DiVA
http://www.diva-portal.org

Postprint

This is the accepted version of a paper presented at *9th International Symposium, ISVC 2013, July 29-31, 2013, Rethymnon, Crete, Greece.*

Citation for the original published paper:

Wahlberg, F., Brun, A. (2013)
Feature Weight Optimization and Pruning in Historical Text Recognition.
In: George Bebis (ed.), *Advances of Visual Computing: 9th International Symposium, ISVC 2013, Rethymnon, Crete, Greece, July 29-31, 2013. Proceedings, Part II* (pp. 98-107). Springer Berlin/ Heidelberg
Lecture Notes in Computer Science
http://dx.doi.org/10.1007/978-3-642-41939-3_10

N.B. When citing this work, cite the original published paper.

Permanent link to this version:
http://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-212536

Feature weight optimization and pruning in historical text recognition

Fredrik Wahlberg, Anders Brun

Centre for Image Analysis
Uppsala University
Sweden

Abstract. In handwritten text recognition, "sliding window" feature extraction represent the visual information contained in written text as feature vector sequences. In this paper, we explore the parameter space of feature weights in search for optimal weights and feature selection using the coordinate descent method. We report a gain of about 5% AUC performance. We use a public dataset for evaluation and also discuss the effects and limitations of "word pruning," a technique in word spotting that is commonly used to boost performance and save computational time.

1 Introduction

In off-line recognition of a historical text, the starting point is are images of some manuscript or other material that can host written characters. With historical documents, problems like degraded ink, rough handling over generations or geometrically distorted parchment due to moisture needs to be taken into account. The problem of performing a full recognition (i.e. computerized transcription) is so far unsolved. To still be able to search and index the treasures in our libraries today, searching for user selected templates, called word-spotting, has been shown to be useful.

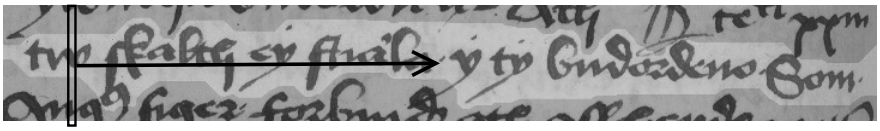

Fig. 1. An automatically segmented line of handwriting in old Swedish from the 16:th century. A sliding window is passed over the text line and feature vectors are extracted along its path.

Written text is linear i.e. each letter should be read before the next (with a few exceptions ranging two or three letters). By treating all text as if it were on the same line, the problem of parsing visual information can be reduced into

a sequence matching problem. Extracting feature vectors along a centre path of each text line is called "sliding window" feature extraction, an illustration is shown in figure 1. In this paper, we will examine the importance of each single feature, potential redundancy and benefits from weighing features differently. We also present a method for finding weight combinations.

In contrast to earlier papers on the same theme [1–3], we have not used pruning (i.e. heuristic exclusion rules based on simple geometric features) to exclude potential word matches. We argue that pruning gives rise to an unnecessary limit on the recognition rate. Different recognition techniques are more or less robust to noise. We have chosen a training free feature matching approach to not "obscure" the performance on the raw data with effects of robustness techniques.

1.1 Previous work

"Sliding window" feature extraction, pioneered by [4] and commonly used in text recognition[5, 6], catches some information relevant to text recognition, at each pixel column. The quantified information is then concatenated into a feature vector. In [7], the most commonly used "sliding window" feature vector was proposed. The authors also proposed a preprocessing scheme for the word image before feature extraction to increase robustness.

In [1], the concept of dynamic time warping (DTW) for word matching was developed, with an application focus. The goal was to be able to create an ordered list of matches between template word images and some collection of word images. DTW finds the optimal flexible best match of two "sliding window" feature vector sequences, returning a dissimilarly cost.

In [3], several computationally fast pruning rules were introduced, later developed further by [8] and used by [2, 1] . By using these rules, a large portion of potential matches can be removed before executing DTW. However, false negatives are introduced, which we argue creates an unnecessary limitation on the word matching.

In [2], some commonly used features, including most from [7] were evaluated. Average precision scores for some features were given and discussed. Collections of features were only analyzed in passing. We have extended this analysis to collections of features together with using optimization to find improved weighting of specific features.

2 Method

Each element in the feature vector, given by the "sliding window" feature extraction scheme, is created from several local features. To examine the effect of weighting the influence of each feature, we have built an evaluation environment reproducing what is described in [1, 2]. Their implementation consisted of a two part word matching pipeline, pruning and DTW, described below. We have used a standard evaluation set and focused on the word spotting problem after text lines and words have been segmented (part of a page can be seen in figure 2).

Below, we describe the effects of pruning and methods for finding feature weights in this context.

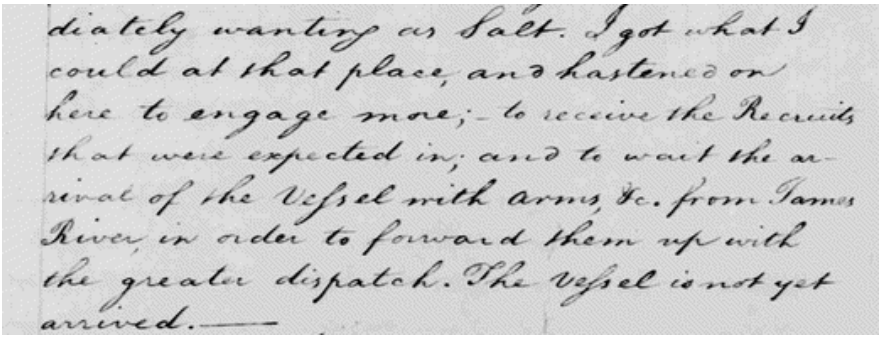

Fig. 2. A part of a page from the Washington letters.

2.1 Sliding window feature extraction

After segmentation and normalization (as in [7]) of the text lines, relevant information at each pixel column was reduced to a vector in a feature space. Each element in a feature vector captured some, to handwriting, important characteristic of the text line around the associated column. Several commonly used features exist in the literature (e.g. [7, 2, 4, 5]), the ones we have used in our experiments are described below.

1. **Projection profile** The vertical projection of the pixel column i.e. the sum of foreground pixels in one column.
2. **Upper contour** The position of the foreground pixel with the highest position i.e. lowest y coordinate value.
3. **Lower contour** The position of the foreground pixel with the lowest position i.e. highest y coordinate value.
4. **Upper projection** The projection profile above the upper baseline. The upper baseline is a vertical line through the image that is placed directly above the upper part of most lower case letters and crosses through ascenders. This roughly gives a projection profile of only the ascenders and upper case letters.
5. **Lower projection** The projection profile below the lower baseline. The lower baseline is a vertical line through the image that is placed directly below the lower part of most letters and crosses through descenders. This roughly gives a projection profile of only the descenders.
6. **Centre of Weight** The mean of all y coordinate values of the foreground pixels in a column.

7. **Foreground/background transitions** The number of transitions between the foreground and the background while following the pixels in a column from lowest to highest y coordinate value.
8. **Second moment** The variance of all y coordinate values of the foreground pixels in a column.
9. **Gradient of the upper contour** First derivative of the upper contour, this feature took neighbouring pixel columns into account to form a more reliable gradient.
10. **Gradient of the lower contour** First derivative of the lower contour, this feature took neighbouring pixel columns into account to form a more reliable gradient.
11. **Foreground fraction** The fraction of pixels between the upper and lower contours belonging to the foreground.

Some of the features listed above were considered by [7] to need preprocessing to increase robustness (e.g. in the upper contour, the ascender of a letter like "k" was spread out or seen as a short spike depending on slant). A "normalization" was introduced to the word images, normalizing letter height and removing slant and skew. In the evaluation database, described below, the normalization step has already been performed. All features were normalized to the interval $[0, 1]$.

2.2 Word-spotting

By using "sliding window" feature extraction, word-spotting can be treated as a sequence matching problem. Word images are compared by treating them as sequences of feature vectors. Using DTW for sequence comparison, gives a dissimilarity measure between two word images. This concept originates from speech recognition, hence the wording of "time warping." Three words from our evaluation set, described below, can be seen to the left in figure 3.

An optimal warping for matching sequences of feature vectors can be found by using dynamic programming. The feature vector sequence from the template word, sequence A, is compared to the sequence of some other word, sequence B. A cost matrix W is generated where each feature vector of A is compared to every feature vector of B, as in equation 1, where $d(.)$ is some distance function (square Euclidean distance in our case).

$$W(i,j) = \sum_{k=1}^{N} d(A_i, B_j)^2 \tag{1}$$

The minimal cost path from the upper left corner to the lower right corner of the matrix represent an optimal warping, with respect to the distance function. The allowed steps through this matrix are down (skip one feature vector of A), right (skip one feature vector of B) or down-right (match). By using the Sakoe-Chiba band constraint [9], matching performance can be increased and calculation time lowered. By only allowing a diagonal path through the weight matrix for warping, pathological warpings (where a very small portion of one

Fig. 3. Right: Illustration of the Sakoe-Chiba constraint on the weight matrix in dynamic time warping. Only the diagonal part of the weight matrix is considered valid for warping (diagonal ±15 elements). The lowest cost path (white line) through the diagonal band represent the warping. Left: Three segmented, binarized and normalized words from the Washington database, used in our evaluation. Note the unnatural slant on the "f" in "from." This is because letters are normalized according to the dominant slant.

word is matched to large parts of another) are avoided. In [1], the allowed warping (i.e. the number of elements around the diagonal) was set to 15. And example, with an allowed warping of ±15 and with the lowest cost path marked, is shown to the right in figure 3.

Pruning In the word spotting pipeline, many potential matches can be removed by pruning. We have examined the rules used in [1, 2] , recommended by [8]. The recommended pruning rules were limits on the area and aspect ratio of the word bounding boxes. One such rule is shown in equation 2, the parameter β was set by experimentation. Also, a rule only allowing matches between words with equal number of descenders was recommended. The number of descenders was defined as the number of connected components below the lower baseline.

$$\frac{1}{\beta} \leq \frac{template_{bbxArea}}{image_{bbxArea}} <= \beta \tag{2}$$

The matching pipeline of [2] was dividend into two steps, pruning and DTW. Using the rules described above reduced the need for matching by DTW and thus the computational cost. We have tried to set the pruning parameters to mimic earlier results, where approximately 85% of the word pairs (potential matches) were ruled out. Using such aggressive pruning identify many true negatives but also misidentifies a significant number of false negatives.

In figure 4, the parts of the Receiver Operating Characteristics (ROC) plot (explained in 3.1) affected by the pruning are shown. We also illustrate in the figure the little room left for DTW to improve on the overall result by using random word dissimilarity scores, instead of DTW. Pruning has been shown to be successfully applied to specific word-spotting scenarios but gives little room for feature based matching. It sets an upper limit on the true positive ratio, Area

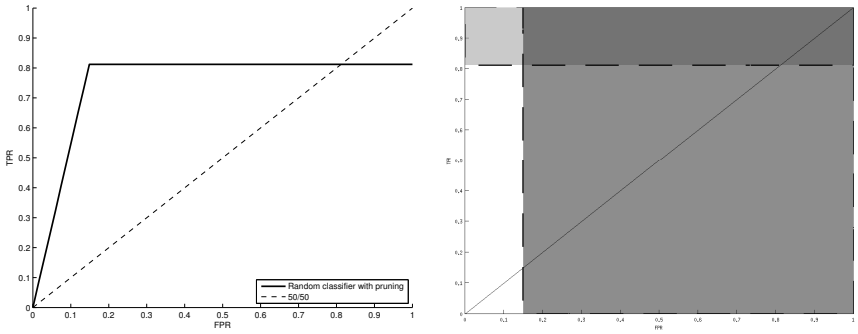

Fig. 4. Left: ROC curve from using pruning and random dissimilarity scores instead of DTW. Right: A ROC plot showning the part affected by DTW, after pruning, in white. The horizonal dashed line is the upper limit of the ROC curve because of false negatives. The vertical dashed line show the dicovered true negatives from the puning and before DTW.

Under the Curve (AUC) (explained in section 3.1) is constrained to the interval $[0.69, 0.82]$. Hence, we will not use it in our evaluation.

2.3 Feature weight parameter space

The 11 features, described above, catches some relevant characteristics of the text (i.e. they contain information relevant to the word matching). At some points many features are strongly correlated, e.g. the upper projection, upper contour and projection at an ascender. If each feature were weighted equally, some word characteristics (e.g. number of ascenders) might dominate the matching. To avoid this in training free matching, manual tuning is required.

Leave one out If some features are redundant, for the current evaluation data, removing it should not give a lower evaluation result than using the full set. By running the word-spotting evaluation one time for each single feature removed, an estimation of which features are the least important can be made. Repeating the process for the remaining features can remove one more etc. This method is not as accurate as an extensive search since not all correlation and noise effects of different feature combinations are taken into account.

Weight optimization The problem of finding the optimal weights, given the data set, can be treated as an optimization problem. The function to minimize is $1-\text{AUC}$, for an 11D vector with weights used in a evaluation. Finding a better weight vector pushes the ROC curve upwards and to the left, which corresponds to earlier true positives.

The discrete nature of evaluating matches between a limited number of words give rise to a piecewise flat parameter space. Commonly used gradient based optimization methods, like gradient descent, then fail to find a direction to move in.

Experiments using the coordinate descent algorithm were successful [10]. When running one iteration, the objective function is minimized one coordinate at a time. The performance increases monotonically, but the algorithm cannot guarantee to find a global optimum. In fact, this kind of block relaxation technique may not even find a local minimum.

In our implementation, each direction was searched using the relatively large step length of 0.1 in a space where each weight can have the interval $[0, 1]$. In cases where the performance did not change with the new weight, we chose the lowest possible weight to create bias towards removing features. We used a training set of 20% (1000 words) of the full database, separate from our evaluation set. The algorithm has empirically been shown to improve the results, even though convergence could not be guaranteed.

3 Experiments

Potential feature redundancy must be seen in context of the remaining features. Removing, or weighting down, one feature does not necessarily mean that it is not a good feature, only that it does not contribute in some collection of features. Also, some features are probably very valuable when separating certain letters but not as useful in most cases. In our evaluation, we could only analyse how important a feature is when comparing whole words.

3.1 Evaluation database

George Washington Papers at the Library of Congress is a collection of letters written by George Washington before, during and after his presidency. The original images (of which a part can be seen in figure 2) are kept by The Uniter States library of congress and from those a public database [11] has been created. All of the material used is from letter book 1, series 2, pages 270-279 & 300-309, written between Aug. 11, 1754 and Dec. 25, 1755 [1]. The database contains 5000 segmented, normalized and labeled words. Three examples of these words are shown in figure 3.

Performance measure Figure 5 and 4 show a Receiver Operating Characteristic (ROC) plot. On the y-axis is the rate of true positive matches (TPR) i.e. the number of correctly identified matches at a specific threshold. On the x-axis, the corresponding rate of false positives (FPR) are given. Along the diagonal, a 50/50 line is drawn to show a comparison to a random classification. The performance measure used in table 1 is the Area Under the Curve (AUC)[12], giving a single number for performance comparison.

[1] http://memory.loc.gov/ammem/gwhtml/gwseries2.html

3.2 Evaluation of single features and collections

In table 1 the results from some evaluations are shown. First, each feature was evaluated separately showing that using only the projection feature was enough to get a significant improvement. Note that this only gives insight into how useful a feature is overall. Second, one feature was taken out and the others weighted uniformly. Third, the lower gradient feature (the least important one from the previous step) was taken out together with each one of the remaining features.

Feature	Single	Leave one out	Leave two out
Projection	**0.877**	0.839	**0.858**
Upper contour	**0.853**	0.840	**0.859**
Lower contour	0.788	0.842	**0.861**
Upper projection	**0.871**	0.840	**0.856**
Lower projection	0.648	0.836	**0.856**
Centre of weight	0.784	0.842	**0.861**
BW transitions	0.802	**0.843**	**0.863**
Second moment	0.792	**0.843**	**0.862**
Upper gradient	0.794	0.841	**0.867**
Lower gradient	0.796	**0.861**	-
Foreground fraction	0.827	0.826	**0.843**

Table 1. Results from performance evaluation of both every single feature and some collections. Performance metric is Area Under the Curve. Bold text indicates that the results was better than using all feature types with uniform weights, giving an AUC of 0.8418.

3.3 Parameter search

In figure 5, the improvement from the weight optimization are shown. In our experiments, a lot of improvement of the AUC score could be achieved by a small number of iterations. We used a subset of the words (20%) for training and a disjunct set for evaluation.

When trying 15 random initial vectors and one with uniform weights (all elements set to 1), the algorithm converged to approximately the same result independent of starting point. This would indicate that the weight parameter space does not contain that many local minima. The AUC score was also improved (> 0.05). We have done preliminary evaluations of the weight vectors on the public data set Saint Gall with encouraging results. This supports the generalizability of the weight vectors trained on the Washington letters.

Some features were in most cases set to zero or a low number. The features most often removed were the contours, centre of weight, second moment, foreground/background transitions and the gradient of the lower contour. This would indicate that these are redundant, assuming our evaluation database. Some examples of vectors in the weight parameter space generated by the optimization algorithm are shown in table 2.

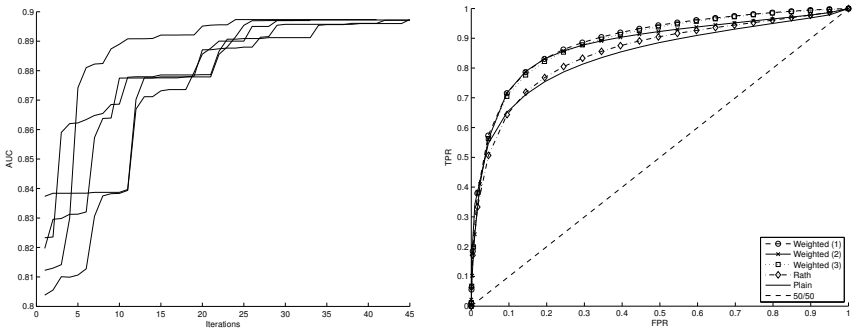

Fig. 5. Left: Curves of AUC over several iterations using random starting vectors. Right: Some results from using weights chosen by the optimizing algorithm. A plain evaluation using uniform weights and the feature subset proposed by [1] are shown as a comparison.

	11 features (ordered as in section 2.1)											AUC
Weights	0.6	0	0	1	1	0.4	0	0	0.2	0	0.3	0.899
	0.9	0	0	0.9	1	0	0	0	0.2	0	0.3	0.900
	0.8	0	0.4	0.9	1	0	0	0	0.2	0	0.3	0.899
Uniform	1	1	1	1	1	1	1	1	1	1	1	0.842
Weights from [1]	1	1	1	0	0	0	1	0	0	0	0	0.851

Table 2. Examples of vectors in the weight parameter space generated by the optimization algorithm.

4 Conclusions

In context of our evaluation strategy, several features seem to be less important or even redundant. One of the most important features seem to be the projection profile, supported by it's high AUC score in table 1. A projection profile "spikes" at ascenders and descenders. Matching using this feature only would give existence and position of ascenders/descenders a high impact. The optimization consistently removed (or significantly lowered) the weight on the contours, centre of weight, second moment, foreground/background transitions and the gradient of the lower contour. Even though the optimizer can not guarantee a global (or local) minimum, the approach was successful in terms of increasing the performance.

We show that the limits set on the DTW by the pruning makes even a random classification of the remaining word pairs perform adequately. We conclude that in previous studies of word spotting, the accuracy was to a large extent limited by the initial pruning step. The following DTW can not bring back false negatives and that limits the AUC to the interval $[0.69, 0.82]$. Compare this to using uniform weights with and AUC of 0.84, using only projection at an AUC of > 0.87 or the features in [1] without pruning where the AUC was 0.85. Using our optimization strategy, the evaluation results was pushed to > 0.89.

References

1. Rath, T.M., Manmatha, R.: Word image matching using dynamic time warping. In: Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on. Volume 2. (2003) II–521–II–527 vol.2
2. Rath, T.M., Manmatha, R.: Features for Word Spotting in Historical Manuscripts. In: International Conference on Document Analysis and Recognition. (2003) 218–222
3. Manmatha, R., Croft, W.B.: Word Spotting: Indexing Handwritten Archives. (1997)
4. Schwartz, R., LaPre, C., Makhoul, J., Raphael, C., Zhao, Y.: Language-independent ocr using a continuous speech recognition system. In: Pattern Recognition, 1996., Proceedings of the 13th International Conference on. Volume 3. (1996) 99–103 vol.3
5. Wienecke, M., Fink, G., Sagerer, G.: Toward automatic video-based whiteboard reading. International Journal of Document Analysis and Recognition (IJDAR) **7** (2005) 188–200
6. Pltz, T., Fink, G.: Markov models for offline handwriting recognition: a survey. International Journal on Document Analysis and Recognition (IJDAR) **12** (2009) 269–298
7. Marti, U.V., Bunke, H.: Using a statistical language model to improve the performance of an hmm-based cursive handwriting recognition system. International Journal of Pattern Recognition and Artificial Intelligence **15** (2001) 65–90
8. Kane, S., Lehman, A., Partridge, E.: Indexing george washingtons handwritten manuscripts. Center for Intelligent Information Retrieval, Computer Science Department, University of Massachusetts, Amherst, MA **1003** (2001)
9. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. Acoustics, Speech and Signal Processing, IEEE Transactions on **26** (1978) 43–49
10. Bertsekas, D.P., Bertsekas, D.P.: Nonlinear Programming. 2nd edn. Athena Scientific (1999)
11. Fischer, A., Keller, A., Frinken, V., Bunke, H.: Lexicon-free handwritten word spotting using character hmms. Pattern Recogn. Lett. **33** (2012) 934–942
12. Bradley, A.P.: The use of the area under the roc curve in the evaluation of machine learning algorithms. Pattern Recognition **30** (1997) 1145–1159