Text Separation from Mixed Documents Using a Tree-structured Classifier

Xujun Peng, Srirangaraj Setlur, Venu Govindaraju
Center for Unified Biometrics and Sensors
Dept of Computer Science and Engineering
SUNY at Buffalo, Amherst, NY 14228, USA
{xpeng, setlur, govind}@buffalo.edu

Ramachandrula Sitaram
HP Labs India
Hosur Main Road, Adugodi
Bangalore 560030, India
sitaram@hp.com

Abstract

In this paper, we propose a tree-structured multi-class classifier to identify annotations and overlapping text from machine printed documents. Each node of the tree-structured classifier is a binary weak learner. Unlike normal decision tree(DT) which only considers a subset of training data at each node and is susceptible to over-fitting, we boost the tree using all training data at each node with different weights. The evaluation of the proposed method is presented on a set of machine-printed documents which have been annotated by multiple writers in an office/collaborative environment.

1. Introduction

In recent years, many information retrieval (IR) systems for machine printed text data have been designed based on advances in optical character recognition (OCR) and IR techniques. These successes have motivated larger efforts at converting documents into digital format. However, the retrieval of annotated machine printed documents which contain both machine printed and handwritten text is still a challenge.

In many applications, we may have a mixture of both machine printed text and handwriting within a single document and we may be interested in who signed a document or what was written on a document and retrieve documents annotated by a particular author or with a specific annotation. Handwritten text identification or separation of ink from mixed documents is a necessary prior step to achieving this objective.

Text identification, especially handwriting identification, can be traced back to the early work on extraction and recognition of handwritten ZIP codes from mail pieces [8]. Much of the research on handwriting identification follows document layout analysis and zone classification: a document is segmented into words, lines and zones [6, 5]. The locations of these zones point to where handwritten information can be found. There have been relatively fewer efforts at identification of handwritten text or ink from mixed documents. In [12], Zheng et al. proposed a two step approach to identify three different types of patches in mixed documents. By projecting each word horizontally, Guo and Ma separated handwritten material from documents using a hidden Markov model [4].

In the case of annotated documents, the imbalance between handwritten and machine printed text is a serious problem. Generally, the amount of machine printed text is much more than the handwritten annotations and will dominate in the data set. If even a small proportion of machine printed text is misclassified as handwritten text, it is still a significant number in comparison to handwritten samples and leads to relatively low precision [12] for identification of handwritten annotations. To overcome the imbalanced data set issue in a form classification problem, a tree-based classification algorithm was introduced in [11], which proposed a hierarchical classification model to classify different tax forms using binary Regularized Least Square classifiers and K-nearest-neighbor classifiers in each tree node. Other boost tree classifiers that are attracting research interest in the pattern recognition and computer vision communities can be extended to the field of document analysis as well[10, 9].

The imbalanced data set problem is amplified even further when we consider annotations that overlap machine printed text. In order to classify machine printed text, handwritten text and overlapped text (as shown in Fig 1) and overcome the imbalanced data set problem, we describe a tree-structured classifier which consists of binary weak classifiers. At each node, we convert a multi-class problem to a bi-class problem by merging all classes except the major class to a single class.

Over-fitting is a potential drawback of a tree-structured classifier because it considers only a subset of

1051-4651/10 $26.00 © 2010 IEEE
DOI 10.1109/ICPR.2010.68

training data at each node and loses the general distribution of the whole data set[3]. Inspired by the Adaboost algorithm which uses and updates the distribution of all training data in each round and is less subject to over-fitting , we propose a new mechanism to train the weak classifier for each node of the tree-structured classifier.

Section 2 introduces the structure of our tree-structured classifier and the procedure used for learning and testing. Section 3 shows the details of experiments, including feature extraction, experimental set up and results. Section 4 presents our conclusions.

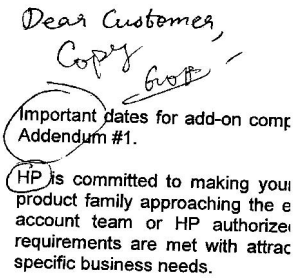

Figure 1. Samples of machine printed text, handwritten text and overlapped text

2. Tree-structured Classifier

The merit of the tree-structured classifier is that it balances the positive and negative samples by merging the classes in the lower level of the tree [11] and its training error can be made to decrease exponentially [3]. Normally, a divide-and-conquer strategy is used to build the tree. The underlying principle of this divide-and-conquer approach is that if current classifier cannot separate the training set with high accuracy, the set is separated into several smaller clusters and classifiers (may differ from previous classifier) are used for each new cluster. Thus, a tree-structured classifier recursively focuses locally on the data set as the tree grows deeper and can achieve high training accuracy. However, a node of a tree-structured classifier loses distribution information from the entire data set and is very susceptible to over-fitting.

2.1 Structure of the classifier

To preserve the advantage of tree-structured classifier and overcome the over-fitting problem, we design a boosting algorithm which has a tree structure and can be also viewed as a combination of cascade classifiers.

Assuming that there are n classes and a total number of m samples in the training data set χ, each sample is associated with a normalized weight w_i which measures the importance of the sample. Our tree-like classifier's job is to find a hypothesis $H : x_i \rightarrow y_i \in \Gamma = \{1, \cdots, n\}$, which maps sample x_i to a target label y_i in label set Γ.

Firstly, we define the structure of the tree-like classifier. Each node $N(i,j)$ of the tree corresponds to a weak classifier $h_{i,j}(\cdot)$, where i indicates the level of the node and j shows the index of the node in this level. Note that even an empty node is assigned an index number in the tree. Then the left child's index of a node $N(i,j)$ is $N(i+1, 2j)$ and the right child's index of node $N(i,j)$ is $N(i+1, 2j+1)$. Each leaf node in the tree corresponds to a target label.

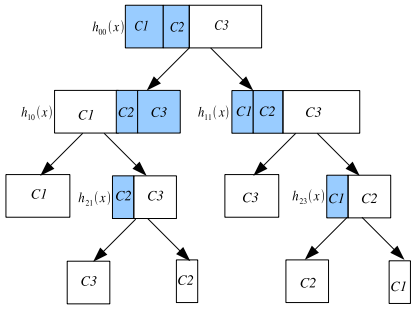

Figure 2. An example of tree-like classifier. In each node, minority classes are merged to a single class and represented as a blue rectangle. Majority class is represented as a white rectangle.

Unlike other algorithms that handle multi-class classification using multi-class weak classifiers at each node or each level, we propose to convert our multi-class classification problem to a two-class classification problem at each node. We will determine which class is the major class, and all other classes are considered as the minor class.

For a given node $N(i,j)$, the summation of the sample weights from each class measures the majority of this node.

$$m(i,j) = \arg\max_{t \in \Gamma} \sum_{k=0, y_k = t}^{m} w_k \qquad (1)$$

Fig.2 shows an example of the tree structure and how to convert multi-class to bi-class problem.

2.2 Learning & Testing Procedure

The normal recursive learning procedure is to split the source set of the parent node into subsets for child

nodes based on the parent node's classification test. Given the training set $\hat{\chi}$ for a parent node $N(i,j)$, each child node has a subset:

$$N(i+1,2j) : x \in \hat{\chi}^+ \mid h_{i,j}(x) < t$$
$$N(i+1,2j+1) : x \in \hat{\chi}^- \mid h_{i,j}(x) > t$$

where t is a threshold for weak classifier $h_{i,j}(x)$ and $\hat{\chi} = \hat{\chi}^+ \cup \hat{\chi}^-$ is satisfied.

As mentioned earlier, this splitting method is prone to over-fitting. Since Adaboost is less susceptible to over-fitting, we integrate a cascade decision mechanism which is similar to Adaboost's boosting algorithm into the tree-structured classifier to overcome the over-fitting problem. As noted in [2], the effect of updating the distribution of training data in Adaboost algorithm is to increase the weight of examples misclassified by h_t, and to decrease the weight of correctly classified examples. Thus, the weight tends to concentrate on "hard" examples.

So, rather than split the training data set and assign a subset to a child node, we propose to use all the training data at each node during recursive learning but assign different weights to them according to their attributes.

For a given parent node $N(i,j)$, we train the weak classifier $h_{i,j}(\cdot)$ using weighted training samples $\hat{\chi}$ with their weights $\hat{w}$ and calculate the threshold t for this node.

The weight of training samples at the left child node $N(i+1,2j)$ is updated as:

$$w_i = \begin{cases} \hat{w_i} & \mid h_{i,j}(x_i) < t \\ \alpha \cdot \hat{w_i} \cdot exp\{-(\frac{h_{i,j}(x_i)-t}{o_{max}-h_{i,j}(x_i)})^2\} & \mid h_{i,j}(x_i) > t \end{cases} \quad (2)$$

Similarly, the weight of training samples for the right child node is updated as:

$$w_i = \begin{cases} \hat{w_i} & \mid h_{i,j}(x_i) > t \\ \alpha \cdot \hat{w_i} \cdot exp\{-(\frac{t-h_{i,j}(x_i)}{h_{i,j}(x_i)-o_{min}})^2\} & \mid h_{i,j}(x_i) < t \end{cases} \quad (3)$$

where o_{max} and o_{min} are maximum and minimum output of $h_{i,j}(\cdot)$ of the parent node given the training samples. The weight for all training samples are initially set to be $\frac{1}{m}$ in our experiment.

The purity p of the node is used as our stopping criteria for each node $N(i,j)$ and updated according to the level of the node to avoid over-fitting as described in following equation:

$$\sum_{k=0,y_k=m(i,j)}^{m} w_k > p \cdot e^{-\frac{i}{\lambda}} \quad (4)$$

where i is the level of the node $N(i,j)$, p is a predefined constant purity threshold (may differ for different classes), λ is the parameter to control the convergence of training procedure and $m(i,j)$ is the majority of current node as defined in Equation 1. Equation 4 illustrates that if the ratio of majority is higher than the purity threshold, a leaf is achieved and its majority id is assigned as the target label of this leaf node.

The testing procedure is similar to the training phase which starts from the root node and achieves one of the child nodes according to the classification test at each parent node. The truth label of the test sample is assigned as the label of the leaf node which is achieved.

3 Experiments

The data used for the experiments is a set of 82 annotated office documents from the HP Labs data set which consists of binarized images scanned at a resolution of 300 dpi. This extremely imbalanced data set contains over 25000 machine printed text patches, about 3200 handwritten text patches and less than 400 overlapped text patches. We used 54 documents for training and the remaining for testing.

Prior to classification, a morphology closing operation was applied to merge small characters to patches which approximately represent words. These patches are the basic unit in our system. At each patch, we extracted patch level features, connected component features and Gabor features as described in [7]. The tree-structured classifier was built as described in Section 2. A modified public ANN tool, FANN [1], was used in our experiments using one hidden layer to train the binary weak learner for each node. The test samples were labeled using this classifier in the same manner.

We measured the performance of the proposed system using precision and recall metrics. Precision for machine printed text in our system is the ratio of patches which are correctly classified as machine printed text to all patches which are classified as machine print. Recall for machine printed text is the ratio of patches which are correctly classified as machine printed text to all machine printed patches in the test set. The same metrics were applied to handwritten text and overlapped text.

In table 1, we compared the proposed tree-structured classifier to a normal decision tree classifier which has a similar training phase but without using updated distribution for training samples, and a backpropagation Neural Network classifier which is implemented using FANN. We see that the Neural Network cannot identify handwritten text and overlapped text very well on this imbalanced set because it tends to focus on the majority class (machine printed text) and loses information about the minority classes (handwritten text and overlapped text). Normal decision tree can achieve

Table 1. The analysis of system performance

	BP Neural network		Normal DT		Proposed method	
	Precision (%)	Recall (%)	Precision (%)	Recall (%)	Precision (%)	Recall (%)
Machine-printed	99.29	95.07	99.40	97.84	99.61	98.74
Handwritten	77.48	89.94	91.04	90.66	92.89	93.67
Overlapped	33.54	77.48	40.61	80.30	57.29	83.33
Overall	N/A	87.50	N/A	89.60	N/A	91.91

slightly better identification performance because as the tree grows deeper, the possibility of focusing on minority classes increases as well. However, our proposed tree-structured classifier had an overall recall of 91.91% which outperformed backpropagation neural network (87.5%) and DT (89.6%) and also significantly increased the precision, especially for overlapped text.

The mis-classification in our system is mainly from machine printed symbols and punctuation marks such as comma and period which are classified as handwritten dots. The imbalanced data set problem which causes the low precision of overlapped text can be overcome by adding more such samples into the training set.

4 Conclusions

In this paper, we present a tree-structured classifier that can take advantage of the global distribution of training samples. The distributions (weights) of training samples are updated based on the classification test at each node during the training phase. Experiments show that the proposed method is more reliable for extremely imbalanced data sets when compared to normal decision tree and other classifiers.

References

[1] http://leenissen.dk/fann/.

[2] Y. Freund and R. E. Schapire. A short introduction to boosting. *In Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 1401–1406, 1999.

[3] E. Grossmann. Adatree: boosting a weak classifier into a decision tree. *Computer Vision and Pattern Recognition Workshop, CVPRW04*, June 2004.

[4] J. Guo and M. Ma. Separating handwritten material from machine printed text using hidden markov models. *Proc. Sixth International Conference on Document Analysis and Recognition*, pages 439–443, 2001.

[5] G. Nagy, S. Seth, and S. Stoddard. Document analysis with an expert system. *Pattern Recognition in Practice II*, pages 149–155, 1984.

[6] L. O'Gorman. The document spectrum for page layout analysis. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 15(11):1162–1173, Nov 1993.

[7] X. Peng, S. Setlur, V. Govindaraju, R. Sitaram, and K. Bhuvanagiri. Markov random field based text identification from annotated machine printed documents. *Proc. IEEE 10th International Conference on Document Analysis and Recognition*, pages 431–435, 2009.

[8] S. N. Srihari, V. Govindaraju, and A. Shekhawat. Interpretation of handwritten addresses in u. s. mailstream. *Proceedings Second International Conference on Document Analysis and Recognition*, pages 291–294, 1993.

[9] Z. Tu. Probabilistic boosting-tree: learning discriminative models for classification, recognition, and clustering. *Proc. 10th IEEE International Conference on Computer Vision ICCV*, 2:1589–1596, 17-21 Oct 2005.

[10] B. Wu and R. Nevatia. Cluster boosted tree classifier for multi-view, multi-pose object detection. *Computer Vision, ICCV, IEEE 11th International Conference on*, pages 1–8, Oct 2007.

[11] J. Xu, V. Singh, V. Govindaraju, and D. Neogi. A hierarchical classification model for document categorization. *Proc. 10th IEEE International Conference on Document Analysis and Recognition*, pages 486–490, Aug 2009.

[12] Y. Zheng, H. Li, and D. Doermann. Machine printed text and handwriting identification in noisy document images. *IEEE J PAMI*, 26(3):337–353, 2004.