Distributed Classifier Training for Large Scale OCR

Thomas M. Breuel
DFKI and U. Kaiserslautern
www.iupr.org

1 Introduction

OCRopus (`www.ocropus.org`) is a new open source OCR system targeted at large books scanning and digital library applications, sponsored by Google for use in the Google Book system. Development started in 2007, with a beta release planned for April 2008. It is based on an earlier handwriting recognition system for U.S. Census forms .

OCRopus currently contains two character recognizers (experimental HMM, shape-based, and scanning neural network recognizers are also being developed but will not be discussed here). The first is based on the Tesseract OCR engine and works by nearest neighbor classification based on a carefully constructed shape matching algorithm in a two-stage hierarchical search . Tesseract returns similarity scores but no posterior probabilities.

The second character recognizer is based on oversegmentation followed by estimation of both a segmentation cost $P(S)$ and posterior probabilities $P(c|x)$. OCRopus inherited a simple single hidden layer neural network classifiers whose input consists of a manually constructed feature map. When trained correctly, such networks give an estimate of $P(S)P(c|x)$, that is, a product of the segmentation cost and the posterior probability. Although this simple architecture has been sufficient for handwriting recognition, it does not scale up to the large number of classes and training examples found in OCR applications. A recent 1000 book data release by Google contains upwards of 5×10^9 training samples, and book scanning projects are aiming for upwards of 20 million books.

The requirements for a character classifier for the OCRopus OCR system are the following:

- it should be able to estimate both segmentation costs and posterior probabilities
- it should allow both parallelizable and distributed training; that is, parts of the model should be trainable independently, without knowledge of each other, and then composable into larger models
- it should support rapid per-book adaptation (style modeling)
- character classification needs to be fast on average

Methods have been proposed for the literature for addressing each of these issues separately. For example, fast classification (on average) can be achieved with boosting and cascading clssifiers (e.g., [1]), style adaptation can be achieved using mixture models (e.g., [3]), learning linear combinations of existing models, or hierarchical Bayesian methods, and several machine learning methods can be parallelized (e.g., [2]). There has also been extensive work on transfer of knowledge from one classification task to another. The models we are considering are also related to neural decision trees and hierarchical mixtures of experts.

2 The Model

The basic idea behind the redesigned character recognition component in OCRopus is the following: whenever we estimate a classifier $\tilde{P}(c|x)$, we also separately estimate the sample distribution $\tilde{P}(x)$. In a parallel, distributed training setting, each distributed learner receives a training batch D_i and then returns the estimates $\tilde{P}_i(c|x)$ and $\tilde{P}_i(x)$ for that training batch. Having an estimate of the sample distribution allows us to use and combine the individual classifiers in a number of different ways:

- $\tilde{P}(c|x)$ and $\tilde{P}(x)$, via Bayes rule, give us an estimate of $P(x|c)$, and thereby also contains all the information necessary for evaluating the segmentation cost $P(S)$. But since $\tilde{P}(c|x)$ is trained separately from the model of the sample distribution, classification is not constrained and limited in the same way is is for generative classifiers (e.g., MDA or mixture densities).

- Multiple classifiers that have been trained in a distributed way on subsets of the data with different sample distributions can be combined by forming the mixture density $\tilde{P}(c|x) \propto \sum_i \tilde{P}_i(x) N_i \tilde{P}_i(c|x)$ and a sample distribution of $\tilde{P}(x) \propto \sum_i N_i \tilde{P}_i(x)$ (after normalization). Note that often only a small number of terms are going to make significant contributions to these sums, so that this approach is also similar to a two-stage classification procedure.

- Style-conscious classification can be carried out in several different ways. Given a new batch B of samples to be classified, the simplest style-conscious classifier is to classify using $\tilde{P}_i(c|x)$ where $i = \arg\max_i \tilde{P}_i(B) = \arg\max_i \prod_{x_j \in B} \tilde{P}_i(x_j)$. A better style-conscious classifier is to form a mixture based on the likelihood of B using each model $\tilde{P}_i(B)$.

- Training batches D_i can be selected in different ways. For OCR on books, the transcribed and aligned characters of each book form a natural training batch. But training batches can also be derived in other ways: (1) the training batches can be generated using a boosting-like algorithm, modeling the distribution of the remaining misclassified samples after each boosting step and (2) the training batches can be split by classes. In each case, although the training methods are different, the integration of the samples into a final classifier is based in the same way on the estimates of the sample distribution.

3 Data and Initial Results

We have implemented distributed training based on the above model in Numerical Python, using Gaussian mixture models with diagonal covariance matrices for the estimation of each batch's sample distribution $\tilde{P}_i(x)$ and using MLPs for estimating $\tilde{P}_i(c|x)$. For the training data, we have used a subset of 20 books ("`Volume_0?0[0-1]`") from the Google 1000 Book release. Training data was generated using the OCRopus system by oversegmenting the input images, performing forced alignment of each word image with the supplied textual transcriptions. 400000 segmented and aligned characters were batched by volume and the batches used as training data for the proposed method. Our initial results suggest that (1) weighted mixture densities formed as described above can be an effective means of performing distributed training of large classifiers for OCR systems, (2) batch likelihoods can be used for selecting style-adapted classifiers that improve classification on individual volumes, and (3) the modified boosting procedure described above can be an effective alternative to standard AdaBoost.

4 Discussion

The simultaneous estimation of $\tilde{P}(x)$ and $\tilde{P}(c|x)$ using separate models is a simple and straightforward engineering solution to the problem of building large, style-adaptive classifiers. Our initial experiments suggest that the approach is effective for OCR tasks. We are currently re-implementing the same methods in C++ for actual deployment in OCRopus.

Given the multitude of potential applications (including rejecting bad data), a simple model of the sample distribution should probably be computed and supplied with any discriminative model.

In order to stimulate research in distributed training, style adaptation, and large training sets, and to compare the proposed simple method with more sophisticated methods in the machine learning literature, we are planning a competition at the ICPR 2008 conference and a release of 60 million aligned and segmented training samples from the Google 1000 books.

References

[1] K. Chellapilla, M. Shilman, and P. Simard. Combining multiple classifiers for faster optical character recognition. In *DAS*, 2006.

[2] Cheng-Tao Chu, Sang Kyun Kim, Yi-An Lin, Yuan Yuan Yu, Gary Bradski, Andrew Ng, and Kunle Olukotun. Map reduce for machine learning on multicore. In *NIPS*, 2006.

[3] P. Sarkar and G. Nagy. Classification of style-constrained pattern fields. In *ICPR*, 2000.

preference: oral/poster
topic: distributed training, style adaptation, many classes, large training sets