Semi-supervised Learning for Handwriting Recognition

Gregory R. Ball and Sargur N. Srihari
Center of Excellence for Document Analysis and Recognition
University at Buffalo
UB Commons
520 Lee Entrance, Suite 202
Amherst, NY 14228
srihari@cedar.buffalo.edu

Abstract

We present a framework of adaptive (self-training) semi-supervised learning as applied to the problem of handwriting recognition. Each problem instance itself is treated as a set of unlabeled "training" data; a general model, trained on a set of labeled data, is adapted into an appropriate problem specific model. Learning is continued until convergence is reached, yielding better results than the generalized model alone. An implementation of the framework was tested on English and Arabic handwritten documents. The initial supervised learning model gave word recognition performance of 81% and 67% for English and Arabic respectively. The subsequent semi-supervised learning adjustments yielded 86% and 77% word recognition performance for English and Arabic respectively.

1 Introduction

In developing solutions using machine learning approaches, often a mixture of labeled and unlabeled data is encountered. Semi-supervised learning attempts to take advantage of this state by using the available labeled data as known examples of mappings while still looking to the unlabeled data to learn even more. Generally, this technique aims to exploit the general nature of large quantities of training data while avoiding the considerable expense in labeling. Other problems, such as handwriting, can lead to similar situations for different reasons. Although large quantities of labeled training data can be expensive, such sets slowly have become available in many languages. However, incorporating the diversity of writing styles into a single model leads to over-generalization.

When algorithms are presented with samples of writing by a single writer to analyze (for example, for recognition), the model is not as efficient in terms of accuracy as a model trained specifically to that writer's style. If training is not performed to the writer's style, the performance will not be ideal. Such training can take the form of manual training, but as in the supervised learning case this can be expensive in terms of actual cost or convenience. A more cost effective solution can be found by approaching the problem in a semi-supervised manner. A generalized writing model is generated by utilizing one of the large datasets consisting of manually truthed documents in advance, and the writer's runtime sample is treated as a unlabeled set for training. Since the learning takes place concurrently with the ultimate desired task (e.g. recognition), modifications to the standard approaches need be made.

One simple modification is the technique of self-training or self-learning, which uses an algorithm to repeatedly execute a supervised method on unknown data. Each execution on the unknown set, part of the set is labeled. A supervised algorithm can integrate the newly labeled information into the overall model, and continue with execution. This idea has been around for a significant period of time [6, 3], remaining popular today, used in such areas as natural language processing, parsing, and machine translation [22]. Other more sophisticated techniques have been put forth as well.

Most relevantly, semi-supervised learning in the sense of self-training has been used in natural language processing. For example, Yarowsky [21] applied the technique to word sense disambiguation, and Riloff [15] applied it to distinguish subjective sentences from objective sentences. Maeireizo [10] investigated co-training in training classifiers that predict emotions in spoken dialogues.

Writer adaptation, specializing from a writer-independent handwriting recognition algorithm to a writer-specific one, has successfully been applied to cursive writing in Latin script both in the context of on-line recognition[19, 5, 4, 20] and off-line

978-0-7695-3725-2/09 $25.00 © 2009 IEEE
DOI 10.1109/ICDAR.2009.249

recognition[20, 12, 13, 11]. On-line handwriting recognition can use writer adaptation to create personalized systems by using supervised learning. Hand held devices, for example, can go through a training process to better recognize a writer's handwriting (enrollment adaptation [13]). As Subrahmonia et al mentioned in [19], "for the users who will make extended use of such a system the gain in productivity due to increased accuracy will offset the initial inconvenience of training." They bootstrap from a writer independent model, adapting the character models with writer input to create a writer dependent model. They found that a few hundred words of training data was able to reduce the error rate significantly. Senior and Nathan [16] were able to use a much smaller set of training words, as few as five, in order to reduce the error rate. They used the Maximum Likelihood Linear Regression technique to adapt the means and variances used in their HMM. Connell and Jain[5, 4] identified character styles (lexemes) of individual writers, and specialized the lexeme model to the writer's training data in order to deal with limited training data. More recently, Haluptzok et al [7] discussed several methods for an on-line system to adjust its recognition system to a specific user's writing involving such ideas as enrollment or incremental adjustment.

Off-line handwriting recognition models can also adapt their independent models with relatively few words. Vinciarelli and Bengio [20] noted that they were able to adapt a writer independent system with 30 words. They attempted to train their system solely on the lone writers' words, and were not able to meet the performance of the adapted system until training with many more words, their lower bound estimate being at least 200. Nosary et al [12, 13, 11] used the recognition output from their system as training data, using batch adaptation as recognition progressed. In batch adaptation, system recognition output is stored and used at a later stage, as opposed to incremental adaptation where each new recognition result is used to improve the system.

In our version of semi-supervised learning for recognition, we use a writer independent system to select samples of the input document for use as unlabeled training data. We iterate, combining the manually labeled with the newly identified unlabeled data and then recomputing the prototype vectors. By combining a traditional hidden Markov model (HMM) recognition engine with a writer adaptation model, we have achieved a significant reduction in our error rate. A key question for any writer adaptation method is how exactly to integrate an author's sampled writing into a writer-dependent model. We explored several different methods for integrating their learned handwriting samples. To evaluate performance for English, we tested on the CEDAR letter database [2]; for Arabic, we tested on the same Arabic dataset we used earlier [1].

2 Model incorporation of unique and internally consistent samples

Whichever technique is used, the key to this model is to treat the runtime, internally consistent sample as an "unlabeled set" for the purpose of semi-supervised learning. Internal consistency is critical for the approach to achieve success. Such consistency can in fact be enforced by data analysis during model adjustment. For example, a single author can be expected to produce characters somewhat consistently. That is, while exceptions may exist for certain character combinations or other minor reasons, the majority of instances of characters of a certain class produced by a writer generally fall in some distribution, a Gaussian for example. If data analysis shows that the characters of a specific class are in fact inconsistent, or that a single sample is an outlier of a writer's production, the learned samples can be discounted or considered to have diminished utility, reflected by reduced weight by the model adjustment algorithm.

One main advantage of this technique is the ability to utilize unusually produced characters that are still internally consistent. This mimics the intuitive process of a human reading a production of an unfamiliar writer who writes some characters uniquely or in a non-standard way. As the reader is exposed to more writing by the same author, the unique nature of writing is internalized and can actually be used to help the reader process additional writing by the same author. Generalized models fail to capture this idea since characters written in a non-standard or unique way will not be weighted heavily when creating the model. An example of this idea is found in Figure 1. Figure 1(a) shows a set of instances of the English character "r" written by several different writers. There is little internal consistency, although the samples are in general somewhat representative of how the majority of the English writing community produces the character. Figure 1(b) consists of instances of the English character "r" written by a single author. In this case, the characters are written in an unusual way, but are internally consistent. A generalized recognition model will be much less likely to correctly recognize this character when written by this writer than a model tuned specifically to the writer's unique style. In traditional supervised approaches, this cannot happen without a expensive enrollment step, avoided by considering the actual sample as a unlabeled training.

3 Supervised Learning

We approach the basic task of recognition with an HMM utilizing a lexicon with a very similar approach as in our earlier work [1]. Each word to be recognized is segmented

Figure 1. (a) set of character "r" by multiple writers (b) sampling of character "r" writing by a single writer

based on ligatures and whitespace separation occurring inside the word [9].

A set of approximately two thousand manually cut Arabic characters (about 10 per class) were used to initialize the character prototypes. The IFN/ENIT[14] database was then used to train the system. Training was accomplished by executing the described recognition on each word in three of the IFN/ENIT sets with a single word lexicon consisting of the ground truth. The representative prototypes were combined and used as our writer independent model.

For both English and Arabic, we use the WMR (word model recognition) feature set, which consists of 74 features. Two are global features (aspect and stroke ratio) of the entire character. The remaining 72 are local features. Each character image is divided into 9 subimages. The distribution of the 8 directional slopes for each subimage form this set (8 directional slopes × 9 subimages = 72 features). $F_{l_{i,j}} = s_{i,j}/N_i S_j, i = 1, 2, ..., 9, j = 0, 1, ..., 7$, where $s_{i,j}$ = number of components with slope j from subimage i, N_i = number of components from subimage i, and $S_j = max(s_{i,j}/N_i)$.

4 Unsupervised Learning

The character models are modified by identifying characters in the process of word recognition. Our algorithm makes use of the words the system is most confident of being correct. Since it is an off line algorithm, we cannot ask the user for additional input, but can proceed by treating the sequence of words with highest confidence as a sequence of truthed handwriting samples. The document consists of a set of words W. We create an empty set W' representing words already used for adaptation. The algorithm first recognizes the entire document, assigning the top-1 choice to each word in the document. The related confidence score for each word (based on the distance from the closest prototype centroid) is also stored. The confidence metric used is the average Euclidean distance of each supersegment (best identified combination of segmentation segments) from its corresponding prototype.

After processing the entire document, the confidence scores are ranked. The words assigned a better confidence score are more likely to be recognized correctly. Since the word with the best confidence score is most likely to be correct, it is added to W'. The word's most likely segmentation is created with respect to the recognized sequence of characters by the recognizer. The features are extracted from the characters, and added to the character prototype sets. The process is repeated, taking the word $w \in W$ with the best score such that $w \notin W'$ at each step. As a result of adding the feature set from the component characters, the original author's individual style is incorporated into the prototypes.

When recognized incorrectly, very short words are somewhat likely to still have high confidence scores. For example, a two character word can sometimes be poorly segmented and have the pieces somewhat resemble other characters. This is less likely to happen with long words since very different words are unlikely to have such ambiguous segmentation points. Words which differ by only a character might be recognized incorrectly, but still contribute mostly correct character prototype information due to the other characters which were recognized correctly. As a result, poorly recognized short words can have a detrimental effect on the the adaptation process. When adapting to an incorrectly recognized word, incorrect prototypes are introduced to the global prototype set. This can have the effect of a worsening recognition rate after their addition.

We experimentally verified (see section 5) that this phenomenon was taking place. To adjust for it, we tried two strategies. Our first strategy was to not adapt to words if the recognized words had fewer than five characters. This resulted in adapting to fewer incorrect samples being used in adaptation. However, since there were fewer samples available for training, this meant that the adaptation had to stop at an earlier point than using all words in the document. Our second strategy was to adapt to longer words first, and adapt to shorter words when the initial adaptation stage had stabilized resulting in fewer incorrect samples being used at the start while still using all available recognized words.

The writer-independent model's prototypes were initialized by clustering the prototype character features using K-means into a set number of clusters. The number of initial clusters as well as how the new prototype features are added affects performance. Another major factor in the performance of adaptation is the method by which the features from the newly learned prototype characters are combined with the original writer-independent prototypes. There are three broad strategies of combining the newly learned characters with the writer-independent models–adding the new feature vectors to the existing cluster centroids, combining the new feature vectors in some way with the writer-independent vectors, or replacing the writer-independent vectors with the writer-dependent features. After each phase, we reinitialize and recompute the prototype models based on the additional samples (the manually labeled along with the newly identified unlabeled data).

5 Experiments

Two sets of experments were conducted, evaluating both Arabic and English performance.

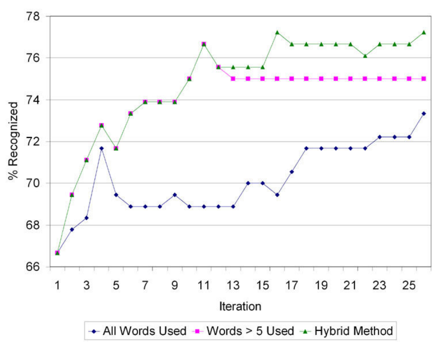

Figure 2. Three methods processing a representative Arabic document

(i) For Arabic, the adaptation algorithm was tested on a database of ten documents, each one or two pages long, written by ten different authors[18, 17]. When creating the sample set, the writers were told to make no effort to write neatly. Since no part of any the documents were used in the training phase, all of the approximately 100 documents were available for the testing phase. Although this dataset is somewhat modest in size, there is a general lack of datasets available consisting of pages of Arabic text. After an initial recognition pass, the word recognition rate was about 67%. After the (unrestricted) adaptation was finished, the recognition rate had improved to 73.3%, about a 6.3% improvement with 170 words used for adaptation, when the maximal correct score had been reached. About 35 words were used for adaptation incorrectly. When using only words with lengths greater than or equal to five characters for adaptation, adaptation stabilized after adapting to about 60 words, which resulted in a document recognition rate of 75%. The word recognition rate for the set of words greater than or equal to five characters itself was initially 64.4% and grew to 76.3%. Only about 14 words were adapted to incorrectly. After adapting to as few as about 20 words, with only about two incorrect words used for adaptation, documentwide word recognition had already grown to about 73%. When using the hybrid approach, the maximum word recognition rate was achieved after adapting to 130 words, about 22 of them incorrectly. The word recognition rate was 77.2%, greater than either of the other two approaches. A representative document was processed and the results of the various methods shown in Figure 3.

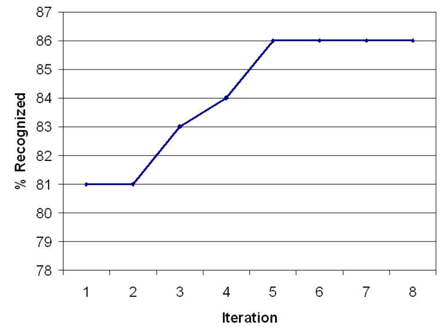

Figure 3. Overall results on an English document

(ii) For English, the adaptation algorithm was tested on a subset of the CEDAR letter dataset, truthed by the transcript mapping described in [8]. Increase in English performance is shown in Figure 5. One interesting result was that the magnitude of improvement seen over Arabic documents was larger than the English documents, likely due to the larger number of characters in Arabic. In either case, the interesting feature is not the raw performance, but the magnitude improvement seen with the prototype integration.

We ran all experiments for a varying number of starting prototypes ranging from one to 20 centroids generated by the K-means algorithm. The performance peaked at about five starting prototypes, with significantly less performance increase observed as the number of prototypes continued to increase. The rapid increase in performance was found while increasing from one to five prototypes is likely a reflection of the incorporation of multiple common styles for writing each character. The slow decrease in performance increase based on adaptation as the number of prototypes continued to increase is likely due to the less weight the newly learned information holds. In other words, when there are 20 prototypes available for a given letter, reflecting many different variations of styles for a given letter, the model is so broad that adding yet another prototype carries little weight. Utilizing alternate strategies for incorporating the features yielded less of an effect on performance, especially when an equivalent number of writer-dependent prototypes were present in the models (i.e., when the replacement method had five learned prototypes vs. the addition method had five learned prototypes).

6 Conclusions and Future Directions

By approaching the task of handwriting recognition with a semi-supervised approach, higher performance is achieved than a standard supervised approach. Such a method avoids the expensive step of training the system through enrollment, especially in the context of offline recognition systems. The significant improvement is realized when we have even only several words written by the same author, for example in the case where a short block of text was written. We have shown that adaptation to the longer words first following by adaptation to the shorter words achieves better results than adapting to the entire set of words in a document or to only the longer words.

Semi-supervised learning often views the problem of labeling a large quantity of training data as expensive. While handwriting recognition systems today are often trained on such tediously labeled sets of data, their input documents themselves are unlabeled by their very nature. The entire purpose of some handwriting tasks (such as recognition) is dependent upon the system's ability to label the data. By treating this set of data as the unlabeled set in a semi-supervised learning approach concurrently with the task itself, performance benefits are realized.

References

[1] G. R. Ball and S. N. Srihari. Writer adaptation in offline arabic handwriting. In *Proc. Document Recognition and Retrieval XV, IST/SPIE Annual Symposium*, San Jose, CA, January 2008.

[2] S.-H. Cha and S. N. Srihari. Writer identification: Statistical analysis and dichotomizer. In *Proceedings of SS&SPR 2000 WVCS - Advances in Pattem Recognition*, volume 1876, pages 123–132, 2000.

[3] O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.

[4] S. Connell and A. Jain. Writer adaptation for online handwriting recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24:329–346, March 2002.

[5] S. D. Connell and A. K. Jain. Writer adaptation of online handwriting models. In *ICDAR '99: Proceedings of the Fifth International Conference on Document Analysis and Recognition*, pages 434–437, Washington, DC, USA, 1999. IEEE Computer Society.

[6] S. Fralick. Learning to recognize patterns without a teacher. *IEEE Transactions on Information Theory*, 13(1):57–64, Jan 1967.

[7] P. Haluptzok, M. Revow, and A. Abdulkader. Personalization of an online handwriting recognition system. In *Tenth International Workshop on Frontiers in Handwriting Recognition*, 2006.

[8] C. Huang and S. N. Srihari. Mapping transcripts to handwritten text. In *Proc. Int. Wkshop Frontiers in Handwriting Recognition(IWFHR-11)*, pages 15–20, La Baule, France, October 2006.

[9] G. Kim. *Handwritten Word and Phrase Recognition*. PhD thesis, University at Buffalo, State University of New York, 1996.

[10] B. Maeireizo, D. Litman, and R. Hwa. Co-training for predicting emotions with spoken dialogue data. In *The Companion Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, 2004.

[11] A. Nosary, L. Heutte, and T. Paquet. Unsupervised writer adaptation applied to handwritten text recognition. *Pattern Recognition*, 37(2):385–388, 2004.

[12] A. Nosary, L. Heutte, T. Paquet, and Y. Lecourtier. Defining writer's invariants to adapt the recognition task. In *Document Analysis and Recognition, 1999. ICDAR '99. Proceedings of the Fifth International Conference on*, pages 765–768, 1999.

[13] A. Nosary, T. Paquet, L. Heutte, and A. Bensefia. Handwritten text recognition through writer adaptation. In *IWFHR '02: Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition (IWFHR'02)*, pages 363–368, Washington, DC, USA, 2002. IEEE Computer Society.

[14] M. Pechwitz, S. S. Maddouri, V. Maergner, N. Ellouze, and H. Amiri. IFN/ENIT - database of handwritten Arabic words. In *Proc. of CIFED'02*, pages 129–136, 2002.

[15] E. Riloff, J. Wiebe, and T. Wilson. Learning subjective nouns using extraction pattern bootstrapping. In *Proceedings of the Seventh Conference on Natural Language Learning*, pages 25–32, 2003.

[16] A. Senior and K. Nathan. Writer Adaptation of a HMM Handwriting Recognition System. In *ICASSP '97: Proceedings of the 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '97)-Volume 2*, volume 2, pages 1447–1450, Washington, DC, USA, 1997. IEEE Computer Society.

[17] S. N. Srihari, H. Srinivasan, P. Babu, and C. Bhole. Handwritten Arabic Word Spotting using the CEDARABIC Document Analysis System. In *Proc. Symposium on Document Image Understanding Technology (SDIUT-05)*, pages 123–132, College Park, MD, November 2005.

[18] S. N. Srihari, H. Srinivasan, P. Babu, and C. Bhole. Spotting Words in Handwritten Arabic Documents. In *Proceedings SPIE*, pages 606702–1–606702–12, San Jose, CA, 2006.

[19] J. Subrahmonia, K. Nathan, and M. Perrone. Writer dependent recognition of on-line unconstrained handwriting. In *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, volume 6, pages 3478–3481, 1996.

[20] A. Vinciarelli and S. Bengio. Writer adaptation techniques in HMM based off-line cursive script recognition. *Pattern Recogn. Lett.*, 23(8):905–916, 2002.

[21] D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, 1995.

[22] X. Zhu. *Semi-Supervised Learning with Graphs*. PhD thesis, Carnegie Mellon University, 2005.