A Discriminative Semi-Markov Model for Robust Scene Text Recognition

Jerod J. Weinman
Dept. of Computer Science
Grinnell College
weinman@grinnell.edu

Erik Learned-Miller and Allen Hanson
Dept. of Computer Science
University of Massachusetts Amherst
{elm,hanson}@cs.umass.edu

Abstract

We present a semi-Markov model for recognizing scene text that integrates character and word segmentation with recognition. Using wavelet features, it requires only approximate location of the text baseline and font size; no binarization or prior word segmentation is necessary. Our system is aided by a lexicon, yet it also allows non-lexicon words. To facilitate inference with a large lexicon, we use an approximate Viterbi beam search. Our system performs robustly on low-resolution images of signs containing text in fonts atypical of documents.

1 Introduction

Noise, unusual fonts and typesetting, and low resolution are endemic in scene text captured by portable cameras so that few assumptions can be made about the input. This frequently makes scene text recognition more difficult than many document recognition problems.

It is common in printed (document and scene) text recognition to assume characters can be binarized and word boundaries easily found. However, in the handwriting recognition problem, characters are typically connected, so recognition must be combined with character segmentation [6]. Furthermore, spaces between handwritten words are not consistently larger than gaps between characters. Most approaches to handwriting perform word segmentation prior to recognition [7, 5, 3], yet some combine candidate word segmentations to find the best interpretation [9]. This is analogous to speech recognition, where there is little to indicate word boundaries [4]. Both character and word segmentation are as difficult in scene text recognition as they are in speech and handwriting recognition due to the wide variety of formats.

When recognizing a small sample of text, as with signs, correctly segmenting words prior to recognition is nearly impossible. Signs are often less constrained by kerning conventions (Fig. 1). Moreover, using the space between characters to determine word boundaries often assumes an error-free character binarization, which is highly unwarranted when noise or low resolution results in broken and/or touching characters.

Figure 1. With few characters, signs make prior word segmentation difficult. Some (top) have larger inter-character gaps than others' inter-word spaces (bottom).

Much prior work in handwritten and printed text recognition is lexicon driven, but words from outside the lexicon must be allowed to handle input variety. Previous work allowing out-of-lexicon recognition has assumed word and character segmentations, or used back-off models, such as character bigrams [15]. While some scene-text readers use lexicons in post-processing [1], integrating a lexicon with recognition generally yields better results [14].

Our goal is to combine the strengths of these approaches. Because the two-step approach is prone to error, we avoid committing to segmentations before performing recognition. In this paper, we present a robust probabilistic model that is aided by a lexicon to help bias the recognition toward known words. However, since many words in the environment are not dictionary words, our system can fall back on a simpler n-gram language model so that text may also be recognized as non-lexicon words. We combine not only character recognition and segmentation, but also word recognition and segmentation. Our system provides greater flexibility by treating an inter-word space as yet another character to recognize, allowing the recognition process

978-1-4244-2175-6/08/$25.00 ©2008 IEEE

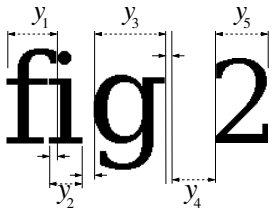


Figure 2. A segmentation of a string of text into 5 primary regions (indicated by the dashed lines) that can be scored for various labelings.

to be guided by *either* an n-gram model or a lexicon. This avoids constraining the system by committing to a segmentation too early. Experimental results on scene text images are given, comparing both open (no lexicon) and closed-vocabulary (lexicon words only) modes with our proposed hybrid mode against commercial OCR software.

2 Recognition Model

We use a discriminative semi-Markov field to perform joint segmentation and recognition [10]. It captures both the dependencies between labels in a sequence and the duration of a particular state along the sequence. By modeling state duration (i.e., the width of a region corresponding to some character), the semi-Markov model is richer than the typical Markov model. Specifically, the probability of a segmentation boundary depends on the width of the segment thus far.

Unlike generative hidden Markov and semi-Markov models, the discriminative semi-Markov model is directly trained to optimize the posterior distribution used for prediction, and it can use richer contextual features of the input image without violating the independence assumptions required of an HMM or introducing the additional model complexity required to handle them appropriately.

Dynamic programming is used to find the most likely character sequence. Here we briefly discuss how a segmentation and labeling is scored. Section 3 describes how to find the optimal segmentation and labeling.

A segmentation induces a set of unknowns $\mathbf{y}$ and a corresponding set of discriminant functions $\{U_C\}_{C\in\mathcal{C}}$. Each unknown $y_i \in \mathcal{Y}$ takes on a label in $\mathcal{Y} \equiv [\texttt{A}-\texttt{Za}-\texttt{z0}-\texttt{9}\sqcup]$ (where $\sqcup$ is an inter-word space). The example in Figure 2 shows one segmentation, or parse, of a text string. In this parse, there are five regions corresponding to character hypotheses that must be given labels. Notice in particular that one of the parse regions, y_4, corresponds to the "space" character. Modeling spaces explicitly as a character allows us to seamlessly integrate word segmentation with character recognition and segmentation. Since different segmentations must compete with each other, two properties of the parse itself are also scored: overlaps of character spans and gaps between them.

Given a segmentation and the set of induced unknowns, we may define the conditional probability

$$p\left(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta}\right) \propto \exp\left\{\sum_{C\in\mathcal{C}} U_C\left(\mathbf{y}_C, \mathbf{x}; \boldsymbol{\theta}\right)\right\}, \quad (1)$$

where $\mathbf{x}$ represents the observed image, and $\mathcal{C}$ is a set of learned discriminant functions. The exact set of functions summed over will depend on the segmentation being evaluated. As an example, for each unknown, there will be one function corresponding to a character recognition discriminant. The notation $\mathbf{y}_C$ indicates the subset of the unknowns that are arguments to U_C. We use five classes of U_C in calculating the score of a parse:

Appearance Each span is scored by a linear discriminant, $U^A_{r,t}\left(y, \mathbf{x}; \boldsymbol{\theta}^A\right) = \boldsymbol{\theta}^A_{r,t}\left(y\right) \cdot F_{r,t}\left(\mathbf{x}\right)$, for a character y and the span width from index r to t, learning not only appearance, but that "w"s are wide and "i"s narrow.

Bigrams Each pair of neighboring character spans is given a bigram score, $U^B\left(y', y\right)$.

Lexicon Parameter U^W can replace U^B to promote character sequences that compose a lexicon word.

Overlap If neighboring characters overlap, a simple term is added, $U^O_{n,r}$, depending on how many pixels overlap, as indicated by n and r.

Gap A gap between characters from n to r is scored by a learned, linear discriminant $U^G_{n,r}\left(\mathbf{x}; \boldsymbol{\theta}^G\right) = \boldsymbol{\theta}^G_{n,r} \cdot F_{n,r}\left(\mathbf{x}\right)$ using the image features.

The model parameters are learned from data. We use decoupled piecewise training to learn the discriminant functions individually [12]. Finding the MAP estimate for the parameters involves solving a convex optimization problem.

3 Model Inference

Recognition in our model means finding the segmentation and corresponding labeling that maximizes a total (summed) score. The constituents of this score—the exponent of Eq. (1)—were described in the previous section. Here we describe how to find the best score.

A two-dimensional dynamic programming table is constructed to find the best parse. If word boundaries were assumed so that a text region contained a single word, we could simply constrain the dynamic program to find parses corresponding to lexicon words, as done by Jacobs et al. [5]. Rather than assume a prior word segmentation, we recognize word boundaries by treating a space as a character and then determine whether what precedes the space is a lexicon word.

Let $S(t, y)$ be the optimal score for a span ending at index t with character y. When $y \neq \sqcup$, the table is built by adding a new span and labeling to the previous best parse via the recurrence

$$S(t, y) = \max_{n,r,y'} S(n, y') + P(n, r, t, y', y, 0), \quad (2)$$

where $P(n, r, t, y', y, 0)$ represents the parse score for adding a segment that starts at r, ends at t, and is labeled y, while the previous character y' ended at n. The last argument to P, $w = 0$, indicates that the additional character is not part of a lexicon word. The base case is $S(0, y) = 0$, with $S(t, y) = -\infty$ for $t < 0$. The parse score is

$$\begin{aligned} P(n, r, t, y', y, w) &= U^O_{n,r} + U^G_{n,r} + U^A_{r,t}(y) W^A_{r,t} \\ &\quad + \left[(1 - w) U^B(y', y) + w U^L\right] W^B_{m,t}, \end{aligned} \quad (3)$$

where m is the *beginning* of the previous span. The total score for a segmentation and labeling is the sum of all the P terms—the exponent of the Markov model (1).

To avoid biasing the total score for parses with more characters, we add weights that assign the appearance and bigram scores to every index of their spans: $W^A_{r,t}$ is the width of the character span, and $W^B_{m,t}$ that of the bigram span. Gaps do not require this because each index of a gap is scored individually. To simplify calculation, we score only segments from r to t that correspond to a set of possible quantized character widths.

Another table $W(t)$ corresponds to the best score for a lexicon word ending at t. When the new $y = \sqcup$, it signals the end of a character string, which may be a lexicon word or not. The recurrence must determine whether the optimal parse is to accept the previous best lexicon word S_ℓ or the non-lexicon parse $S_{\bar{\ell}}$ ending before the space:

$$S(t, \sqcup) = \max\left\{S_\ell(t), S_{\bar{\ell}}(t)\right\}, \quad (4)$$

$$S_\ell(t) = \max_{n,r} W(n) + P(n, r, t, \widehat{y}_n, \sqcup, 0) \quad (5)$$

$$S_{\bar{\ell}}(t) = \max_{n,r,y'} S(n, y') + P(n, r, t, y', \sqcup, 0) \quad (6)$$

where $\widehat{y}_n$ is the last character of the word ending at n.

The lexicon-based table W is similar to S, but with an extra layer of complexity:

$$W(t) = \max_n S(n, \sqcup) + B(n, t). \quad (7)$$

This builds upon previous scores, but adds a term $B(n, t)$ that is the optimal score for any lexicon word ending at t, with a preceding space ending at n. Unlike S, the character sequences of the lexicon word score B are constrained to be from lexicon words. Thus, for the kth lexicon word, we can define a score for a parse up to the ith character, y^k_i, that ends at location t, with the beginning of the word preceded by a space that ends at n:

$$\begin{aligned} C(n, t, i, k) = \max_{m,r} \{&C(n, m, i - 1, k) \\ &+ P\left(m, r, t, y^k_{i-1}, y^k_i, 1\right)\}. \end{aligned} \quad (8)$$

As before, the term P is a score for a parse of a particular character including the appearance model and gap/overlap scores. However, the language model is altered since the character is now part of a (hypothesized) lexicon word. The score for adjacent characters that constitute a lexicon word thus replaces the bigram score entirely, as indicated by the argument $w = 1$. Now, the $B(n, t)$ term of Eq. (7) is the highest-valued, complete parse of all lexicon words over the span from n to t,

$$B(n, t) = \max_k C\left(n, t, \left|\mathbf{y}^k\right|, k\right), \quad (9)$$

where $i = \left|\mathrm{y}^k\right|$ is the length of the kth word.

Although the complexity is linear in the number of lexicon words and the length of the text to be parsed, we have proposed a method that hypothesizes every word beginning at every location. This is impossibly slow in practice, so approximations must be introduced.

We eliminate words from consideration based on the relative score of all sub-word parses within a given span. The term $C(n, t, i, k)$ represents optimal scores over the span from n to t, where the subword varies with the arguments i and k, most of which have extremely low scores. To more quickly find the optimal *full* word parse $B(n, t)$ calculated from C, we eliminate these unlikely candidates. Such beam search is common in speech recognition [4]. A standard strategy is to sort the scores C for a given span (n, t), keeping the N best sub-words (i, k). We use $N = 10$.

4 Image Features and Pre-Processing

Even- and odd-phased steerable pyramid filter responses [11] are each rectified into positive and negative components. These four feature images (times six orientations) form the feature vector $F(\mathbf{x})$. See Fig. 3. To normalize for brightness and contrast, all the rectified responses within a 32 pixel square window are ℓ_2-normalized, clipped to a threshold (0.2) and renormalized. A simple linear discriminant (see "Appearance" in Section 2) is used for recognition.

Figure 3. An image is transformed by the steerable pyramid and rectified. For one orientation, the responses of even (0) and odd (1) filters for one orientation along with their positive (+) and negative (-) rectified components are shown.

The features are calculated on the original grayscale image over the entire text region. Recognition of a character is done without explicit prior binarization, segmentation or deskewing. Thus, features from neighboring characters may be included in the feature vector. The discriminant used for recognition must be robust to these neigboring distractor features and some perspective distortion.

Classifiers may be used to detect approximate scene text baselines [2, 13]. At each pixel in all of the detected regions, our function U^A outputs a score for a combined character identity and approximate width.

We assume a one-dimensional string of text, but not that it is perfectly horizontal or linear. Because the detections are text regions, we must transform the set of character scores (residing at each detected pixel) into a one-dimensional representation, as follows.

The set of unique columns in each detected region become indices of a 1-D array. For each column, we assign the score for each hypothesis (character, width, space, gap, etc.) to be its maximum over the detected rows in that column. The resulting array has a score for labeling each array segment, or span, as a particular character or an intra-word/inter-character gap.

Within each text region (detection) connected component, we perform a course binarization of the image using Niblack's algorithm [8]; this is not required, but it speeds recognition by limiting the number of segmentations. We assume this is an over-segmentation, so that components may be combined, but not split.

5 Experiments

Our evaluation is conducted on a set of 85 sign images, containing 183 words and 1144 characters in a wide variety of fonts. They are scaled (as they would be by a text detector), to roughly a 12.5px x-height, and a rough baseline is given manually. We compare the results to commercial OmniPage 15 software.

Our model has an "open vocabulary" mode with no lexicon, using Eq. (2) for $y = \sqcup$. Alternatively, a "closed vocabulary" mode forces words to be from the lexicon if $S_{\overline{\ell}}(t) = -\infty$.

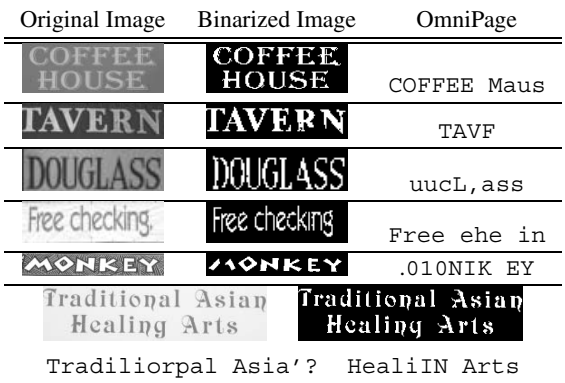


Figure 4. Example inputs and results. These examples are correctly recognized by our model.

The model parameters are learned from data: 934 training fonts for U^A, and a corpus of 82 English books (49 million characters) for U^B. U^O is a truncated quadratic with a scale that is a large fraction of the appearance scores (roughly one-third for the largest overlap of 7px). We allow a two pixel overlap without penalty due to character width quantization. The lexicon bias U^W is twice the median bigram score U^B for all bigrams in our 245,000 word lexicon.

Some of the signs in the evaluation data contain fonts with aspect ratios not present in training. Ideally, such fonts would be in our training data. We could easily accomplish this by simply stretching or shrinking our training fonts horizontally. As an alternative, we hypothesize an unknown aspect ratio by running the recognition/parser at several horizontal scales and keeping whichever has the highest final score. This takes longer, and adding additional training fonts would likely perform better.

A mixed vocabulary mode yields better performance than either open or closed vocabulary modes—a 13% error reduction (Tab. 1). In order to achieve optimal performance, OmniPage requires binarized input (using Niblack's algorithm [8]). When we optimize for the aspect ratio of the unknown font we reduce our error by nearly 15%—a 10% error reduction over OmniPage. This approach to handling narrow and wide characters fixes many errors, but it also introduces some. In the ideal case, if adding such characters to the training data only improved the results and did not make them worse, our error rate would drop by 33%. Our model also robustly handles recognition and word segmentation at lower resolutions than it was trained upon (Fig. 5).

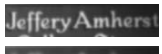
σ	Image	Binarized	OmniPage	Image	Binarized	OmniPage
0.00			`Resumes`			`Je eryAmherst`
0.66			`Resumes`			`JefferyAmherst`
1.33			`WM/ I WS`			`JONryAmhent`
2.66			`11.-4her`			`Jo 116 fp Ak 11_Se-`

Figure 5. Example recognition comparison at lower resolutions. The input images are low-pass filtered by a Gaussian of scale σ px. Our model misreads only the last line.

Table 1. Recognition comparison.

Method	Char. Error (%)
OmniPage, Grayscale Input	23.51
OmniPage, Binarized Input	16.61
Semi-Markov, Open Vocab.	20.28
Semi-Markov, Closed Vocab.	20.45
Semi-Markov, Mixed Vocab.	17.66
Semi-Markov, Multiscale	15.04
Semi-Markov, "Ideal"	11.89

6 Conclusion

We have presented a model that can correctly and flexibly recognize scene text under a variety of conditions including unusual fonts, low resolution, and non-lexicon words. It performs better than commercial document recognition software, which has been used in previous scene text readers. Integrating a lexicon improves performance while also allowing non-lexicon words when warranted by the data. Unlike earlier work [14], we need not perform prior word segmentation, because word recognition is integrated with segmentation, just like character segmentation and recognition. Because no binarization is required, we can recognize characters at lower resolution. Our robust probabilistic model can be learned from data with no hand-tuning of parameters.

Acknowledgments

This work was supported in part by NIH grant 1R21EY018398-01 and NSF grants IIS-0100851, IIS-0326249 and IIS-0546666.

References

[1] R. Beaufort and C. Mancas-Thillou. A weighted finite-state framework for correcting errors in natural scene OCR. *Proc. ICDAR*, 2:889–893, 2007.

[2] X. Chen and A. L. Yuille. Detecting and reading text in natural scenes. In *Proc. CVPR*, pages 366–373, 2004.

[3] C. Huang and S. N. Srihari. Word segmentation of off-line handwritten documents. In *Proc. Document Recognition and Retrieval*, 2008.

[4] X. Huang, A. Acero, and H.-W. Hon. *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. Prentice Hall PTR, 2001.

[5] C. Jacobs, P. Y. Simard, P. Viola, and J. Rinker. Text recognition of low-resolution document images. In *Proc. ICDAR*, pages 695–699, 2005.

[6] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.

[7] R. Manmatha and J. L. Rothfeder. A scale space approach for automatically segmenting words from historical handwritten documents. *IEEE Trans. PAMI*, 27(8):1212–1225, 2005.

[8] W. Niblack. *An Introduction to Digital Image Processing*. Prentice-Hall, Englewood-Cliffs, NJ, 1986.

[9] J. Park, V. Govindaraju, and S. N. Srihari. Efficient word segmentation driven by unconstrained handwritten phrase recognition. In *Proc. ICDAR*, pages 605–608, 1999.

[10] S. Sarawagi and W. W. Cohen. Semi-Markov conditional random fields for information extraction. In *NIPS*, pages 1185–1192.

[11] E. P. Simoncelli and W. T. Freeman. The steerable pyramid: A flexible architecture for multi-scale derivative computation. In *Proc. ICIP*, pages 444–447, 1995.

[12] C. Sutton and A. McCallum. Piecewise training of undirected models. In *Proc. UAI*, pages 568–575, 2005.

[13] J. J. Weinman, A. R. Hanson, and E. Learned-Miller. Joint feature selection for object detection and recognition. Tech. Report UM-CS-2006-054, University of Massachusetts Amherst, Oct. 2006.

[14] J. J. Weinman, E. Learned-Miller, and A. Hanson. Fast lexicon-based scene text recognition with sparse belief propagation. In *Proc. ICDAR*, pages 979–983, Sept 2007.

[15] D. Zhang and S.-F. Chang. A Bayesian framework for fusing multiple word knowledge models in videotext recognition. In *Proc. CVPR*, volume 2, pages 528–533, 2003.