Evaluation of lexicon size variations on a verification and rejection system based on SVM, for accurate and robust recognition of handwritten words

Yann Ricquebourg, Bertrand Coüasnon, Laurent Guichard

Evaluation of lexicon size variations on a verification and rejection system based on SVM, for accurate and robust recognition of handwritten words

Yann Ricquebourg[a], Bertrand Coüasnon[a] and Laurent Guichard[b]

[a]Université Européenne de Bretagne, IRISA/INSA, Rennes, France
[b]E2I SAS, Cesson-Sévigné, France

ABSTRACT

The transcription of handwritten words remains a still challenging and difficult task. When processing full pages, approaches are limited by the trade-off between automatic recognition errors and the tedious aspect of human user verification. In this article, we present our investigations to improve the capabilities of an automatic recognizer, so as to be able to reject unknown words (not to take wrong decisions) while correctly rejecting (i.e. to recognize as much as possible from the lexicon of known words).

This is the active research topic of developing a verification system that optimize the trade-off between performance and reliability. To minimize the recognition errors, a verification system is usually used to accept or reject the hypotheses produced by an existing recognition system. Thus, we re-use our novel verification architecture[1] here: the recognition hypotheses are re-scored by a set of support vector machines, and validated by a verification mechanism based on multiple rejection thresholds. In order to tune these (class-dependent) rejection thresholds, an algorithm based on dynamic programming has been proposed which focus on maximizing the recognition rate for a given error rate.

Experiments have been carried out on the RIMES database in three steps. The first two showed that this approach results in a performance superior or equal to other state-of-the-art rejection methods. We focus here on the third one showing that this verification system also greatly improves results of keywords extraction in a set of handwritten words, with a strong robustness to lexicon size variations (21 lexicons have been tested from 167 entries up to 5,600 entries) which is particularly relevant to our application context cooperating with humans, and only made possible thanks to the rejection ability of this proposed system. The proposed verification system, compared to a HMM with simple rejection, improves on average the recognition rate by 57% (resp. 33% and 21%) for a given error rate of 1% (resp. 5% and 10%).

Keywords: Handwritten word recognizer, Verification system, SVM re-scoring, Rejection method, Lexicon size variation, RIMES database

1. INTRODUCTION

In the context of document recognition, transcription of handwritten words is still challenging, especially in historical documents processing. Due to the difficulty or the degradation of the document handwritings, state-of-the-art automatic recognition is not yet able to fully transcribe this kind of document. Therefore, human help is necessary to assist the document analysis system, by correcting by hand the recognition difficulties or ambiguities. However those user confirmations may become tedious if too often submitted to the human, while avoiding wrong automatic decision is necessary.

A simple two-stage approach can be used, where each document is first processed by the system and thanks to its rejection capabilities, then the rejected words are annoted by the user, and so on. More evolved strategies tend to organize this cooperation by creating word clusters that "look the same" in order to optimize the resort

Further author information:
Yann Ricquebourg: yann.ricquebourg@irisa.fr
Bertrand Coüasnon: bertrand.couasnon@irisa.fr
Laurent Guichard: laurent.guichard@eii.fr

to the user, and to take benefit from the similarity among the cluster for his task, see Figure 1. We presented such a strategy,[2] at collection level, on documents for the French Revolution. To make it possible, the strategy relies on the rejection capabilities of the underlying classifier that must avoid recognizing wrongly and, on the other hand, must reach as much automatic recognition as possible. In our application context we address large collections of handwritten documents where we absolutely need to resort to humans for some difficult cases, see Figure 2a. But this strategy is only reasonable if the classifier has rejection capability, otherwise all the pages would be likely to be human-checked. Here, the system we propose will be able to outline the problematic images, given an acceptable error-rate. Moreover, these images to be human-checked will clustered with similarity to improve the process as mentioned before, see Figure 2b-c.

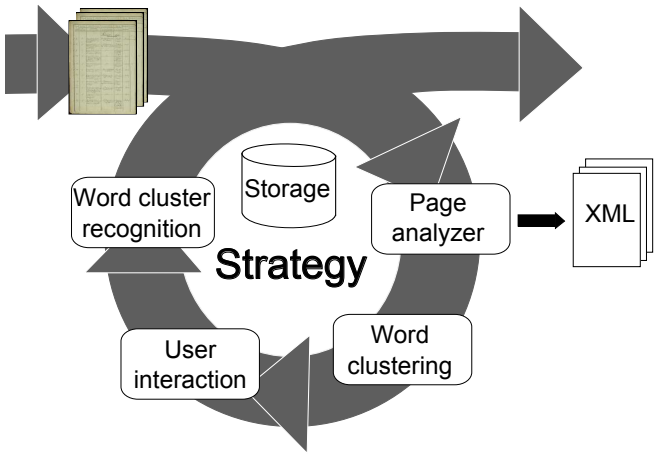

Figure 1. Example of strategy to process a collection of documents, including user interaction

Then, the interest in developing effective verification systems (VSs) for handwritten word recognition applications (HWR) that can distinguish when their outputs are not recognized with enough certainty (and should be consequently rejected) is still an active research topic. Commonly, VSs involve two parts: the confidence measures computation (CMs), which gives an idea of the achieved recognition quality of each word image, and the thresholding-based procedure, which stands for trading off between errors and rejections.

In the literature we can find a wide diversity of VSs for HWR. On the one hand are the VSs directly applying a rejection rule to the HWR hypotheses scores.[3–5] For HWR based on Hidden Markov Models (HMMs), which is by far the most successfully used statistical approach according to the state-of-the-art, VS rejection mechanisms rely usually on the same HMM decoding scores. Those approaches are limited by the intrinsic nature of the HWR, aimed at maximizing the recognition but not the rejection. On the other hand, there are some VSs which re-score HWR hypotheses before performing the accept/reject action. This is the case described in,[6] where a multi-layer perceptron (MLP) is used to reevaluate the hypotheses. Because this classifier is not specifically suitable for rejection tasks, the use of support vector machines (SVM) to re-score these HWR hypotheses emerges as a promising alternative, as they already proved their ability to verify isolated handwritten digits.[7,8]

As mentioned above, VS approaches rely on thresholding methods, which intend to adjust threshold values to decide whether accept or reject given recognized hypotheses. The formulation of the best error-reject trade-off and the related optimal rejection rule is given in.[9] According to this, the optimal error-reject trade-off is achieved only if the *a posteriori probabilities* of the classes are known exactly. As they are always affected by errors,[10] suggests the use of multiple rejection thresholds to obtain the optimal decision and reject regions. Nevertheless, in the field of HWR, most VSs do not take into account this and use just one single threshold. An inherent difficulty of the multi-threshold VSs, within the context of HWR based on HMMs, lies in how to define the appropriate classes associated to each of the threshold, which do not necessarily correspond to the lexicon words. Another difficulty is also the tuning of rejection thresholds, which has been already investigated in,[3,10–12] where different algorithms are proposed but none of them guarantee an optimal solution.

This paper summarizes two main contributions which aim at improving both rejection and recognition capabilities of the verified HWR. The first one describes, in section 2, a VS approach which uses an alternative

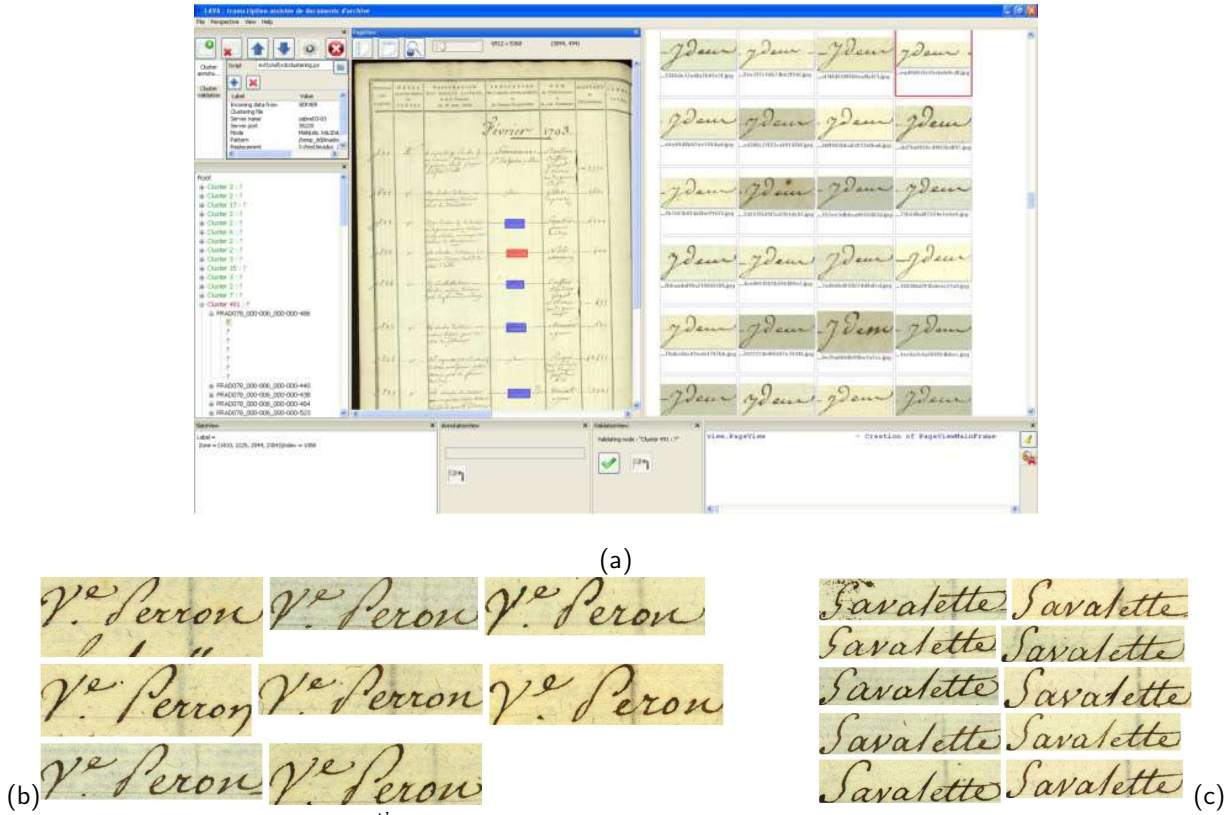

Figure 2. (a) Application concerning 18^{th} century French revolutionary sales documents, with human interactions for rejected (unrecognized) words, then submitted to the user clustered with similar words to optimize the interaction and the results: here cluster "Idem" (b) cluster "Peron" (c) cluster "Savalette"

SVM-based confidence measure relying on the grapheme segmentation information from the HMMs Viterbi decoding, and applies multiple thresholds to optimize the error-rejection trade-off. The second contribution focuses on presenting, in section 3, our specific algorithm for computing multiple rejection threshold values based on dynamic-programming which, unlike other approaches, guarantees an optimal solution. Then the resulting system is evaluated on the RIMES database (which contains around 67,000 French handwritten words) in section 4. Those results and comparisons tend to focus on the fact that this study supplies a system suitable for our application context: giving robustness to lexicon size variations, thanks to the rejection capabilities that we added.

2. PROPOSED VERIFICATION SYSTEM APPROACH

The proposed VS is suitable for HWR based on grapheme/character-segmentation (explicit or implicit). For a given word image input s, the HWR outputs the N-best recognized hypotheses along with their corresponding grapheme segmentations and recognition scores. This list of N-best hypotheses serves as input of our VS approach. To represent this list, we use the following notation: $\langle h_1 = (w_1, r_1), \ldots, h_N = (w_N, r_N)\rangle$, where w_i and r_i denote respectively the transcription and grapheme segmentation of the i^{th} recognized hypothesis h_i of word image s. In turn, each hypothesis $h_i = (w_i, r_i)$ is associated with a sequence of grapheme-label and sub-image pairs: $\langle (c_{i,1}, g_{i,1}), \ldots, (c_{i,n_i}, g_{i,n_i})\rangle$, where n_i is the number of recognized (grapheme/character) labels of the corresponding hypothesis transcription w_i. Furthermore, each h_i has an associated probability $P_{HWR}(h_i)$ emitted by the HWR.

Our VS approach proposed previously[1] is composed of three successive steps: *grapheme feature extraction*, *N-best hypotheses re-scoring* and *hypothesis selection and verification*, that we explain hereafter in more details and for clarity of the experiments.

2.1 Feature extraction

This first step makes use of the segmentation information provided by HWR to split input word image into the corresponding grapheme sub-images (i.e. character images in our case). Then, a feature extraction process transforms each of these sub-images into a 95-dimensional real-value vector composed of the following set of features:

- 8^{th} order Zernike moments[13] (giving 45 components). We use to retain these features, unlike interesting studies,[14] as they experimentally turned out to be really efficient (enabling us to reach first rank[15] in two of the RIMES recognition tasks, concerning alphabetic and alphanumeric symbols):

$$Z_{pq} = \frac{p+1}{\pi} \int_0^{2\pi} \int_0^{+\infty} \overline{V_{pq}(r,\theta)} f(r,\theta)\, r \,\mathrm{d}r \,\mathrm{d}\theta \tag{1}$$

  where p is the radial magnitude and q is the radial direction, and $\overline{V}$ denotes the complex conjugate of a Zernike polynomial V, defined by $V_{pq}(r,\theta) = R_{pq}(r)e^{iq\theta}$ where $p-q$ is even with $0 \leqslant q \leqslant p$ and R is a real-valued polynomial:

$$V_{pq}(r,\theta) = R_{pq}(r)e^{iq\theta} \quad \text{where } p-q \text{ is } \textit{even} \text{ and } 0 \leqslant q \leqslant p$$

$$R_{pq}(r) = \sum_{m=0}^{\frac{p-q}{2}} (-1)^m \frac{(p-m)!}{m!(\frac{p-2m+q}{2})!(\frac{p-2m-q}{2})!}\, r^{p-2m}$$

- Histograms of 8-contour directions using Freeman chain code representation (each grapheme zoned in 6 areas (2x3), implying 6 histograms, giving 48 components);
- Normalized foreground pixels distributions onto ascender and central grapheme areas (giving 2 components). These two grapheme areas (ascender and central also including descender) are defined on the whole word image of corresponding graphemes by both base- and upper-lines of handwritten text.

2.2 Re-scoring

The second step performs a re-scoring of each N-best recognized hypotheses by using SVM classifiers, each of which modeling a specific grapheme class c from the whole grapheme classes set considered in the recognition. In this way, given a pair $(c_{i,j}, g_{i,j})$ with $i \in [1, N]$ and $j \in [1, n_i]$, the corresponding SVM assigns to it a new score $P_{\text{SVM}}(c = c_{i,j}|g_{i,j})$ which represents the (approximate) posterior probability that grapheme $g_{i,j}$ belongs to class $c_{i,j}$. The SVM classifiers used here rely on a Gaussian kernel as it embeds intrinsic knowledge, which is particularly well suited for verification. Indeed, the Gaussian radial basis function of the kernel has the following expression:

$$K_g(x, x_k) = \exp(-\gamma||x - x_k||^2) \tag{2}$$

where γ is a user defined parameter, x a sample vector and x_k a support vector. The support vectors are selected during the training phase. The above expression tells that the kernel value decreases as the sample vector gets away from the support vectors which enables the rejection of unknown or damaged graphemes.

The SVM output score is approximated to a *posterior probability* by using the *softmax* function, as described in.[16]

2.3 Selection and verification

Once all individual grapheme probabilities have been computed, a global SVM score of hypothesis h_i is calculated as the geometric mean of their respective grapheme scores:

$$P_{\text{SVM}}(h_i) = \sqrt[n_i]{\prod_{j=1}^{n_i} P_{\text{SVM}}(c = c_{i,j}|g_{i,j})} \tag{3}$$

We noticed out of some informal experiments that this way of computing the SVM global score works properly well for this case. Moreover, this makes the SVM score independent from hypothesis length (number of graphemes) and thereby comparable across different length hypotheses.

The final confidence measure (CM) of hypothesis h_i is then computed by linearly combining their respective global HMM score (given by the HWR system) and SVM score:

$$P(h_i) = \alpha P_{\text{SVM}}(h_i) + (1-\alpha) P_{\text{HMM}}(h_i) \quad \forall i \in [1, N] \tag{4}$$

This linear combination of classifier scores aims at balancing their effects by the empirically tuned coefficient α.

Once all hypotheses of the N-best list have been re-scored, the third and last step is in charge to select the best one (i.e. with the maximal CM score) and to perform the accept/reject action on it. In order to do this, the hypotheses are first re-ordered according to their new CM scores, defining a new list: $\langle \hat{h}_1, \ldots, \hat{h}_N \rangle$, such that $P(\hat{h}_i) \geqslant P(\hat{h}_j) \quad \forall ij, 1 \leqslant i < j \leqslant N$. Then, the reject/accept decision is performed by a thresholding mechanism using the computed difference between the two best re-scored hypotheses

$$d_{12} = P(\hat{h}_1) - P(\hat{h}_2)$$

as a value to be compared with the concerned threshold. Experiments conducted by other works[6] have shown that this strategy gives the best results.

As was mentioned in section 1, the proposed verification mechanism is based on multiple class-dependent thresholds. To define these classes, we have clustered all word transcriptions into different length-classes from the HWR lexicon according to their length. It is worth noting that the use of length-class-dependent thresholds serves to compensate the inaccuracy of the *a posteriori probabilities* mentioned earlier and also somewhat to mitigate the problem related to the empirical normalization that does not make fully comparable, for example, 10-characters words with 2-characters words. Formally, the set of length-classes is defined as:

$$\Omega = \{length(w) : w \in \text{Lex}\}$$

where *length* is a function returning the number of graphemes of word transcription w. We also use $\omega_j \in \Omega$ with $j \in [1, |\Omega|]$ to denote an element belonging to Ω. Thus, each of the length-classes $\omega_1, \omega_2, \ldots, \omega_{|\Omega|}$ has been linked to a respective threshold $t_1, t_2, \ldots, t_{|\Omega|}$, whose value is set up during the tuning phase. The description of this tuning is detailed in section 3.

For a given selected hypothesis $\hat{h}_1$ and its associate threshold $t_{\hat{\jmath}}$ ($t_{\hat{\jmath}} \rightarrow \omega_{\hat{\jmath}} = length(\hat{h}_1)$) the verification process performs the accept/reject action of word image s, following:

if $d_{12} \geqslant t_{\hat{\jmath}}$ **then** accept $\hat{h}_1$ **else** reject $\hat{h}_1$

3. MULTIPLE THRESHOLDS TUNING ALGORITHM

As described above, the verification system presented here rely on a set of previously set-up thresholds. Looking for the best thresholds is not a trivial problem, involving a combinatorial optimization over all their possible values.

Let S be a validation set of word images samples on which threshold values tuning is carried out. Likewise, let $S_i \subseteq S$, $i \in [1, |\Omega|]$ be sets of word samples with the same lengths:

$$S_i = \{w : length(w) = \omega_i, w \in \text{Lex}, \omega_i \in \Omega\}$$

Additionally, the following definitions for *performance* (PFR), *error rate* (ER) and *rejection rate* (RR) for our VS will be adopted:

$$\text{PFR} = \frac{Corr}{|S|} \quad \text{ER} = \frac{Err}{|S|} \quad \text{RR} = 1 - \text{PFR} - \text{ER} \tag{5}$$

where $Corr$ and Err are respectively the number of accepted words correctly and incorrectly classified.

In a similar way,[10] the problem of tuning a set of thresholds $t_1, \ldots, t_{|\Omega|}$ can be formulated in terms of PFR and ER as follows:

$$\begin{cases} \max\limits_{t_1,\ldots,t_{|\Omega|}} \mathrm{PFR}(t_1, \ldots, t_{|\Omega|}) \\ \\ \mathrm{ER}(t_1, \ldots, t_{|\Omega|}) \leqslant \mathrm{ER}_{max} \end{cases} \tag{6}$$

where ER_{max} is a given maximal error rate. The final goal here is to find the threshold values that maximize the performance of the system without exceeding ER_{max}.

Existing state-of-the-art algorithms for multiple thresholds tuning are not optimal.[10–12] The new tuning-threshold algorithm presented here is inspired from the *0-1 KnapSack problem* resolution based on dynamic programming.[17] Actually, this dynamic-programming-based approach leans on expression (7) rather than (6), where absolute values $Corr$ and Err are used instead of PFR and ER:

$$\begin{cases} \max\limits_{t_1,\ldots,t_{|\Omega|}} Corr(t_1, \ldots, t_{|\Omega|}) \\ \\ Err(t_1, \ldots, t_{|\Omega|}) \leqslant Err_{max} \end{cases} \tag{7}$$

For convenience reasons, we define the auxiliary function $\mathbf{F} : s \mapsto (Corr_s, Err_s, P_s) \quad s \in S_i, \forall i \in [1, |\Omega|]$, which returns, for each $s \in S_i$, the associated P_s (corresponding to d_{12} in our VS), as well as the $Corr_s$ and Err_s (number of samples correctly and incorrectly classified) computed on the samples $s' \in S_i$ such as $P_{s'} \geqslant P_s$. Furthermore, we introduce the accumulator function $A(l, Err)$, which returns the maximal number of well recognized samples that can be obtained with a number of errors lower or equal to Err considering only samples belonging to the class sample sets: $S_1, \ldots, S_l$ where $l \in [1, |\Omega|]$. Thus, $A(l, Err)$ can be recursively defined as follows:

$$\begin{cases} A(0, Err) = \max\limits_{s \in S_1, Err_s \leqslant Err} Corr_s \\ \\ A(l, Err) = \max\limits_{s \in S_l, Err_s \leqslant Err} A(l-1, Err - Err_s) + Corr_s \end{cases} \tag{8}$$

The algorithm 1 finds the optimal solution for $A(|\Omega|, Err_{max})$ using dynamic programming. Computation of $A(l, Err)$ is made iteratively until $l = |\Omega|$ and $Err = Err_{max}$. For each iteration, the sample that maximizes $A(l, Err)$ is stored in the auxiliary variable $B(l, Err)$ enabling to recover the threshold values set which maximized $A(|\Omega|, Err_{max})$. Basically, the running time of this algorithm depends on the size of validation set and the given maximal *error rate* $O(|S| \times Err_{max})$. The algorithm 2 recovers the threshold values by backtracking through the information stored in $B(l, Err)$.

4. EXPERIMENTS

4.1 Experimental setup

Three kinds of experiments were conducted.

First ones aimed at demonstrating that our Verification System with its SVM-based CM and its novel multiple thresholds computation mechanism improves the trade-off between error and rejection compared to other systems already published.

Second experiments comparatively evaluated our multiple thresholds tuning algorithm.

Third experiments addressed the issue of our VS robustness in real application context where the configuration lexicon may be smaller or larger than the set of sample labels.

Those experiments have been carried out on the RIMES database from the ICDAR 2009 and 2011 handwritten words recognition competitions.[18,19] The database contains a total of about 67 000 French handwritten words with their transcriptions. Table 1 presents basic statistical information of the corpus along with the partition definition used to carry out the experiments.

Algorithm 1 Forward pass: Compute $A(|\Omega|, Err_{max})$

s_0: default sample defined by $\mathrm{F}(s_0) = (0, 0, 1.0)$
{// Initialization:}
for $Err = 0$ **to** Err_{max} **do**
  $A(0, Err) \leftarrow 0$
end for
{// Fill the accumulator A:}
for $l = 1$ **to** $|\Omega|$ **do**
  for $Err = 0$ **to** Err_{max} **do**
    $A(l, Err) \leftarrow A(l-1, Err)$
    $B(l, Err) \leftarrow s_0$
    for all $s \in S_l$ **do**
      $(Corr_s, Err_s, \_) \leftarrow \mathrm{F}(s)$
      if $Err_s \leqslant Err$ **then**
        $aux_s \leftarrow A(l-1, Err - Err_s) + Corr_s$
        if $aux_s > A(l, Err)$ **then**
          $A(l, Err) \leftarrow aux_s$
          $B(l, Err) \leftarrow s$
        end if
      end if
    end for
  end for
end for

Algorithm 2 Backward pass: Track back the thresholds

t: set of thresholds to be tuned
{// Initialization:}
$l \leftarrow |\Omega|$
$Err \leftarrow Err_{max}$
{// Get the thresholds:}
while $l > 0$ **do**
  $s \leftarrow B(l, E)$
  $(\_, Err_s, P_s) \leftarrow \mathrm{F}(s)$
  $t(l) \leftarrow P_s$ {// Threshold for class ω_l}
  $E \leftarrow Err - Err_s$
  $l \leftarrow (l-1)$ {// Next class}
end while

The HWR used here (denoted **HMM-ST** hereafter, for *HMM with Single Threshold*), is a standard HMMs-based recognizer which extracts feature vectors using a sliding window. It models lexicon words by a concatenation of continuous left-to-right grapheme HMMs and uses the Viterbi algorithm to look for the HMM-concatenated models that maximize the probability to produce the given feature vector sequence (for details, see[19]). As the grapheme HMMs are restricted to lower case letters without accent, the data sets where first normalized.

The SVM classifiers used to re-score graphemes (**SVM-MT-DPR** for *SVM with Multiple Thresholds from*

Table 1. Basic statistics of the RIMES-DB words corpus and its standard partition

	2009			2011	Total
	Training	Valid.	Test	Test	
words	44 196	7 542	7 464	7 774	66 976
charact.	230 259	39 174	38 906	40 185	308 339

Dynamic Programming) have Gaussian kernels and were trained with the one-against-all strategy for multi-class SVM classification. In this sense, grapheme samples to train SVM classifiers were obtained through segmenting the word images of the training set with our HMMs-based HWR in forced alignment mode.

To assess our contributions (VS and multiple thresholds tuning algorithm), comparisons have been made with other methods already published. The RIMES-DB partition sets used in the experiments are highlighted in table 1. Parameters learning, as well as multiple threshold tuning have been performed on the 2009 validation set for all the tested algorithms. Finally, reported results of the three experiments have been obtained on the 2009 and 2011 test sets.

For third experiments, a range of 21 increasing size lexicons was built from the 5 600 words lexicon of the 2011 ICDAR competition, called l_{full} lexicon. By regularly removing 150 words from the exact lexicon l_{exact} made of 1 667 unique labels of the 2011 test set, lexicons l_{1-} to l_{10-} were created having 1 517 to 167 entries. In the other direction, lexicons l_{1+} to l_{9+} were also regularly produced by adding words from l_{exact} to l_{full} lexicons.

For the VS using multiple rejection thresholds, a number of 17 thresholds were set according to the number of classes produced by regrouping the lexicon words with the same lengths, (in other words lexicons contain words varying from 1 to 17 characters). The number of hypotheses generated by the HWR for each recognized word image was set to 10.

The third experiments are comparing our VS to the reference system **HMM-ST** when the input lexicon is varying in size. To evaluate the results of the VS proposed, the following measures have been adopted:

- *True rejection rate* (TRR): the number of wrong recognized words that are rejected divided by the number of well recognized words.
- *False rejection rate* (FRR): the number of well recognized words that are rejected divided by the number of wrong recognized words.
- *Lexicon coverage* (LC): LC $= \frac{|Lex|}{|UL|}$ where UL denotes the set of the (unique) sample labels.
- *Performance* (PFR) and *Error Rate* (ER): already defined in section 3, expression (5).
- *Lexicon relative performance* (LPFR): LPFR$= \frac{Corr}{|S_{Lex}|}$ where S_{Lex} is the number of words in data set whose ground truth label is present in the lexicon.

We also use the *Receiver Operating Characteristic* (ROC) curve, that plots FRR against TRR for different thresholds values. The area under a ROC curve provides an adequate overall estimation of the classification accuracy. This area is denoted as AROC. It is worth noting that an AROC value of 1.0 would indicate that all words can be correctly classified. Both ROC curves and the AROC measure are used to conveniently evaluate and compare our VS performance against other already published approaches. The *Performance* (PFR) versus *Error Rate* (ER) curve is also plotted to demonstrate the increase of well recognized words brought by the VS.

4.2 Evaluation

As we compared our system to others,[1] our VS approach revealed that SVM with single or multiple thresholds were the best performing approaches in the FRR range between 0% and 30%. and corroborated the proposed CM quality. Furthermore, **SVM-MT-DPR** approach outperforms all of the others, including SVM with single threshold. It is worth mentioning also the improvement in term of performance even without rejection. Indeed, the performance of the base HWR, **HMM-ST**, increases from 78.6% to 83.7% when multi-threshold-based scheme is incorporated.

For this algorithm detailed here to determine the multiple thresholds, previous experiments also showed the good performance of the **SVM-MT-DPR**, with ROC curves for FRR values over other competitors, as well as its good generalization ability. In addition this algorithm turned out to be 6 times less CPU consuming that state-of-the-art competitors (see[1]).

Concerning the robustness of our VS, Table 2 details the performances of the reference approach (**HMM-ST**) and ours (**SVM-MT-DPR**), evaluated on the 2011 RIMES-DB Test (7,774 handwritten words). It shows

AROC values and lexicon relative performances (LPFR) for error rates fixed to meaningful percentages (1%, 5% and 10%) obtained for the 21 lexicons.

An overall comparison of the AROC means, computed over all lexicons (illustrated on Figure 3 for the l_{exact} lexicon) shows an area difference of 0.06 between **SVM-MT-DPR** and **HMM-ST**, which obtain 0.91 and 0.85 respectively. This global superiority is confirmed when looking at LPFR means. Our VS reaches an average LPFR of 50.3% (resp. 72.6% and 77.8%) for a constant ER of 1% (resp. 5% and 10%) while **HMM-ST** gets 32% (resp. 54.7% and 64.2%). It confirms the results obtained in the previous experiments.

Table 2. Evaluation (on 7,774 handwritten words) of **SVM-MT-DPR** (S2) and **HMM-ST** (S1) robustness to lexicon variations (l_{10-} has 167 entries, l_{exact} has 1,600 entries and l_{full} has 5,600 entries)

Lexicon	LC	AROC		Max theoric. PFR	LPFR for a constant ER set to					
					1%		5%		10%	
		S1	S2		S1	S2	S1	S2	S1	S2
l_{10-}	0.11	0.86	0.91	14.7	10.5	32.1	35.6	63.1	47.1	70.9
l_{9-}	0.20	0.86	0.93	21.1	19.7	36.8	44.8	68.9	55.8	74.8
l_{8-}	0.30	0.85	0.92	29.4	19.4	38	44.9	70.5	56.3	76.4
l_{7-}	0.39	0.83	0.90	40.2	16	33.2	38.9	65.85	51.9	71.9
l_{6-}	0.49	0.82	0.89	51.4	12.2	24.9	38.4	63.4	51.7	70.3
l_{5-}	0.58	0.83	0.89	60.5	14.6	26.6	40.6	63.4	53.9	70.5
l_{4-}	0.67	0.82	0.90	71.9	18.5	36.4	44.2	63.5	56.7	72
l_{3-}	0.77	0.86	0.92	86.2	32.6	52.6	57.8	66	66.5	78
l_{2-}	0.87	0.87	0.91	93.7	37.5	56	61.3	74.6	69	79.7
l_{1-}	0.94	0.87	0.92	98.1	43.7	62.3	63.9	77	71.6	81
$l_0(=l_{exact})$	1.0	0.88	0.92	100	47	64.2	65.9	79	73.5	82.7
l_{1+}	1.28	0.88	0.92	100	46.5	63.7	65.8	78.9	73.4	82.7
l_{2+}	1.54	0.88	0.92	100	45.6	63.2	65.6	78.7	73.1	82.5
l_{3+}	1.84	0.83	0.92	100	44.1	62.3	63.9	77.7	71.7	81.6
l_{4+}	2.15	0.83	0.91	100	43.7	61.2	63.2	77.2	71.2	81.3
l_{5+}	2.46	0.83	0.91	100	41.7	60	61.8	76.9	70.1	80.9
l_{6+}	2.76	0.86	0.91	100	40.6	59.1	61.2	76.3	69.4	80.4
l_{7+}	3.06	0.86	0.91	100	36.7	57.9	59.4	75.3	67.9	79.8
l_{8+}	3.36	0.85	0.91	100	35.2	56.4	57.9	74.1	66.7	79
l_{9+}	3.67	0.87	0.91	100	33.8	55	56.8	73.5	65.7	78.5
$l_{10+}(=l_{full})$	3.81	0.85	0.90	100	32.8	53.5	55.8	72.8	64.7	77.9
Mean		0.85	0.91		32	50.3	54.7	72.6	64.2	77.8
Std. dev.		0.019	0.008		12.3	13.2	10.2	5.3	8.2	4.2

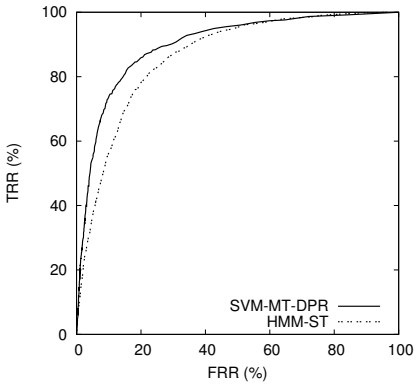

Figure 3. ROC curves for each of the two recognizers: HMM-ST is the base recognizer to be verified, SVM-MT-DPR is our verification-rejection system combined with the base recognition.

Nevertheless, what we want to evaluate here is the robustness of the VSs to lexicon size variations. For this, the standard deviation of the different measures is relevant. Thus, **SVM-MT-DPR** AROC standard deviation

is more than twice lower than the one of **HMM-ST**. In the same way, the LPFR variations when the lexicon is reduced or increased are less important for our VS. This is also what we want to reach in our context. As visible on Figure 4:

- The performance of our VS is quite stable as the lexicon size increases over the l_{exact} size (in other words, when including more words than necessary): this demonstrates the good stability of this system, remaining a significant improvement over the base one.
- Moreover, the performance is not too drastically collapsing when the lexicon size decreases (when it contains fewer words than needed): this demonstrates the good ability of this system to reject unknown words.

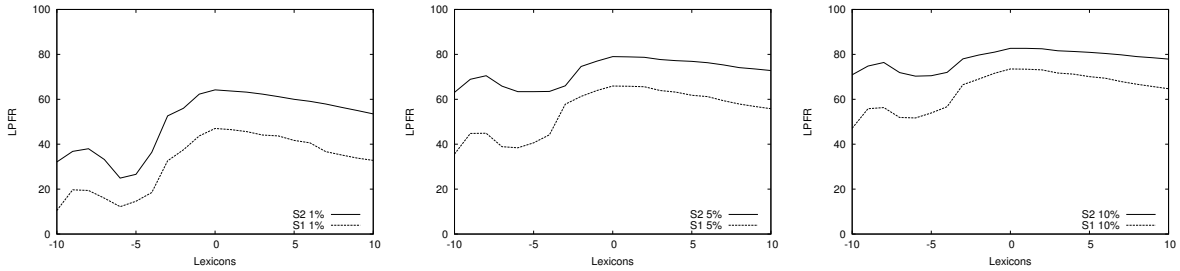

Figure 4. Lexicons, from Table 2, relative performances (LPFR) for a constant Error Rate of 1% (left), 5% (center) and 10% (right) compared for systems S1 (HMM-ST) and S2 (SVM-MT-DPR).

Those two results are the ones we needed to achieve in our application context where the user may be asked for confirmation in the recognition process: we need as many correct automatic answers as possible, even when the system has been in production for a long time with an important known word set. But the system takes also good decisions when it's facing a new area with few knows words, and will nevertheless prompt to the human only the unknown words, correctly rejected, that will progressively increase the lexicon.

It may be worth wondering why, out of experimentations, there is a noticeable decrease near l_{5-} in each case of Figure 4. From our point of view, this comes from the fact that with a smaller lexicon, the system has more chances to get ambiguity between an image (out of lexicon) and several other words of the reduced lexicon, thus involving more rejected words for ambiguity. Then, those cases became rare as the lexicon become even smaller.

5. REMARKS AND CONCLUSION

This paper purpose addresses rejection ability: it introduces an alternative verification system (independent from the given prior HWR), using a confidence measure based on SVMs re-scoring and multiple rejection thresholds to verify handwritten word recognized hypotheses. The experimental results obtained show that the proposed approach boosts the rejection capabilities of the HWR as, for example, the performance increases from 53.6% to 68.4% for an error rate set to 2.5%. It also improves the global recognition performance which rises from 78.6% to 83.7% when rejection is disabled.

A specific algorithm to tune multiple rejection thresholds has also been presented. It was confirmed experimentally that this tuning algorithm based on dynamic-programming produces very optimum results and is less time-consuming than other published algorithms.

We also experimentally showed on the 2011 RIMES-DB test (7,774 handwritten words), that the proposed verification system is robust to lexicon size variations, from 167 entries up to 5,600 entries, that makes it suitable for a context of optimal rejection to cooperate with a human user. We wanted to study how the system behave in a realistic application context, where the first lexicons are quite small and will grow as the human users validate the unknown words: the recognizer turned out to keep quite stable results. Moreover this proposed verification system, compared to a HMM with simple rejection, improves on average the recognition rate by 57% (resp. 33% and 21%) for a given error rate of 1% (resp. 5% and 10%).

REFERENCES

[1] Guichard, L., Toselli, A. H., and Couasnon, B., "Handwritten word verification by svm-based hypotheses re-scoring and multiple thresholds rejection," in [*ICFHR*], **1**, 57–62 (2010).

[2] Guichard, L., Chazalon, J., and Couasnon, B., "Exploiting collection level for improving assisted handwritten words transcription of historical documents," in [*ICDAR*], 875–879 (2011).

[3] Koerich, A. L., "Rejection strategies for handwritten word recognition," in [*IWFHR*], 479–484 (2004).

[4] Kapp, M. N., Freitas, C., and Sabourin, R., "Handwritten Brazilian month recognition: An analysis of two NN architectures and a rejection mechanism," *IWFHR* **0**, 209–214 (2004).

[5] Madhvanath, S., Kleinberg, E., and Govindaraju, V., "Holistic verification of handwritten phrases," *TPAMI* **21**(12), 1344–1356 (1999).

[6] Koerich, A. L., Sabourin, R., and Suen, C. Y., "Recognition and verification of unconstrained handwritten words," *TPAMI* **27**(10), 1509–1522 (2005).

[7] Bellili, A., Gilloux, M., and Gallinari, P., "An mlp-svm combination architecture for offline handwritten digit recognition," *IJDAR* **5**, 244–252 (July 2003).

[8] Chatelain, C., Heutte, L., and Paquet, T., "A two-stage outlier rejection strategy for numerical field extraction in handwritten documents," in [*ICPR*], **3**, 224–227 (2006).

[9] Chow, C. K., "On optimum error and reject tradeoff," *Information Theory Society* **16**, 41–46 (Jan. 1970).

[10] Fumera, G., Roli, F., and Giacinto, G., "Reject option with multiple thresholds," *Pattern Recognition* **33**, 2099–2101 (2000).

[11] Kapp, M. N., de A. Freitas, C. O., and Sabourin, R., "Methodology for the design of NN-based month-word recognizers written on Brazilian bank checks," *Image and Vision Computing* **25**(1), 40 – 49 (2007). SIBGRAPI.

[12] Mouchere, H. and Anquetil, E., "A unified strategy to deal with different natures of reject," in [*ICPR*], **2**, 792–795 (2006).

[13] Teague, M. R., "Image analysis via the general theory of moments," *Journal of the Optical Society of America (1917-1983)* **70**, 920–930 (August 1980).

[14] Camastra, F. and Vinciarelli, A., "Cursive character recognition by learning vector quantization," *Pattern Recognition Letters* **22**, 625–629 (2001).

[15] Grosicki, E., Carré, M., Brodin, J.-M., and Geoffrois, E., "Results of the RIMES evaluation campaign for handwritten mail processing," in [*ICDAR*], (2009).

[16] Milgram, J., Cheriet, M., and Sabourin, R., "Estimating accurate multi-class probabilities with support vector machines," in [*IJCNN*], **3**, 1906–1911 (2005).

[17] Martello, S. and Toth, P., [*Knapsack Problems: Algorithms and Computer Implementations*], John Wiley & Sons, Chichester, NY, revised ed. (1990).

[18] Grosicki, E. and Abed, H. E., "ICDAR 2009 handwriting recognition competition," in [*ICDAR*], (2009).

[19] Grosicki, E. and Abed, H. E., "ICDAR 2011 French handwriting recognition competition," in [*ICDAR*], (2011).