characteristics of these two datasets and the performance obtained by the WSA classifier using different percentage of labeled data. Additionally, figure 3 compares the performance of WSA, SA, and AdaBoost. For the airplanes collection, their is

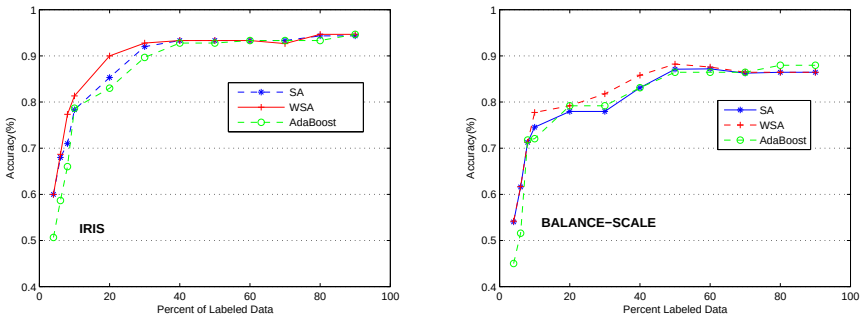

Fig. 2. Performance of WSA (red/cross), SA (blue/asterisk) and AdaBoost (green/circle) on the Iris and Balance-Scale data data sets form the UCI repository.

a significant improvement using WSA vs. AdaBoost, for most percentages of labeled data; while for the birds collection, their accuracy is similar, although slightly better with WSA. In both cases, WSA is superior to SA, which confirms that weighting unlabeled data is important; wrongly labeled data could even decrease the performance of the classifier, as shown in these experiments.

Table 2. Accuracy with different percentage of labeled data.

Dataset	**C**lasses	**N**um.Inst.	**1**0%	**3**0%	**5**0%	**7**0%	**9**0%
Airplanes	sky, jet, cloud, plane, sunset, helicopter	127	40.83	55.00	76.66	90.08	99.16
Birds	branch, bird, tree, grass, nest, rock	225	32.08	51.16	74.16	86.25	90.10

6 Conclusions

In this paper we proposed WSA, a semi-supervised ensemble of classifiers for automatic image annotation. It is based on AdaBoost using naive Bayes as its base classifier. It incorporates unlabeled instances, which are annotated based on the classifier from the previous stage, and then used to train the next classifier. These unlabeled instances are weighted according to a confidence measure based on the class probability given by the classifier of the previous stage. The main differences between WSA and AdaBoost are: (i) WSA uses labeled and unlabeled data, (ii) the base classifiers create new class labels for the unlabeled instances, and (iii) the weights assigned to the unlabeled data depends on its predicted probability class.

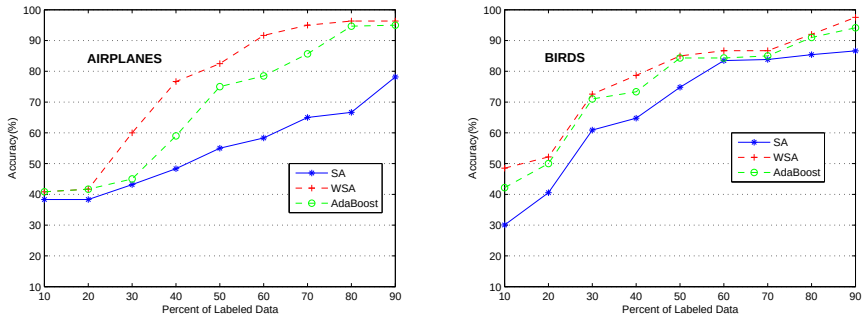

Fig. 3. Performance of SWA (red/cross), SA (blue/asterisk) and AdaBoost (green/circle) on images of airplanes and birds from the Corel database.

Initial experiments on images and other data show promising results. Using unlabeled data we can improve the performance of AdaBoost, in particular, when there is a large number of unlabeled instances. Also, WSA has a better performance than SA which shows that using the probability class value on the unlabeled instances can have a positive effect as it reduces the unwanted bias that the unlabeled data can produce in the classifier.

As future work we plan to perform a more comprehensive experimentation with other data sets.

Referencias

1. Selim Aksoy and Robert Haralick. Textural features for image database retrieval. In *CBAIVL'98: Proceedings of the IEEE Workshop on Content - Based Access of Image and Video Libraries*, page 45, Washington, DC, USA, 1998. IEEE Computer Society.
2. Kristin Bennett, Ayhan Demiriz, and Richard Maclin. Exploiting unlabeled data in ensemble methods. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 289–296, NY, 2002. ACM Press.
3. Lianping Chen, Guojun Lu, and Dengsheng Zhang. Content-based image retrieval using gabor texture features. In *In Proceedings of First IEEE Pacific-Rim Conference on Multimedia (PCM'00)*, pages 1139–1142, Sydney, Australia, 2000.
4. Corel. Corel images, 2003.
5. C.L. Blake David Newman, S. Hettich and Christopher Merz. UCI repository of machine learning databases, 1998.
6. Yoav Freund and Robert Schapire. Experiments with a new boosting algorithm. In *International Conference on Machine Learning*, pages 148–156, 1996.
7. Ludmila Kuncheva. Using measures of similarity and inclusion for multiple classifier fusion by decision templates. *Fuzzy Set and Systems*, 122(3):401–407, 2001.
8. Wei Li and Sun Maosong. Automatic image annotation based on wordnet and hierarchical ensembles. In *Proceedings of 7th International Conference on Computational Linguistics and Intelligent Text Processing, CICLing 2006*, volume 3878

of *Lecture Notes in Computer Science*, pages 417–428, Heidelberg, 2006. Springer Verlag.

9. Nando de Freitas Pinar Duygulu, Kobus Barnard and David Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *Proceedings of the 7th European Conference on Computer Vision-Part IV*, pages 97–112, London., 2002. Springer-Verlag.
10. Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, 2000.
11. Jiayu Tang, Jonathon S. Hare, and Paul H. Lewis. Image auto-annotation using a statistical model with salient regions, 2006.
12. Ian Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques*. 2nd Edition, Morgan Kaufmann, San Francisco, 2005.