Boosted Decision Trees for Word Recognition in Handwritten Document Retrieval

Nicholas R. Howe
Dept. of Computer Science
Smith College
Northampton, MA-01063
nhowe@email.smith.edu

Toni M. Rath*
Dept. of Computer Science
University of Massachusetts
Amherst, MA-01003
trath@cs.umass.edu

R. Manmatha*
Dept. of Computer Science
University of Massachusetts
Amherst, MA-01003
manmatha@cs.umass.edu

ABSTRACT

Recognition and retrieval of historical handwritten material is an unsolved problem. We propose a novel approach to recognizing and retrieving handwritten manuscripts, based upon word image classification as a key step. Decision trees with normalized pixels as features form the basis of a highly accurate AdaBoost classifier, trained on a corpus of word images that have been resized and sampled at a pyramid of resolutions. To stem problems from the highly skewed distribution of class frequencies, word classes with very few training samples are augmented with stochastically altered versions of the originals. This increases recognition performance substantially. On a standard corpus of 20 pages of handwritten material from the George Washington collection the recognition performance shows a substantial improvement in performance over previous published results (75% vs 65%). Following word recognition, retrieval is done using a language model over the recognized words. Retrieval performance also shows substantially improved results over previously published results on this database. Recognition/retrieval results on a more challenging database of 100 pages from the George Washington collection are also presented.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Retrieval Models; I.7.5 [**Document and Text Processing**]: Document CaptureOptical character recognition (OCR)

General Terms

Algorithms, Measurement, Experimentation

*This work was supported in part by the Center for Intelligent Information Retrieval and in part by the National Science Foundation under grant number IIS-9909073. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor.

SIGIR'05, August 15–19, 2005, Salvador, Brazil.

Keywords

Handwriting retrieval, historical manuscripts, adaboost, decision theory

1. INTRODUCTION

One of the great challenges of the 21st century is to make all the information in the world accessible. Google and other search engines have tried to do this for the web and more recently are attempting to do this for printed books. However, providing access to the large archives of historical handwritten material is a challenge - the recognition and retrieval of *offline* handwritten material has still not been adequately addressed. Many such archives exist ranging from the Presidential papers of George Washington at the Library of Congress to Isaac Newton's manuscripts at the University of Cambrige Library [23]. Such collections may often be large - for example George Washington's collections at the Library of Congress exceed 140,000 pages.

Current handwriting recognition technology works in constrained domains like postal address recognition/bank check recognition but recognition rates for large vocabulary material are much lower. While this is true for modern documents, the challenge is even greater for historical handwritten manuscripts where the pages can suffer from many problems like ink bleeding and dirt on the paper or in the scanned images.

Rath et. al. [23] recently proposed the first known automatic handwriting retrieval system. Their approach used relevance models. This paper proposes an alternative solution using handwriting recognition followed by retrieval using language modeling. The recognition scheme proposed here involves a number of novel ideas for handwriting recognition/retrieval. These include the use of pixels in normalized word images at multiple scales (image pyramids) as features and the use of a powerful combination of machine learning techniques - specifically decision trees and AdaBoost. An innovative approach to creating additional training data for improving performance is also proposed. Results on the publicly available dataset used in Lavrenko et. al. [12] show that a substantial increase in recognition performance (75% vs 65%) is obtained using the approaches described here. Substantial improvements in retrieval performance over those presented by Rath et. al. [21] are also obtained. Further results are presented on a larger and more challenging database of 100 pages of George Washington's manuscripts.

Pixels in image pyramids of normalized word images form the features. This is unlike many word recognition schemes

[18, 10, 15, 12] which need to compute a number of different kinds of features before recognition. Eliminating the need to extract and compute features removes one potential source of error, but can also make the recognition problem itself harder. As a special case, consider systems that attempt to recognize individual letters and build words out of them [18]. With this approach, segmenting the letters into recognizable units becomes the limiting step. By contrast, segmentation of individual word images is easier [14] (though not trivial), and the primary challenge becomes an image classification problem. Fortunately, focusing on classification of word images in relatively untouched form means that powerful machine learning techniques may be brought to bear: boosted decision trees, in this case.

Boosting is a so-called *ensemble* classification technique, meaning that it determines its prediction via the weighted vote of a diverse set of *base classifiers*, each of which has been trained on a different weighting of the training data. The experiments use AdaBoost [7] because it is one of the oldest and best understood boosting algorithms. AdaBoost trains successive versions of its base classifier on differently weighted combinations of the training data, progressively focusing as it goes on the harder-to-classify examples. The weighted vote of the full set of classifiers developed typically exhibits predictive accuracy much greater than any individual classifier in the ensemble. Although AdaBoost can use "simple" base classifers, in practice a relatively powerful base classifier such as a decision tree usually generates correspondingly stronger results.

One of the challenges in modeling handwritten material is the skewed distribution of class frequencies (the words follow a Zipfian distribution) and the consequent paucity of training data for most word classes. We show how the training data can be augmented using stochastically distorted copies of the original data to substantially improve performance.

The rest of the paper is structured as follows. Related work is discussed in Section 2. The word classification algorithms are discussed in Section 3. Section 4 presents the experimental results for recognition. The retrieval approach and the retrieval results are presented in Section 5. Section 6 concludes the paper.

2. RELATED WORK

While recognition rates for printed material in standard fonts are high, the problem of off-line handwriting recognition is still a difficult one. (The on-line handwriting problem is easier because of access to information about pen velocity and position.) The difficulty arises from the fact that print is very consistent while there is much more variation in handwriting. For domains where either the vocabularies are limited (check processing) or there are constraints (postal address sorting), handwriting recognition has been well researched and recognition rates are fairly high [18].

Large vocabulary recognition for both modern and historical documents is, however, a challenging problem. Even for modern high-quality document images created specifically for recognition experiments, current state-of-the-art recognizers often have word error rates in excess of 50% [15, 26]. Historical manuscripts are even more challenging because of the many problems that occur including ink fading, ink bleeding, shine through and blotches; consequently there is little work in this area. The 20 page set used in one of these experiments is a publicly available dataset. The authors introducing that dataset [12] used word level HMMs to get an average word error rate of about 60% without bigrams and up to about 40% using bigrams from external corpora. Similarly high error rates for historical documents have been reported by others [25].

There have been two other approaches to indexing and retrieving historical handwritten manuscripts. One approach called word spotting involves creating an index analogous to the index at the back of a book by using word matching techniques [13, 22]. A number of different matching techniques for this problem were investigated in [22], including dynamic time warping of 1D features and shape context matching. While reasonable matching can be achieved, the techniques are very computationally expensive and it is currently impossible to build a system for even a small number of pages within a reasonable amount of time.

A second approach [23] involves using relevance-based language models to annotate word images with probabilities and then using a language model approach to retrieve page images. This approach has produced mean average precision values ranging from 53% for one word queries to about 84% for four word queries [21] on the same dataset.

Previous recognition and retrieval approaches have usually created different kinds of features to characterize a word image which are then used for recognition [12] or retrieval [23]. Our approach here is significantly different in that no features are explicitly computed. Instead, the recognition process uses values sampled directly from the word image at varying resolutions.

The combination of boosting with decision trees has been used previously in machine learning research [5], but has not to our knowledge been applied before for handwritten word recognition. Indeed, even boosting has only recently begun to see widespread use in image processing. Notable applications are to content-based retrieval of photographic images [24, 9] and for object detection [27, 4].

3. CLASSIFICATION ALGORITHM

The boosted word classification algorithm works directly with the pixel representation of the word image rather than higher-level features that must be extracted. Handwritten words belonging to a single class have similar but not identical ink distributions. In particular, the relative positions of individual features within the word (letters and letter segments) will shift from example to example, making straightforward use of the pixel representation ineffective. Nevertheless, the pixel representation contains information about word identity that can be amplified by boosting. Figure 1 shows a composite image of 21 examples of the word *Instructions*. Blurring indicates areas of inconsistency, yet some features can also be clearly distinguished. If each pixel in the image is taken as a potential feature for classification, then the clearer areas will contain more reliable features. Heuristics such as information gain can help to identify which features are reliable, and boosting can amplify their effectiveness.

3.1 A Common Framework

In order for pixels to serve as features for word image classification, each word image must be mapped onto a common pixel grid. For simplicity, assume that all word images are initially scaled and translated in such a manner that their horizontal midline (computed as described below) spans the

Figure 1: A composite image created from 21 superimposed versions of the word *Instructions*.

Standard Φ_0 Φ_1

Figure 2: A standard image, and grids Φ_0 and Φ_1.

interval from the origin to $(1, 0)$ on the Cartesian plane. (Call this the *standard image*.) Resampling from each standard image at some uniform grid of points will produce a common pixel representation, as required. The composite image in Figure 1 was produced by taking the average of 21 resampled standard images.

Variations in word length and aspect ratio present some difficulty in choosing the resolution and extent of this grid. Long words require high resolution to capture all their details, but tend not to extend far in the vertical direction. Short words do not require high resolution but do cover greater vertical extent. Unfortunately, accommodating both extremes at once requires a grid of both high resolution and large vertical extent, resulting in astronomical data sizes.

A pyramid approach provides the answer. We define a family of standard grids, beginning with a base grid Φ_0 covering the square region $([0, 1], [-0.5, 0.5])$ and broken into 32×32 pixels.[1] Successively refined grids cover the same square region with double the linear resolution, e.g., Φ_1 has 64×64 pixels. Figure 2 shows an example. The grid pixels themselves can be thought of as being organized in a tree-like structure, where each grid point in Φ_k has four children in Φ_{k+1}.

The pyramid of grids can be represented efficiently in two ways. First, the standard image usually will not cover the full vertical extent of the grid. Portions above and below the edges of the standard image may be represented using a single default value (e.g., white). Second, data need only be stored for Φ_k with resolution up to that of the reference image. If $\hat{k}$ denotes the index of the highest resolution stored, then values of $\Phi_k(i, j)$ for $k > \hat{k}$ may be computed according to Equation 1 below. The value of $\hat{k}$ will vary for each standard word image, but range in the experiments from one to six with a median at four.

$$\Phi_k(i, j) = \Phi_{\hat{k}}\left(\left\lceil \frac{i}{2^{(k-\hat{k})}} \right\rceil, \left\lceil \frac{j}{2^{(k-\hat{k})}} \right\rceil\right), \text{ for } k > \hat{k} \qquad (1)$$

[1] Although a few words have aspect ratios taller than they are wide, this square area captures all the detail of interest for most words.

3.2 Boosting and Decision Trees

Before going further, a brief review of AdaBoost is in order, with specifics about its application to word images. Boosting requires a base classifier capable of learning to classify arbitrarily weighted training data with an accuracy of at least 50%. ("Arbitrarily weighted" means that the contribution of each training example to the overall accuracy can vary relative to the others, as dictated by the needs of the boosting algorithm.) In two-class problems almost any classifier will do, but situations with many potential classes (e.g., word image recognition) present more limited choices because there are more ways to be wrong. Decision trees present themselves as the foremost option because they are well understood and can achieve arbitrary accuracy on the training data in practice. (At each node of the decision tree, the training examples are split into two smaller subgroups by comparing the value of a chosen pixel location in each to a chosen threshold. The subnodes can further split the data until each leaf contains exactly one training example, which can then be assigned the correct label.)

Of course, a tree trained to 100% accuracy overfits the data and generalizes poorly to new examples. Decision tree algorithms typically include steps to prune overfitted trees by removing branches statistically unlikely to generalize well. Less precisely, one can simply terminate growth of the decision tree early, before 100% accuracy is reached. In experiments, the latter approach has proved more efficient and may actually produce slightly better results. Accordingly, we stop the growth of a tree branch when the subset of training examples it contains is dominated by a majority class (thus ensuring that the training set error will be less than 50%).

C4.5 [20] provides the algorithm for building the decision tree, with some modifications designed to support the grid pyramid data structure described above. When building each node of the decision tree, a feature (i.e., pixel location) and threshold value must be chosen as the split criterion. Examples are split at the node depending on whether their value for the chosen feature is less than the threshold or not. C4.5 exhaustively examines all possible features and split points to determine the best criterion. With the grid pyramid, fully exhaustive search is not feasible. Instead, only Φ_0 is examined exhaustively, and the location and threshold offering the greatest information gain is retained. The search then proceeds selectively to its children in Φ_1, from there to the children of the best of those locations, and so on until the maximum resolution available has been reached. At the end, the grid level, location, and threshold with the highest information gain becomes the decision criterion for the node. For both generalization accuracy and speed, only a set number of thresholds are considered as possible decision criteria (7 in the experiments below).

As the experiments below make clear, single trees do not generalize well for handwritten word images, even with pruning. The boosting stage raises generalization accuracy considerably. AdaBoost proceeds in a series of rounds. Initially, a base classifier is generated normally from the training data. If not overtrained, it will classify some of the training examples correctly, and misclassify others. AdaBoost adjusts the succeeding rounds of boosting by raising the weights of the misclassified items and lowering the weights of the correctly classified ones. This has the effect of forcing the base classifier to work harder to get a correct result for the

Original Warped Overlay

Figure 3: A word image, a resampled version, and an overlay of the two (for comparison).

items that were previously misclassified. After many rounds of boosting, a weighted vote of the classifiers from all the rounds classifies the training set perfectly and shows good generalization to unseen examples.

The number of rounds of boosting may be set via cross-validation on subsets of the training data. In practice, beyond a certain point the results are not very sensitive to the number of rounds [8]. Therefore, the experiments in this paper train consistently for 200 rounds, which appears more than adequate in cross-validation tests.

3.3 Supplementary Training Examples

Among other factors, the paucity of training examples for many classes makes generalization difficult. Zipf's law [28] implies that most of the words in any natural training set will make only a few appearances. For example, in the GW20 data set described in the next section, 681 of the 1187 (57%) word labels appear only once, and another 183 appear twice. Only 179 appear more than five times, but the most common word ("the") appears 238 times.

The skewed distribution of training examples can work against accurate classification in two ways. Many algorithms for machine learning implicitly assume that class frequencies are roughly comparable. Even when they do not, it is very difficult to learn a class from a single exemplar, because no information on variance can be directly obtained. A body of work on *one-shot learning* [6] has attempted to address this shortfall by applying variance models learned from the high-frequency classes. Although this paper does not use a variance model learned from the training data, the experiments in the next section do generate new training examples for low-frequency classes via stochastical distortions of the available examples. While this approach may not capture all the variance that would be present in actual data, it does help to improve the overall word classification accuracy.

To generate a distorted word image, one can sample from the original using a grid of points whose positions have been perturbed from a uniform lattice. For greater realism, nearby points should be perturbed by similar amounts. Starting with the uniform lattice, initial random perturbations along each coordinate axis are smoothed to yield the desired spatial coherence at a chosen scale. Further global adjustments ensure that the borders of the warped grid are rectangular. Sampling from the resultant warped grid yields a new word image that is a distorted version of the old, as shown in Figure 3.

The experiments described in the next section use this process to generate additional training examples that vary slightly from the originals. The number of times each training example must be duplicated depends upon the number of other examples of its class present in the training set. The number of duplicates is chosen so that in the end there are at least eight examples of each class. So for example, a single example of a class is duplicated seven times, a pair of examples three times each, a triplet twice each, and so on. Word classes with eight or more examples in the training set require no duplication at all.

Table 1: Results (accuracy) of word image classification experiments on the GW20 data set compared with the previous best results in [12] Note that our technique does not exploit bigram statistics.

Experiment	All words	Excluding OOV
I. Decision tree	31.1 ± 4.6 %	36.8 ± 4.6 %
II. Boosted trees	51.1 ± 5.2 %	61.6 ± 4.6 %
III. Added examples	63.6 ± 5.3 %	75.2 ± 4.1 %
IV. Previous best with bigrams	55.1 ± 7.0%	65.1 ± 6.0%
V. Previous best without bigrams	49.7 ± 7.0%	58.6 ± 6.0%

4. CLASSIFICATION EXPERIMENTS

Tests of the word classification system use pages of handwritten text from the letters of George Washington at the Library of Congress. 20 pages from this source are publicly available and form the GW20 set. GW20 was previously used for handwriting recognition [12] and for line retrieval [21]. An additional 100 pages not overlapping with GW20 form the GW100 test set. Both sets include words written by multiple hands because Washington employed secretaries. Each page in the data sets was manually segmented to extract images of individual words (4856 in GW20 and 21324 in GW100). All of the word images were labeled by an annotator with their ASCII equivalent. The labeling is used as the ground truth for our recognition and retrieval experiments.

The GW20 data form the basis for a set of experiments on word classification rates. Each experiment adopts a cross-validation framework, with each fold using nineteen pages for training and a single page for testing. As a control, the first experiment employs a single decision tree for word recognition. (Unlike the trees used for boosting, these are standard C4.5 trees, grown to completion and then pruned. However, the majority-trees used elsewhere produce similar results.) Following that, the second experiment uses AdaBoost with decision trees as the base learner. Experiment three augments the original training data with additional modified training examples, as described in Section 3.3. (Note that there is no experiment using AdaBoost with a base classifier simpler than decision trees, because the threshold of 50% accuracy required for the base classifier cannot be achieved by simple means.)

Table 1 shows the results of these experiments. The first column of numbers gives the percentage of correct classifications for each experiment (mean and standard deviation taken over all the folds) using all available test words. The second column gives the classification rate excluding out-of-vocabulary *OOV* words. OOV words are those for which no training example is available in a particular fold, and which consequently cannot be classified correctly by supervised learning algorithms.

As expected, AdaBoost substantially improves classification accuracy as compared to the base classifier (a decision tree). More interestingly, the addition of artificially generated examples to the training set yields significant further improvement. Despite the fact that the new examples added are merely modified versions of ones already present in the training set, their inclusion appears to promote better generalization in the final boosted classifier. The performance on the final experiment exceeds previously published results on the same data [12].

Following the results above, a boosted decision tree classifier is trained on the full GW20 data set, with artificial training examples added in the sparsely represented classes. In tests on the larger GW100 set, this classifier achieves 44% accuracy over all words, or 59% when OOV terms are excluded. (Decreased image quality and a greater prevalence of OOV words explain the lower performance on GW100 as compared to GW20.) The predictions of this classifier may then be used for retrieval purposes on the GW100 set.

5. RETRIEVAL

This work adopts the language modeling approach to retrieval [19, 11, 23]. In particular the retrieval experiments use the query likelihood formulation, where the documents are ranked according to the probability of the query given the document i.e. $P(Q|D)$. AdaBoost provides classification decisions rather than probabilities, meaning that only the most likely label for each word image is preserved. (Although the AdaBoost algorithm generates a score for each potential word label to be used for classification, these scores provide low correlation with class probabilities [16].) Given these strictures, we use two approaches for retrieval and estimation of the probabilities of different words.

The first approach simply assumes that a text document contains all the recognized outputs (i.e., no classification errors occurred). Invoking maximum likelihood, the probabilities of different words are estimated to be equal to their frequencies in each recognized document. (Smoothing with the collection frequency is not observed to change the results much, so is omitted.) Obviously, this approach ignores the fact that some of the words may have been misclassified and hence the frequency estimates may be incorrect. The problem becomes more pronounced as the word error rate increases, spawning a need to better assess the uncertainty in the AdaBoost classification. Because the scores computed by AdaBoost are not completely reliable for use as probabilities, we turn for the second approach to a regularization scheme based upon classification rank information.

Researchers in both text [2, 3] and image retrieval [17] have noted that rank information may be more useful than actual probabilities. The key insight is that a few top terms are most important, some are moderately important and the rest are irrelevant. By fitting a Zipfian distribution to the retrieval terms they obtain a set of probabilities that yield good results. In a similar vein, we infer a probability distribution from the rank ordered output of the AdaBoost classification algorithm. In other words, AdaBoost assigns a given word image a score for each possible class in the lexicon. Normally, the word image is then assigned to the class with the highest score (as in the first approach). For the second approach, we instead rank the top n classes according to the scores. The scores are discarded and the rank order is preserved. The probabilities for these classes are determined by fitting it to a Zipfian distribution by assigning probabilities

$$P(r) = Z\frac{1}{r}$$

to the annotation term at rank r, where Z is a normalization constant to ensure P is a probability distribution. Instead of the document having one possible word at each image position, the document now contains a probability distribution at each position. These distributions are used to estimate the actual term probabilities. That is, the document models are now the average of all per-word-image annotation distributions. The experiments show improved performance using this approach on GW100, where the overall word error rate is higher.

The AdaBoost classification outputs on the GW20 and GW100 sets were used as input to the *Lemur* toolkit [1], a software package which implements various retrieval methods that are mostly in the language modeling spirit. We evaluated the retrieval performance on our automatically classified test collections with the query-likelihood ranking method. The experiments performed include 20 cross-validation runs using 19/1 train/test splits of the GW20 set, plus two experiments which use GW20 for training with retrieval evaluation on GW100.

Because of the limited size of GW20 (4856 images), we perform line retrieval. GW20 consists of 657 lines, which are randomly split into 10 disjoint parts, as in previous work [21]. Each cross-validation run uses one part as the testing set, with the remainder forming the training set. For evaluation purposes, a line is considered relevant to a query if it contains all of the query terms. (Page retrieval also makes use of this principle in later experiments.) The experiment includes all 1- to 4-word queries consisting of terms with training data available, and for which at least one relevant item exists in the testing set. Stop words are removed, except from the 1-word queries, for consistency with the query selection technique used in previous work [21].

The GW100 dataset is large (21324 word images) and allows for tests of full page retrieval using GW20 as the training set. Again, we extract all 1- to 4-word queries which occur in the testing set, but also in the training set (always excluding stop words). However, because of the very large number of queries, the longer query sets are limited to somewhat less than 2000 members by choosing only the queries with the most relevant items in the test set.

Table 2 shows the mean average precision results obtained on the two data sets. GW20-ta and GW100-ta use only the top-ranked annotation for each word image, essentially performing retrieval on top of an automatic transcription. The results for GW20 are averaged over 10 cross-validation runs. Retrieval on GW20 performs very well, substantially better than previously reported results on the same data set [21] , which were 54%, 63%, 78% and 89% for 1- to 4-word queries respectively. It should be pointed out, however, that the results in [21] were obtained with automatically segmented pages, so slightly worse performance is to be expected.

The results of the GW100-ta run are significantly lower (also see recall precision graph in Figure 4), because the task is much harder. Numerous factors combine to make lower recognition/retrieval rates likely, even expected. The ink in the word images is much more uneven, with fading evident in many words. The training/test split of 1/5 is far more challenging than the 19/1 split for the GW20 collection. Another factor may be the larger vocabulary size of the GW100

Table 2: **Retrieval results (average precision in percent) on the GW20 and GW100 data sets, and number of queries run on each set. For the GW20 previous best results are those reported in [21]. For GW100, retrieval was performed using only the top annotation term for each word image (GW100-ta) and using the top 20 annotations with probabilities (GW100-pa).**

Test run	1 word	2 words	3 words	4 words
GW20-ta	79.53%	80.17%	86.48%	90.04%
# Queries	1869	1348	708	210
Previous best GW20	54%	63%	78%	89%
GW100-ta	12.59%	27.98%	37.71%	39.57%
GW100-pa	20.33%	30.67%	37.24%	38.40%
# Queries	544	1874	1777	1557

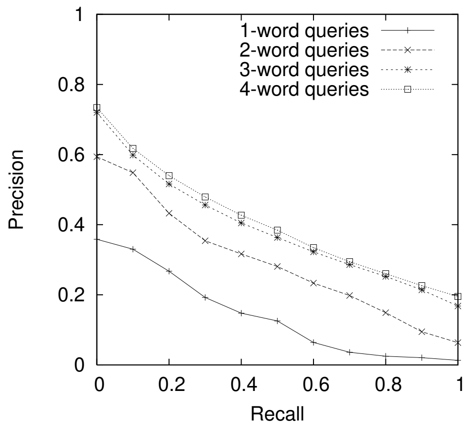

Figure 4: Recall-precision graph for the GW100 data set. Only the top-ranked annotation of each word image was used.

set when compared to GW20, making classification more error-prone. More than a quarter (25.7%) of the vocabulary in GW100 does not appear in GW20. Other reasons include the larger number of retrieval units (100 pages in GW100 vs. roughly 66 lines in GW20) and the inclusion of greater handwriting variation in GW100. (GW100 is less temporally coherent than GW20, and handwriting varies over time.) Given all of these factors, the lower GW100 results are not surprising, especially considering that the performance of the best text retrieval engines on text databases does not exceed an average precision of 50%.

With the higher word error rate on this test set, the hard categorization done by AdaBoost becomes an impediment to retrieval. To improve performance we now use the second approach for estimating term probabilities using ranks described above. In particular we estimate the probabilities for the GW100 collection for the top 20 ranks for each word image.

This procedure achieves significantly higher performance for 1-word and 2-word queries, but about the same or slightly worse for 3- and 4-word queries (see run GW100-pa in Table 2). Three and four word queries already benefit from the redundancy provided in the longer queries which probably explains why their performance does not increase. This result shows clearly how retrieval of automatically recognized objects can benefit from probabilistic annotation. The key is to avoid hard (and potentially wrong) classification decisions and to retain information about alternative categories when the class predictions are of dubious quality.

6. CONCLUSION

Although this work uses standard machine learning techniques to tackle the word recognition problem, several points deserve notice because they highlight special features of the word recognition domain. First, most learning algorithms are not designed to deal with training data that exhibits the highly skewed distribution of class frequencies seen here. The technique used to mitigate this situation (synthetic variation on existing data) does not always work because the synthetic training examples are not truly independent of the originals. Yet the technique appears highly effective for word images. Second, most applications eschew point samples as features because individual pixels in most images do not correlate well with semantic classes. The success of point features in this case may be attributed to careful alignment of the word images, consistency of handwriting style, and the ability of boosting to act as a feature selector.

While the recognition/retrieval rates obtained for the GW20 collection are very good, it is clear that the problem becomes more challenging when the datasets are larger, more noisy and the training/test splits are changed to use small amounts of training as in the GW100 collection. Using soft classification decisions can improve the performance for queries of smaller length.

Many of the problems with the GW100 collection may simply be due to the more challenging image processing required. We believe that the approaches described here can be modified to substantially improve the performance on this dataset along with improvements in image processing. Machine learning and information retrieval techniques will play an important part in this endeavour (as here). This will take us closer to the day when all the world's material can be indexed.

7. ACKNOWLEDGMENTS

We would like to thank the Library of Congress for providing the digitized manuscript images that were used in this work.

8. REFERENCES

[1] The lemur toolkit for language modeling and information retrieval, 2005. available at http://www-2.cs.cmu.edu/~lemur/.

[2] V. N. Anh and A. Moffat. Robust and web retrieval with document-centric integral impacts. In *Proc. 2003 Text Retrieval Conference*, November 2003.

[3] V. N. Anh and A. Moffat. Collection-independent document-centric impacts. In *Proc. Australasian Document Computing Symposium*, pages 25–32, 2004.

[4] V. Athitsos, J. Alon, S. Sclaroff, and G. Kollios. Boostmap: A method for efficient approximate similarity rankings. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 268–275, 2004.

[5] L. Breiman. Arcing classifiers. *The Annals of Statistics*, 26(3):801–849, 1998.

[6] L. Fei-Fei, R. Fergus, and P. Perona. A bayesian approach to unsupervised one-shot learning of object categories. In *International Conference on Computer Vision*, pages 1134–1141, 2003.

[7] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the Thirteenth International Conference on Machine Learning*, pages 148–156, 1996.

[8] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *The Annals of Statistics*, 28(2):337–407, 2000.

[9] N. R. Howe. A closer look at boosted image retrieval. In *International Conference on Image and Video Retrieval*, pages 61–70, 2003.

[10] G. Kim, V. Govindaraju, and S. N. Srihari. Architecture for handwritten text recognition systems. In S.-W. Lee, editor, *Advances in Handwriting Recognition*, pages 163–172. World Scientific, 1999.

[11] J. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieva. In *the 24th annual international ACM SIGIR conference*, pages 111–119, 2001.

[12] V. Lavrenko, T. Rath, and R. Manmatha. Holistic word recognition for handwritten historical documents. In *Proc. of the IEEE Workshop on Document and Image Analysis for Libraries DIAL'04*, pages 278–287, 2004.

[13] R. Manmatha and W. B. Croft. Word spotting: Indexing handwritten manuscripts. In M. Maybury, editor, *Intelligent Multi-media Information Retrieval*, pages 43–64. AAAI/MIT Press, 1997.

[14] R. Manmatha and N. Srimal. Scale space technique for word segmentation in handwritten manuscripts. In *In the Proc. of the Second International Conference on Scale-Space Theories in Computer Vision (Scale-Space'99)*, pages 22–33, Sep. 1999.

[15] U.-V. Marti and H. Bunke. Using a statistical language model to improve the performance of an hmm-based cursive handwriting recognition system. *Int'l Journal of Pattern Recognition and Artifical Intelligence*, 15(1):65–90, 2001.

[16] D. Mease, A. J. Wyner, and A. Buja. Boosted classification trees and class probability/quantile estimation. Technical report, submitted to Journal of Machine Learning Research, 2004.

[17] D. Metzler and R. Manmatha. An inference network approach to image retrieval. In *Proc. International Conference on Image and Video Retrieval (CIVR-2004)*, pages 42–50, July 2004.

[18] R. Plamondon and S. N. Srihari. On-line and off-line handwriting recognition: A comprehensive survey. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(1):63–84, 2000.

[19] J. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *the 21st annual international ACM SIGIR conference*, pages 275–281, 1998.

[20] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[21] T. Rath, V. Lavrenko, and R. Manmatha. A statistical approach to retrieving historical manuscripts without recognition. Technical report, Center for Intelligent Information Retrieval technical report MM-42, 2003.

[22] T. Rath and R. Manmatha. Word image matching using dynamic time warping. In *Proceedings of CVPR'03*, volume 2, pages 521–527, 2003.

[23] T. Rath, R. Manmatha, and V. Lavrenko. A search engine for historical manuscript images. In *The 27th Annual international ACM SIGIR Conference*, pages 369–376, 2004.

[24] K. Tieu and P. Viola. Boosting image retrieval. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume I, pages 228–235, 2000.

[25] V.Govindaraju and H. Xue. Fast handwriting recognition for indexing historical documents. In *Proc. of the Int'l Workshop on Document Image Analysis for Libraries (DIAL)*, pages 314–320, 2004.

[26] A. Vinciarelli, S. Bengio, and H. Bunke. Offline recognition of unconstrained handwritten texts using hmms and statistical language models. *IEEE Transactions PAMI*, 26(6):709–720, 2004.

[27] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 511–518, 2001.

[28] G. Zipf. *Human Behaviour and the Principle of Least-Effort*. Addison-Wesley, Cambridge, MA, 1949.