

## Fisher kernels for handwritten word-spotting

Florent Perronnin  
Textual and Visual Pattern Analysis  
Xerox Research Centre Europe, France  
Florent.Perronnin@xrce.xerox.com

Jose A. Rodriguez-Serrano\*  
Computer Vision Centre  
Universitat Autònoma de Barcelona, Spain  
cojar@lboro.ac.uk

### Abstract

*The Fisher kernel is a generic framework which combines the benefits of generative and discriminative approaches to pattern classification. In this contribution, we propose to apply this framework to handwritten word-spotting. Given a word image and a keyword generative model, the idea is to generate a vector which describes how the parameters of the keyword model should be modified to best fit the word image. This vector can then be used as the input of a discriminative classifier. We compare the performance of the proposed approach with that of a generative baseline on a challenging real-world dataset of customer letters. When the kernel used by the classifier is linear, the performance improvement is marginal but the proposed system is approximately 15 times faster than the baseline. If we use a non-linear kernel devised for this task, we obtain a 15% relative reduction of the error but the detector is approximately 15 times slower.*

### 1. Introduction

Handwritten word-spotting (HWS) is the task which consists in detecting keywords in collections of handwritten document images [6]. This is a typical two-class pattern classification problem: given a candidate word image and a keyword hypothesis, a match is declared if the score of the word image on the keyword model exceeds a threshold.

Two main types of word-spotting approaches can be identified. In *query-by-string* (QBS) approaches, character models – typically Hidden Markov Models (HMMs) – have been pre-trained. At query time the models of the characters forming the string are concatenated into a word-model. In *query-by-example* (QBE) approaches, one or several images of the word to be spotted are provided as queries and the most similar candidate images are returned. QBS and

QBE approaches have their own advantages and disadvantages. QBE approaches require examples of the word to be spotted which is not the case of QBS. On the other hand, QBS approaches require large amounts of labeled data to train character models which is not the case of QBE. In the following, we focus on the QBE scenario.

While early approaches to QBE HWS were based on holistic features [6, 4], the state-of-the-art is to characterize a word-image with a sequence of feature vectors extracted using a sliding window approach. In their influential work, Rath and Manmatha proposed to use the dynamic time warping (DTW) algorithm to match word images [9]. More recently, Rodríguez and Perronnin proposed to train a whole-word HMM using the query images and to score the candidate images on the HMM [11].

While the use of a HMM may improve the retrieval accuracy over DTW, especially in the challenging case of a multi-writer scenario where it can model the style variability, it has the inherent limitations of generative approaches to pattern classification. Indeed, while *generative* approaches focus on the accurate modeling of class-conditional probabilities, *discriminative* approaches focus directly on the problem of interest which is finding class boundaries. Despite the theoretical and often practical superiority of discriminative approaches, generative approaches still have a number of properties which make them attractive such as their ability to handle variable-length or missing data as is the case of the HMM.

In this article, we propose to apply the Fisher kernel (FK) principle to the HWS problem. The FK was introduced by Jaakkola and Haussler [3] to combine the benefits of the generative and discriminative worlds. The idea is to characterize a signal with a gradient vector derived from a generative probability model. In the HWS case, the signal corresponds to the sequence of vectors extracted from a word image and the probabilistic model is a HMM trained using samples of the word to be spotted. This gradient vector can then be used as the input of any discriminative classifier. In our case, we will make use of a kernel classifier based on sparse logistic regression (SLR) [5]. We will experiment

\*J.A. Rodriguez-Serrano was a visitor at the Xerox Research Centre Europe while this work was conducted. He is now with the Computer Science Department, Loughborough University, United Kingdom

with two kernels: the linear kernel and a non-linear kernel specially devised for this task.

We note that kernel classifiers have already been applied to problems related to HWS. One of the earliest attempts is that of Bahlmann et al. [1] who introduced a DTW kernel for the problem of online character recognition. While this framework could theoretically be applied to our case, we believe that the FK has a major benefit which is speed. Indeed, the classification of a sequence in [1] requires to compute many DTWs – one per support vector – whereas in the FK framework the classification involves vector-to-vector distances which are much faster to compute than DTWs. In the case of the linear kernel, a single dot product is computed which results in a very fast classification. As for the FK framework, it was applied by Sangnansat et al. [12] and Do and Artières [2] to the problem of online character recognition. However, to the best of our knowledge, this is its first application to HWS.

The remainder of the article is organized as follows. In the next section, we describe in detail the proposed solution. In section 3, we evaluate experimentally the proposed approach on a challenging real-world dataset of customer letters. We compare its performance, both in terms of classification accuracy and speed, with the baseline system of [11]. In the case of the linear kernel, the performance improvement we obtain is marginal but the proposed system is approximately 15 times faster to run than the baseline. In the case of the non-linear kernel, we obtain a 15% relative reduction of the error but the detector is approximately 15 times slower to run. Finally, conclusions are drawn.

## 2. Proposed solution

We start by reviewing the FK framework. We then turn to its application to the HWS problem. We also provide some details on the choice of the classifier. Finally, we summarize the training and test steps for clarity.

### 2.1 Fisher kernel principle

Let  $p$  be a probability density function (pdf) whose parameters are denoted  $\lambda$ . Let  $X = \{x_t, t = 1 \dots T\}$  be a set of samples which is supposed to have been generated by  $p$ . Jaakkola and Haussler propose to characterize  $X$  with the following gradient vector [3]:

$$G_\lambda(X) = \nabla_\lambda \mathcal{L}(X|\lambda). \quad (1)$$

where  $\mathcal{L}(X|\lambda)$  denotes the log-likelihood function  $\log p(X|\lambda)$ . Intuitively, the gradient of the log-likelihood describes the direction in which the model parameters should be modified to best fit the data. It transforms a variable-length sample  $X$  into a fixed length vector whose size depends on the number of parameters of the model.

This gradient vector can then be classified using any discriminative classifier. For those discriminative classifiers which use an inner product term – such as kernel classifiers – it is important to normalize the input vectors. In [3], the Fisher information matrix  $F_\lambda$  is suggested for this purpose:

$$F_\lambda = E_X [G_\lambda(X)G_\lambda(X)^T], \quad (2)$$

where  $T$  denotes the transposition operation and  $E_X$  the expectation under the random variable  $X$ . The normalized gradient vector is thus given by:

$$F_\lambda^{-1/2} \nabla_\lambda \log p(X|\lambda). \quad (3)$$

Because of the cost associated with its computation and inversion,  $F_\lambda$  is often approximated by the identity matrix and no normalization is performed. As explained in the next section, we will use a diagonal approximation of  $F_\lambda$ .

### 2.2 Fisher kernels for HWS

Let us now turn to the application of this general framework to the HWS problem. In our case,  $X$  is the sequence of feature vectors extracted from a candidate image. The pdf  $p$  models the word to be spotted.

A suitable statistical model for sequences of vectors is the HMM. HMMs have three sets of parameters: initial occupancy probabilities, transition probabilities and emission probabilities. In continuous-state HMMs, as is our case, it is standard to use Gaussian Mixture Models (GMMs) for emission probabilities. In such a case, the emission probability parameters can be subdivided into mixture weights, mean vectors and covariance matrices. We assume that the covariance matrices are diagonal as (i) any distribution can be approximated with an arbitrary precision by a weighted sum of Gaussians with diagonal covariances and (ii) the computational cost of diagonal covariances is much lower than the cost involved by full covariances.

The gradient vector is, by definition, the concatenation of the partial derivatives with respect to the model parameters. In the following, we focus on the derivatives with respect to the mean and variance as we found in preliminary experiments that the derivative with respect to the other parameters (initial occupancy probabilities, transition probabilities and mixture weights) only had a very small impact on the classification accuracy for our problem.

Let  $\mu_{i,j}$  and  $\sigma_{i,j}$  be respectively the mean and standard deviation of the  $j$ -th Gaussian in the  $i$ -th state. The superscript  $d$  denotes the  $d$ -th dimension of a vector. Let  $\gamma_t(i, j)$  denote the occupancy probability, i.e. the probability for observation  $x_t$  to have been generated in the  $i$ -th state by the  $j$ -th Gaussian. This posterior probability can be computed exactly using the forward-backward algorithm or approximately using the Viterbi algorithm (see for instance [8] for

an introduction to HMMs and their associated algorithms). Simple mathematical derivations provide the following formulas:

$$\frac{\partial \mathcal{L}(X|\lambda)}{\partial \mu_{i,j}^d} = \sum_{t=1}^T \gamma_t(i,j) \left[ \frac{x_t^d - \mu_{i,j}^d}{(\sigma_{i,j}^d)^2} \right], \quad (4)$$

$$\frac{\partial \mathcal{L}(X|\lambda)}{\partial \sigma_{i,j}^d} = \sum_{t=1}^T \gamma_t(i,j) \left[ \frac{(x_t^d - \mu_{i,j}^d)^2}{(\sigma_{i,j}^d)^3} - \frac{1}{\sigma_{i,j}^d} \right]. \quad (5)$$

It is important to note that these partial derivatives can be computed through the combination of the following statistics:  $\sum_{t=1}^T \gamma_t(i,j)$ ,  $\sum_{t=1}^T \gamma_t(i,j)x_t^d$  and  $\sum_{t=1}^T \gamma_t(i,j)(x_t^d)^2$  which are the quantities which are accumulated during Maximum Likelihood Estimation (MLE). Hence, we conclude that the cost of computing the gradient vector is (almost) identical to the cost of MLE training using only the sample  $X$ . Since MLE is a required building block in HMM-based recognition systems, the FK implementation represents only a small add-on to the already existing code.

As for  $F_\lambda$ , we extend the diagonal approximation derived in [7] for GMMs. Let  $f(\mu_{i,j}^d)$  and  $f(\sigma_{i,j}^d)$  be the terms on the diagonal of  $F_\lambda$  which correspond respectively to  $\partial \mathcal{L}(X|\lambda)/\partial \mu_{i,j}^d$  and  $\partial \mathcal{L}(X|\lambda)/\partial \sigma_{i,j}^d$  such that the normalized partial derivatives are  $f^{-1/2}(\mu_{i,j}^d)\partial \mathcal{L}(X|\lambda)/\partial \mu_{i,j}^d$  and  $f^{-1/2}(\sigma_{i,j}^d)\partial \mathcal{L}(X|\lambda)/\partial \sigma_{i,j}^d$ . We have (approximately):

$$f(\mu_{i,j}^d) = T \times w_i / (\sigma_{i,j}^d)^2, \quad (6)$$

$$f(\sigma_{i,j}^d) = 2 \times T \times w_i / (\sigma_{i,j}^d)^2. \quad (7)$$

Let  $D$  be the dimensionality of the feature vectors and  $M$  be the number of Gaussians in the HMM. The size of the gradient vector is thus  $2 \times D \times M$  in the case where we consider only the mean and variance. For our problem, typical values of  $D$  and  $M$  are on the order of 100, which means that the dimensionality of the gradient vector is on the order of 10,000. The hope is that this high-dimensional vector contains sufficient discriminative information for accurate classification.

### 2.3 Classification

We focus on the case of kernel classifiers. Given a test sample  $z$  and a set of training samples  $z_i$  with their associated labels  $y_i$  ( $y_i = +1$  if  $z_i$  corresponds to the class and  $-1$  if it does not), the score  $s(z)$  of  $z$  has the following form:

$$s(z) = f \left( \sum_{i=1}^N \alpha_i y_i K(z, z_i) + b \right), \quad (8)$$

where  $f$  is a non-decreasing function and where the parameters  $\alpha_i$  and  $b$  are learned. We used Sparse Logistic Regression (SLR) [5], i.e. in our case  $f$  is a sigmoid function and

at training time a Laplacian prior on the  $\alpha$  values is used for regularization. A good property of SLR is that it induces a sparse solution (i.e. many of the  $\alpha_i$  values are exactly 0) as is the case of the SVM. In practice, the SLR and SVM exhibit very similar performance [5].

We experimented with two types of kernels:

- The *linear* kernel. Its obvious advantage is speed at runtime as the sum  $\sum_{i=1}^N \alpha_i y_i K(z, z_i)$  can be rewritten as  $z^T (\sum_{i=1}^N \alpha_i y_i z_i) + b$  and classification requires just a dot product. We note however that the linear kernel relies on the assumption that the dot product is a good measure of similarity, which is not necessarily the case for gradient vectors.
- A *non-linear* kernel. We propose the following distance between gradient vectors:

$$D(z, z_i) = \left\| \frac{z}{\|z\|_1} - \frac{z_i}{\|z_i\|_1} \right\|_1, \quad (9)$$

where  $\|\cdot\|_1$  is the L1 norm (i.e. Manhattan distance). Experimentally, we found that Eq. (9) is a good dissimilarity measure between gradient vectors. The superiority of this distance for the considered problem can be motivated as follows. First, the vectors  $z$  and  $z_i$  are normalized because what matters when comparing two gradient vectors is not so much their norm as their direction. Second, it was found experimentally that the distribution of the values in a given dimension of the normalized vectors had a Laplacian shape and the appropriate distance when samples are distributed according to such a distribution is L1. To transform this distance into a similarity, we use the exponential function:

$$K(z, z_i) = \exp(-\beta D(z, z_i)) \quad (10)$$

where  $\beta$  is a parameter of the system. We note that  $K$  is not guaranteed to be a true Mercer kernel (as is the case for instance of the DTW kernel introduced in [1]) but that this is not a practical limitation of our system as the SLR training is still a convex optimization problem. To avoid the need of tuning  $\beta$ , we follow the common practice of setting its value to the inverse of the mean value of the distance  $D(\cdot)$  as estimated on a subset of the whole training set (see for instance [13]).

### 2.4 Summary

For clarity, we now summarize the training and test steps of the proposed algorithm. To train a detector for a given keyword, we assume that we have a set of positive images corresponding to that keyword and a set of negative images which do not correspond to that keyword. The steps are as follows:

1. For each image, extract a sequence of feature vectors.
2. Using the Baum-Welch algorithm (c.f. [8]), train an HMM modeling the keyword using only the positive images.
3. For each image, compute the gradient vector using Eq. (4) and (5). The occupancy probabilities  $\gamma_t(i, \hat{j})$  are computed using the Viterbi or the forward-backward algorithm. Normalize the gradient vectors using Eq. (6) and (7).
4. Using the normalized gradient vectors of both positive and negative images, train a discriminative classifier.

To test an image on a keyword hypothesis, one should follow these steps:

1. Extract a sequence of feature vectors from the image.
2. Compute and normalize the gradient vector.
3. Score the normalized gradient vector on the classifier and take a decision.

### 3. Experimental validation

In this section, we first describe the experimental conditions. We then report results for the proposed approach using the linear and non-linear kernel, and compare their performance with the system of [11]. Finally, we analyze the computational cost of the proposed approach.

#### 3.1 Experimental setup

Keyword-spotting experiments are carried out on a dataset containing 630 scanned letters written in French submitted to the customer department of a large corporation. This is a very challenging dataset because of the wide variety of writing styles (630) and because the writing is unconstrained and the letters contain artifacts such as words stricken-out and spelling mistakes. Word images are generated using standard segmentation techniques based on projection profiles and the clustering of gap distances.

In this work, we consider the problem of spotting 10 frequent words namely: (1) abonnement (subscription), (2) the company name (not provided in this article for obvious confidentiality reasons), (3) contrat (contract), (4) demande (request), (5) madame (Mrs), (6) monsieur (Mr), (7) résiliation (cancellation), (8) résilier (to cancel), (9) salutation (greeting) and (10) veuillez (if you please). The number of labeled positive samples varies from 208 to 750. The fact that these words are among the most frequent ones does not mean that they are easy to detect. Actually, these values have to be compared to the 180K candidate word hypotheses generated by the segmentation process. For instance,

the most frequent word corresponds to only 0.4% of the word images. Thus the probability that these samples appear by chance among the top retrieved results is very small. We chose the most frequent words to ensure that we have enough positive samples to estimate robustly the retrieval accuracy.

To reduce the number of candidate images to test, the initial set of words is pruned using holistic features fed into an SLR classifier. In the current setting, the fast rejection step eliminates on average 94% of the word images for each keyword class by falsely rejecting 10% of the real positives keywords. The numeric results reported in this section refer to the set of samples not eliminated by the fast rejection step.

The “surviving” examples are normalized with respect to skew, slant and text height. Then for each word image a sequence of features is extracted using a sliding window approach. We report results using the local gradient histogram (LGH) features as they have shown state-of-the-art performance [10]. In a nutshell, the sliding window is adjusted to the area containing the pixels and split into  $4 \times 4$  sub-windows. In each sub-window, a histogram of gradients is computed (8 bins) and the final vector is the concatenation of the 16 histograms which results in 128-dimensional features.

We use as a baseline the system of [11] which models keywords with a HMM and normalizes the scores using a GMM. Best results were obtained with keyword HMMs with 10 states and 16 Gaussians per state and a normalization GMM with 512 Gaussians. Scoring on the HMM keyword models is based on the Viterbi algorithm.

As for the proposed approach, we also used HMMs with 10 states per character to model the keywords. However, we used models with a single Gaussian per state as increasing the number of Gaussians did not improve the classification accuracy. To compute the gradient vectors, we estimated the posterior probabilities  $\gamma_t(i, j)$  using the Viterbi algorithm for a fair comparison with [11]. We note anyway that the use of the more exact forward-backward algorithm only had a very limited impact on the classification accuracy. For the SLR classification, the same regularization parameter was fixed for all keywords.

For the evaluation, we use precision-recall curves. To summarize the performance of a system with a single figure, we measure the Average Precision (AP) for the 10 keywords. The letters are split into 5 folds and evaluation is performed using 5-fold cross validation.

#### 3.2 Results

The results of the evaluation are reported in Table 1. We can see that the FK with a linear kernel performs only marginally better than the baseline system while the FK

word ID	[11]	FK-L	FK-NL
1	90.6	<b>93.1</b>	92.6
2	94.8	95.5	<b>95.6</b>
3	90.4	91.8	<b>92.3</b>
4	82.7	83.8	<b>85.8</b>
5	90.1	91.8	<b>92.8</b>
6	86.1	80.3	<b>88.9</b>
7	85.9	85.9	<b>86.1</b>
8	71.6	70.4	<b>74.6</b>
9	86.9	90.7	<b>91.0</b>
10	<b>90.1</b>	88.8	90.0
Average	86.9	87.2	<b>89.0</b>

**Table 1. Average precision (in %) for the baseline system of [11] and for the proposed system based on FK using a linear kernel (FK-L) and a non-linear kernel (FK-NL).**

with a non-linear kernel increases the AP by 2.1% absolute on average (15% relative decrease of error).

However, classification accuracy is not the only measure of performance that should be taken into account and the computational cost is also of paramount importance for applications of practical value. The following times were measured on a Dell™1950 which contains 2 quadricore processors (using a single core) at 3.1 Ghz and with 16 GB of RAM. For the baseline system, the computational cost is split between HMM scoring (20 ms) and GMM scoring (15 ms) which makes a total of 35 ms per word image per keyword model. For the proposed system, the computational cost can be split between the computation of the gradient vector (2.5 ms) and the classification cost (25  $\mu$ s for the linear kernel and 0.5s for the non-linear kernel). We note that the computation of the gradient vector is much faster than the HMM scoring for the baseline approach although both algorithms require to run Viterbi. The difference is due to the much lower number of Gaussians required by our approach (1 Gaussian per state versus 16 for the pure generative baseline). Hence, the cost of scoring a single word image on a single keyword model is on the order of 2.5 ms for the linear kernel (approximately 15 times faster than the baseline) and 0.5 s for the non-linear kernel (approximately 15 times slower to run than the baseline).

## 4. Conclusion

In this article, we applied the FK framework to the task of HWS. Given a word image and a keyword hypothesis, the idea is to generate a vector which describes how the parameters of the keyword generative model should be modified to best fit the word image. This vector can then used as in-

put to a discriminative classifier. In our case, we make use of a kernel classifier based on sparse logistic regression. We experimented with two kernels: the linear kernel and a non-linear kernel devised specially for this task. We compared the performance of the proposed approach with that of a pure generative baseline. When the kernel used by the classifier is linear, the performance improvement was shown to be marginal but the proposed system is approximately 15 times faster than the baseline. When we use a non-linear kernel, we obtained a 15% relative reduction of the error but the detector is approximately 15 times slower.

In this work, we focused on the QBE approach to HWS and therefore we assumed that the number of keyword hypotheses was limited (e.g. 10 in our experiments). An interesting challenge would be the application of the FK framework to QBS systems where the number of words to spot ranges from a few thousands to several tens of thousands.

## References

- [1] C. Bahlmann, B. Haasdonk, and H. Burkhardt. On-line handwriting recognition with support vector machines - a kernel approach. In *IWFHR*, 2002.
- [2] T. M. T. Do and T. Artires. Model selection with support vector machines. In *ICFHR*, 2008.
- [3] T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *NIPS*, 1999.
- [4] P. Keaton, H. Greenspan, and R. Goodman. Keyword spotting for cursive document retrieval. In *DIA*, 1997.
- [5] B. Krishnapuram, L. Carin, M. Figueiredo, and A. Hartemink. Sparse multinomial logistic regression: Fast algorithms and generalization bounds. *IEEE PAMI*, 27(6):957–968, 2005.
- [6] R. Manmatha, C. Han, and E. M. Riseman. Word spotting: A new approach to indexing handwriting. In *CVPR*, 1996.
- [7] F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *CVPR*, 2007.
- [8] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. of the IEEE*, 77:257–286, 1989.
- [9] T. M. Rath and R. Manmatha. Word image matching using dynamic time warping. In *CVPR*, 2003.
- [10] J. A. Rodríguez and F. Perronnin. Local gradient histogram features for word spotting in unconstrained handwritten documents. In *ICFHR*, 2008.
- [11] J. A. Rodríguez and F. Perronnin. Score normalization for HMM-based handwritten word spotting using a universal background model. In *ICFHR*, 2008.
- [12] P. Sangnansat, W. Asdornwised, and S. Jitapunkul. On-line Thai handwritten character recognition using hidden Markov models and support vector machines. In *ISCIT*, 2004.
- [13] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: an in-depth study. Technical Report RR-5737, INRIA, 2005.