A Dempster–Shafer Theory based combination of handwriting recognition systems with multiple rejection strategies

Yousri Kessentini, Thomas Burger, Thierry Paquet

A Demspter-Shafer Theory based combination of Handwriting Recognition Systems with multiple rejection strategies

Yousri Kessentini[a,b], Thomas Burger[c], Thierry Paquet[b]

[a] *University of Sfax, ISIMS, MIRACL Laboratory ISIM Sfax, B.P. 242, 3021, Sfax, Tunisie*
[b] *Université de Rouen, Laboratoire LITIS EA 4108, site du Madrillet, St Etienne du Rouvray, France*
[c] *iRTSV-BGE (Univ. Grenoble Alpes - CNRS - CEA - INSERM), 38000 Grenoble, France*

Abstract

Dempster-Shafer theory (DST) is particularly efficient in combining multiple information sources providing incomplete, imprecise, biased, and conflictive knowledge. In this work, we focused on the improvement of the accuracy rate and the reliability of a HMM based handwriting recognition system, by the use of Dempster-Shafer Theory (DST). The system proceeds in two steps: First, an evidential combination method is proposed to finely combine the probabilistic outputs of the HMM classifiers. Second, a global post-processing module is proposed to improve the reliability of the system thanks to a set of acceptance/rejection decision strategies. In the end, an alternative treatment of the rejected samples is proposed using multi-stream HMM to improve the word recognition rate as well as the reliability of the recognition system, while not causing significant delays in the recognition process. Experiments carried out on two publically available word databases (RIMES for Latin script and IFN/ENIT for Arabic script) show the benefit of the proposed strategies.

Keywords: Handwriting recognition ; Dempster-Shafer theory ; ensemble classification ; rejection strategies ; representation of uncertainty ; imprecision.

Email addresses: yousri.kessentini@litislab.eu (Yousri Kessentini), thomas.burger@cea.fr (Thomas Burger), thierry.paquet@litislab.eu (Thierry Paquet)

1. Introduction

After about forty years of research in off-line handwriting recognition, the performances of current systems are still insufficient, as for many applications, more robust recognition is required. The use of Hidden Markov models (HMMs) in handwriting recognition systems has been widely studied during the last decade [1, 2, 3, 4, 5]. In fact, HMMs have a huge capacity to integrate contextual information and to absorb the variability. Furthermore, these models benefit from the experience accumulated in the domain of automatic speech recognition. Multiple HMM classifier combination is an interesting solution to overcome the limitations of individual classifiers [6, 7, 8, 9]. Various combination strategies have been proposed in the literature. They can be grouped into two broad categories: feature fusion methods and decision fusion techniques. The first category commonly known as early integration [10], consists in combining the input feature streams into a unique feature space, and subsequently use a traditional HMM classifier to model the combined observations in the unique input feature space. In contrast, decision fusion, known as late integration [11], consists in combining the single stream classifier outputs (decisions). A particular method within the decision fusion framework of sequence models falls into the multi-stream hidden Markov model paradigm. Such an approach has been successfully applied in [3] for handwritten word recognition. Beside, some research works stress the real interest of the Dempster-Shafer Theory (DST) [12, 13, 14, 15, 16, 17] to combine classifiers in a manner which is both accurate and robust to difficult conditions (set of weak classifiers, degenerated training phase, overly specific training sets, large vocabulary, etc.).

Generally, in the overall recognition process, high recognition rates is not the only measure to characterize the quality of a recognition system. For practical applications, it is also important to look at reliability. Reliability is related to the capability of a recognition system not to accept false word hypotheses and not to reject true word hypotheses. Rejection strategies are able to improve the reliability of handwriting recognition systems. Contrarily to classifier com-

bination, rejection strategies do not increase the recognition rate but, at least, reduce the number of errors and suggests an alternative treatment of the rejected samples [18, 19, 20, 21]. The rejection strategies are typically based on a confidence measure. If the confidence measure exceeds a specific threshold, the recognition result is accepted. Otherwise, it is rejected. Generally, this rejection may occur when 1) more than one word appears adequate; 2) no word appears adequate. As presented by Chow in [22], a pattern x is rejected if the word of maximal probability (among the possible words referred to as $\omega_i, i \in [1, N]$) conditionally to x is lower to some threshold:

$$\max_{i=1,\ldots,N} \mathbb{P}(\omega_i|x) < T \tag{1}$$

where $T \in [0, 1]$. On the other hand, the pattern x is accept and assigned to the class i, if $\max_{i=1,\ldots,N} \mathbb{P}(\omega_i|x) \geq T$. Fumera et al [23] point out that Chows rule provides the optimal error-reject trade-off, only if the posteriori probabilities are exactly known, which does not happen in real applications since they are affected by significant estimation errors. In order to overcome such a problem, Fumera et al. have proposed the use of multiple rejection thresholds for the different classes to obtain the optimal decision and reject regions, even if the a posteriori probabilities are affected by errors. It has been demonstrated that class-dependent rejection thresholds provide a better error reject trade-off than a single global threshold. In handwriting recognition field, many works have tested these two strategies. In [20], varieties of rejection thresholds including global, class-dependent and hypothesis-dependent thresholds are proposed to improve the reliability in recognizing unconstrained handwritten words. In [19], the authors present several confidence measures and a neural network to either accept or reject word hypothesis lists for the recognition of courtesy bank check amounts. In [24], a general methodology for detecting and reducing the errors in a handwriting recognition task is proposed. The methodology is based on confidence modeling and its main originality is the use of two parallel classifiers for error assessment. In [25], the authors propose multiple rejection thresholds to verify word hypotheses. To tune these rejection thresholds, an algorithm based

on dynamic programming is proposed. It focuses on maximizing the recognition rate for a given prefixed error rate. It was demonstrated in [26] that the class-dependent reject thresholds can be further improved if a proper search algorithm is used to find the thresholds. In [26], the authors use Particle Swarm Optimization (PSO) to determine class-related rejection thresholds. PSO is a population based stochastic optimization technique developed by Eberhart and Kennedy in 1995 [27]. It shares many similarities with evolutionary computation techniques such as genetic algorithms, but unlike genetic algorithms, PSO has no evolution operators such as crossover and mutation. In order to show the benefits of such an algorithm, the authors have applied it to optimize the thresholds of a cascading classifier system devoted to recognize handwritten digits.

In this article, we present a novel DST strategy to improve the performances and the reliability of a handwriting recognition system. Thus, the first contribution of this paper is to propose a DST combination method that can be applied for classification problems with large number of classes. Then, the second goal is to take advantage of the expressivity of DST to characterize the quality/reliability of the classification results. To do so, we compare different acceptation/rejection strategies for the classified words. In the end, an alternative treatment of the rejected samples is proposed using multi-stream HMM to improve the word recognition rate as well as the reliability of the recognition system, while not slow down the recognition process. The article is organized as follows: In Section 2, we recall the basis of a HMM classifier for handwriting recognition, and we present a background review on the basis of the Dempster-Shafer Theory. Section 3 describes the different steps of the DST-based ensemble classification method. Section 4 addresses in details the proposed post-processing module, where different acceptance/rejection strategies are presented. In Section 5, the overall system organization is presented, and each processing step is described. In section 6, we evaluate the performance of the proposed approaches. The conclusion and perspectives of this paper are presented in the last section.

2. Preliminaries on handwriting recognition and DST

In this work, we focus on an improvement of a multi-script handwriting recognition system using a HMM based classifiers combination. We combine the probabilistic outputs of three HMM classifiers, each working on different feature sets: upper contour, lower contour and density. A post-processing module based on different acceptance/rejection strategies, for reducing the error rate of the recognition system. In the end, an alternative treatment of the rejected samples is proposed using multi-stream HMM to improve the word recognition rate as well as the reliability of the overall recognition system. In the next subsection, we recall the basis of a HMM classifier for handwriting recognition, the multi-stream formalism, and we present a background review on the basis of the Dempster-Shafer Theory.

2.1. Markovian models for handwritten word recognition

One of the most popular technique for automatic handwriting recognition is to use generative classifiers based on Hidden Markov Models (or HMM) [28].

For each word ω_i of a lexicon $\Omega_{lex} = \{\omega_1, ..., \omega_V\}$ of V words, a HMM λ_i is defined. Embedded training is used where all character models are trained in parallel using Baum-Welch algorithm applied on word examples. The system builds a word HMM by concatenation of the character HMM corresponding to the word transcription of the training utterance, so that practically, its training phase is conducted by using the Viterbi EM or the Baum-Welch algorithm.

In the recognition phase, feature vectors extracted from a word image ω^* are passed to a network of lexicon entries formed of V word HMM built by the concatenation of their character HMM. The character sequence providing the maximum likelihood identifies the recognized entry. The Viterbi decoding algorithm provides a likelihoods $\mathbb{P}(\omega^* = \omega_i|\lambda_i), \forall i \leq V$, and the ω^* is recognized as the word ω_j for which $\mathbb{P}(\omega^* = \omega_j|\lambda_j) \geq \mathbb{P}(\omega^* = \omega_i|\lambda_i), \forall i \leq V$. The overall recognition process is presented in Figure 1).

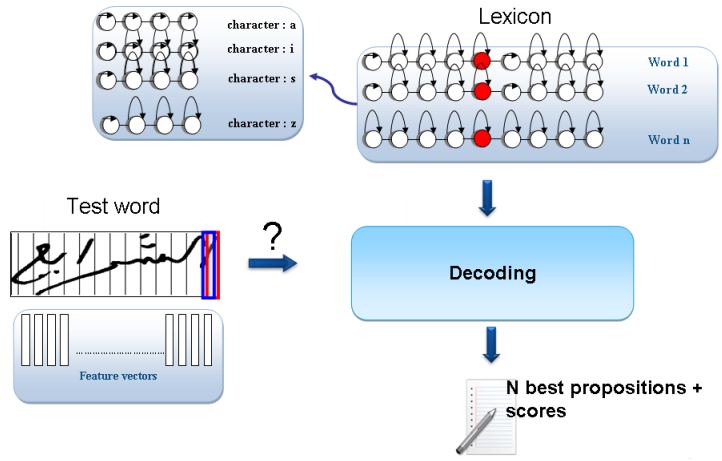

Figure 1: Handwritten word recognition scheme using HMM

2.2. Multi-stream HMM

The multi-stream formalism is an adaptive method to combine several individual feature streams using cooperative Markov models. This problem can be formulated as follows: assume an observation sequence X composed of K input streams X^k (with $\{k = 1, \ldots, K\}$) representing the utterance to be recognized, and assume that the hypothesized model M for an utterance is composed of J sub-unit models M_j (with $j = \{1, \ldots, J\}$) associated with the sub-unit level at which we want to perform the recombination of the input streams (e.g., characters). To process each stream independently of each other up to the defined sub-unit level, each sub-unit model M_j is composed of K models M_j^k (possibly with different topologies). Recombination of the K stream models M_j^k is forced at some temporal anchor states ($\otimes$ in Figure 2). The resulting statistical model is illustrated in Figure 2. Detailed discussion of the mathematical formalism is given in our previous work [3].

We have shown in [3] that the multi-stream framework improves the recognition performance compared to the mono-stream HMM and to the classical combination strategies. However, this improvement is extremely demanding

from a computational point of view, as complexity is a major concern of the multi-stream approach, specially when dealing with a large lexicon [29]. This is why, in this work, the multi-stream decoding is introduced after a first classification stage that allows to reduce the size of lexicon and decide whether a second classification stage is needed or not. Such a strategy does not slow done the recognition process.

In this work, we have 3 feature sets (streams), one is based on lower contour features, the second one is based on the upper contour features and the last one is based on density feature as described in section 5.2

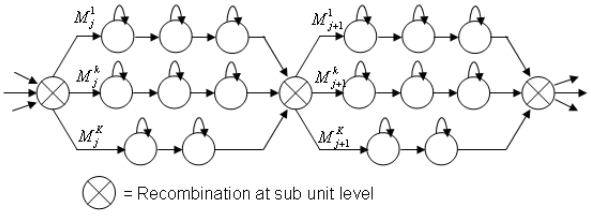

Figure 2: General form of K-stream model with anchor points between sub-units models

2.3. Basics of Dempster-Shafer theory

Let $\Omega = \{\omega_1, ..., \omega_V\}$ be a finite set, called the **frame**, or the **state-space**, made of exclusive and exhaustive classes (for instance, the words of a lexicon). A **mass function** m is defined on the powerset of Ω, noted $\mathcal{P}(\Omega)$ and it maps onto $[0, 1]$ so that $\sum_{A \subseteq \Omega} m(A) = 1$ and $m(\emptyset) = 0$. Then, a mass function is roughly a probability function defined on $\mathcal{P}(\Omega)$ rather than on Ω. Of course, it provides a richer description, as the support of the function is larger: If $|\Omega|$ is the cardinality of Ω, then $\mathcal{P}(\Omega)$ contains $2^{|\Omega|}$ elements.

It is possible to define several other functions which are equivalent to m by the use of sums or Möbius inversions. The belief function *bel* is defined by:

$$bel(A) = \sum_{B \subseteq A, B \neq \emptyset} m(B), \qquad \forall A \subseteq \Omega \tag{2}$$

$bel(A)$ corresponds to the probability of all the evidences that imply A. Dually, the plausibility function pl is defined by :

$$pl(A) = \sum_{B \cap A \neq \emptyset} m(B), \forall A \subseteq \Omega \tag{3}$$

It corresponds to a probabilistic upper bound (all the evidences that do not contradict A). Consequently, $pl(A) - bel(A)$ measures the **imprecision** associated to subset A of Ω.

A subset $F \subseteq \Omega$ such that $m(F) > 0$ is called a **focal element** of m. If the c focal elements of m are nested ($F_1 \subseteq F_2 \subseteq \ldots \subseteq F_c$), m is said to be **consonant**. If there is at least one focal element A of cardinality $|A| = k$ and no focal element of cardinality $> k$, then, the mass function is said to be k**-order additive**, or simply, k**-additive** [30].

Two mass functions m_1 and m_2, based on the evidences of two independent and reliable sources, can be combined into a new mass function by the use of the **conjunctive combination**, noted $\textcircled{\cap}$. It is defined $\forall A \subseteq \Omega$ as:

$$[m_1 \textcircled{\cap} m_2](A) = \frac{1}{1 - \mathcal{K}_{12}} \sum_{B \cap C = A} m_1(B) \cdot m_2(C) \tag{4}$$

where $\mathcal{K}_{12} = \sum_{B \cap C = \emptyset} m_1(B) \cdot m_2(C)$ measures the **conflict** between m_1 and m_2. $\mathcal{K}_{12}$ is called the **mass of conflict**.

The most classical way to convert a mass function onto a probability (for instance, to make a decision), is to use the pignistic transform [13]. Intuitively, it is based on the idea that the imprecision encoded in the mass function should be shared equally among the possible outcomes, as there is no reason to promote one of them rather than the others. If $|A|$ is the cardinality of the subset $A \subseteq \Omega$, the **pignistic probability** $\overline{m}$ of m is defined as:

$$\overline{m}(\omega_i) = \sum_{A \ni \omega_i} \frac{m(A)}{|A|} \qquad \forall \omega_i \in \Omega \tag{5}$$

Dually, it is possible to convert a probability distribution onto a mass function. The **inverse pignistic transform** [31] converts an initial probability

distribution p into a consonant mass function. The resulting consonant mass function, denoted by $\widehat{p}$, is built as follows: First, the elements of Ω are ranked by decreasing probabilities such that $p(\omega_1) \geq \ldots \geq p(\omega_{|\Omega|})$. Second, we define $\widehat{p}$ as:

$$\begin{aligned} \widehat{p}\left(\left\{\omega_1, \omega_2, \ldots, \omega_{|\Omega|}\right\}\right) = \widehat{p}\left(\Omega\right) &= |\Omega| \times p(\omega_{|\Omega|}) \\ \forall\ i < |\Omega|,\ \widehat{p}\left(\{\omega_1, \omega_2, \ldots, \omega_i\}\right) &= i \times [p(\omega_i) - p(\omega_{i+1})] \\ \widehat{p}\left(.\right) &= 0 \qquad \text{otherwise.} \end{aligned} \tag{6}$$

In this work, we refer to $\widehat{\widehat{m}}$ as the **pignistic discounting** of m, i.e. the application of the inverse pignistic transform to the pignistic probability derived from a mass m. The interest [31] of the pignistic discounting is that it associates to m the least specific (according to the commonality value) consonant mass function which would lead to the same decision as m.

3. Evidential combination strategy

In order to improve the recognition accuracy, it is possible to define several HMM classifiers, each working on different features. Here, we summarize our previous works to derive an efficient ensemble classification technique based on DST [32, 33]. Our aim is to combine the outputs of HMM classifiers in the best way. To do so, we have to (1) build the frame, (2) convert the probabilistic output of each of our Q classifiers into a mass function, (3) compute the conjunctive combination of the Q mass functions, and (4) design a decision function by using the pignistic transform.

3.1. Building dynamic frames

In handwritten word recognition, the set of classes is very large compared to the cardinality of the state space in classical DST problems (up to $100,000$ words). When dealing with a lexicon set of V words, the mass functions involved are defined on 2^V values. Moreover, the conjunctive combination of two mass

functions involves up to 2^{2^V} multiplications and 2^V additions. Thus, the computational cost is exponential with respect to the size of the lexicon. Dealing with $100,000$ word lexicon is not directly tractable.

To remain efficient, even for large vocabularies, it is mandatory either to reduce the complexity, or to reduce the size of the lexicon involved. To do so, as noted in the previous section, consonant mass functions (with only V focal elements) may be considered. Moreover, it is also possible to dynamically reduce the size of the lexicon by eliminating all the word hypothesis which are obviously not adapted to the test image under consideration. This can be done using a two stage classification scheme where the first stage selects a restricted list made of the most likely word hypothesis. Hence, we consider only the few word hypothesis among which a mistake is possible because of the difficulty of discrimination. Consequently, instead of working on $\Omega_{lex} = \{\omega_1, ..., \omega_V\}$, we select dynamically another frame $\Omega_{\mathcal{W}}$, defined according to each particular test word $\mathcal{W}$ we aim at classifying. That is why we say that such a frame is dynamically defined.

This strategy is rather intuitive and simple. On the other hand, to our knowledge, no work has been published on a comparison of the different strategies which can be used to define such frames. We have presented in [32] a detailed description of the dynamic definition of the state-space. In this work, we consider the union of increasing Top N lists, until M words are common to these lists. This method has given the best performance in our previous work [32]. We recall here the main principles of this method. Let us consider Q classifiers. Each classifier q provides an ordered list $l^q = \{\omega_1^q, ..., \omega_N^q\}$ of the TOP N best classes. Here, the frame $\Omega_{\mathcal{W}}$ is made of the union of all the words of the output lists l^q, $\forall q < Q$. Obviously, $|\Omega_{\mathcal{W}}|$ depends on the lists: if the Q classifiers globally concur, their respective lists are similar and very few words belong to the union of the lists. On the contrary, if the Q classifiers mostly disagree, an important proportion of their N words are likely to be found in their union. Hence, we adjust the value of N to control the size of the powerset,

in practice a powerset size between 15 and 20 is used. The idea motivating this strategy is the following: if a single classifier fails and provides too bad a rank to the real class, the other classifiers will not balance the mistake when considering the intersection strategy. Then, the union may be preferable.

3.2. Converting log-likelihoods into mass functions

The conversion of the probabilistic outputs into mass functions rises two difficulties. First of all, in case of HMM classifiers, the "real" probabilities are not available as output: the probability propagation algorithm underlying HMM implies a very wide range of numerical values that leads to overflows. This is why, instead of a classical likelihood, a log-likelihood is used. Moreover, it is regularly re-scaled during the computation, so that, at the end, $\mathbb{R}$-values are given rather than $[0, 1]$-values.

The second problem is that, a mass function provides a richer description than a probability function. Thus, the conversion from a probability into a mass function requires additional information.

Finally, we have to convert a $\mathbb{R}$-valued set of V scores onto a mass function which is a richer description, as it is defined with 2^V distinct values. Amongst the various methods that have been tested to achieve this conversion [34], we have chosen the following procedure for each of the Q classifiers:

1. Convert the set of $L_q(\omega_i)$ onto a new subjective probability distribution p_q, where $L_q(\omega_i)$ design the likelihood of the q-th classifier for ω_i. Note that $p_q(\omega_i)$ is supposed to be a fair evaluation of $\mathbb{P}(\omega * | \lambda_i, q)$, in spite of that $\sum_i \mathbb{P}(\omega * | \lambda_i, q) \neq 1$, whereas $\sum_i p_q(\omega_i) = 1$.
2. Convert this subjective probability into a mass function by adding the constraints that (1) the mass function is consonant, (2) the pignistic transform of the mass function corresponds to the subjective probability p_q. Under these two assumptions, it is proved that the mass function is uniquely defined [35].

Practically, the conversion from the $\mathbb{R}$-valued scores $L_q(\omega_i),\ i \leq V$ to subjective probabilities $p_q(\omega_i)$ is achieved by applying the following sigmoid function

that maps $\mathbb{R}$ onto $[0, 1]$:

$$p_q(\omega_i) = \frac{1}{1 + e^{-\lambda(L_q(\omega_i) - \tilde{L_q})}} \quad \text{with} \quad \lambda = \frac{1}{\max_i |L_q(\omega_i) - \tilde{L}_q|} \tag{7}$$

where $\tilde{L}_q$ is the median of the $L_q(\omega_i), \forall q$. Then, the set of $p_q(\omega_i),\ i \leq V$ is re-scaled so that it sums up to 1. Finally, the mass functions m_q are defined using equation (6). Once built, the mass functions m_q are combined together into a new mass function $m_\cap$ using the conjunctive combination (equation (4)).

4. Decision making and rejection strategies

At this level, it would be most natural to directly use the pignistic transform to make a decision on $m_\cap$. However, we propose here to improve the reliability of the proposed recognition system, by the introduction of an acceptance/rejection stage of the words to classify. As the DST has a rich semantic interpretation, we propose two different strategies for this acceptance/rejection post-processing. The point is not necessarily to compare them with respect to their performances, but rather to chose the one which is the most adapted to the scenario, as each strategy does not reject or accept the words on the basis on the same assumptions. The two strategies are based on a measure of conflict and a measure of conviction respectively [36]. The first one aims at evaluating the extent to wish the classifiers concur or not. The second one aims at evaluating if the knowledge resulting from the combination of the classifier is precise enough or not. By applying a threshold on one of these measures, it is possible to tune the importance of the rejection.

Let us first introduce some additional notations: After applying the pignistic transform on $m_\cap$, one denotes by $\omega_{(i)}$ the word the pignistic probability of which is the ith greatest (in other words, $w_{(1)}$ corresponds to the decision made according to the pignistic transform).

4.1. The conflict-based strategy

The first measure aims at quantifying the conflict among the evidence that have led to the classification. Intuitively, a high measure of conflict is supposed

to correspond to a situation where it is sound to reject the item, as there is contradictory information, whereas, low measure of conflict indicates that the evidences concur, and that rejection should not be considered. Several measures are available to quantify the conflict between several sources (such as described in [37]), among which, the **mass of conflict** from the conjunctive combination. This latter is really interesting, but in this work, we have chosen another measure, which is highly correlated with the mass of conflict, while being both easier to tune and more meaningful. Let us note that recent axiomatic works on measuring the conflict between various sources in the framework of DST justify the use of the measure we use here [38, 39].

Let ω_* be an unknown word from the test set, and ω_1 the class that have been ranked first by the classification process (the output of which is the mass function $m_\cap$). We define $Flict$, the measure of conflict, as:

$$Flict(\omega_*) = 1 - pl_\cap(\{\omega_{(1)}\}) = bel_\cap(\{\Omega \setminus \omega_{(1)}\}) \tag{8}$$

It corresponds to the sum of the mass of the evidences which do not support the decision which has been made. This measure is really interesting, as it is easy to interpret, and as it takes its value in $[0, 1]$. On the other hand, if one wants to be really discriminative by rejecting a huge proportion of the test set, this measure is not adapted, as potentially too many test words may have a null measure of conflict.

Finally, all the words which have been accepted are classified according to the decision promoted by the pignistic transform given in equation (5).

4.2. The conviction-based strategy

For a dedicated word, the second measure aims at quantifying the conviction of the decision which has been made, i.e. whether at the end of the classification process, a class is clearly more likely than the other, or, on the contrary, whether the choice relies on a very weak preference of a class with respect to the others. Of course, we expect that a low measure of conviction corresponds to a situation where there is not enough evidence to make a strong choice (and thus, rejection

is an interesting option), and a high measure of conviction indicates that there is no room for hesitation, nor rejection. As with the measure of conflict, we do not detail the comparative study of several measures of conviction, and we focus on the chosen one. We define the measure of conviction as:

$$Viction(\omega_*) = \frac{1}{\sum_{A \subseteq \Omega} \widehat{\overline{pl_\cap}}(A) - \widehat{\overline{bel_\cap}}(A)} \tag{9}$$

i.e. the inverse of the sum over $\mathcal{P}(\Omega)$ of the measure of imprecision of the pignistic discounting $\widehat{\overline{m_\cap}}$ of $m_\cap$. Indeed, $Viction$ is a fair measure of conviction (lower values corresponding to strong impercision and thus to decision supported by a weak conviction), however, in case $\sum_{A \subseteq \Omega} \widehat{\overline{pl_\cap}}(A) - \widehat{\overline{bel_\cap}}(A) = 0$, it is undefined. Finally, this is why, we consider $\frac{1}{Viction}$:

$$Viction(\omega_*) = \sum_{A \subseteq \Omega} \widehat{\overline{pl_\cap}}(A) - \widehat{\overline{bel_\cap}}(A) \tag{10}$$

Unfortunately, it loses the semantics of a conviction (as the greatest values corresponds to the weakest decision support), yet, it does not changes its use for tuning and prevent any division by zero, and simplifies the implementation. Contrarily to *Flict*, $Viction$ can be tuned according to the whole rejection spectrum, however its tuning is more difficult, as its bounds depend on $|\Omega|$. As with the conflict-based strategy, all the words which have been accepted are classified according to the decision promoted by the pignistic transform given in equation (5).

Remark 1. *The main interest of* $Viction$ *is that it can be defined in a completely probabilistic context, without an ensemble classification based on DST. As a matter of fact,* $\overline{m_\cap}$ *corresponds to a probability distribution (such as the one provided by any probabilistic classifier). As a consequence, in a probabilistic case, the classifier provides a probability distribution* p*, and then, a consonant mass* $m_p = \widehat{p}$ *is derived by applying the inverse pignistic transform to* p*. If* pl_p *and* bel_p *are the plausibility and belief functions of* m_p*, we have:*

$$Viction(\omega_*) = \sum_{A \subseteq \Omega} pl_p(A) - bel_p(A) \tag{11}$$

and this measure does not require any DST-based classifier nor any DST-based ensemble classification to be used.

Example 1. *Let us illustate the computation of Viction on a small example: the frame is made of two possible options A and B. The output of the ensemble classification is either a mass function, the pignistic transform of which reads $BetP(A) = 0.6$ and $BetP(B) = 0.2$, or directly a probability distribution, which reads $\mathbb{P}(A) = 0.6$ and $\mathbb{P}(B) = 0.2$. If one applies the inverse pignistic transform to this distrubtion, one obtains $m(\{A\}) = 0.2$ and $m(\{A, B\}) = 0.8$, so that:*

- $pl(\{A\}) - bel(\{A\}) = 1 - 0.2 = 0.8$
- $pl(\{B\}) - bel(\{B\}) = 0.8 - 0 = 0.8$
- $pl(\{A, B\}) - bel(\{A, B\}) = 1 - 1 = 0$

So that finally, on this example, the Viction coefficient equals 1.6.

5. System description

The input of our system is a word image. In the first step, pre-processing is applied to the word image and three feature sets are extracted corresponding to lower contour features, upper contour features and density features. In the second step, we combine the outputs of HMM classifiers using the evidential combination approach as described in section 3. Another module decides if the word hypothesis is accepted or rejected[1]. Finally, if the word is accepted, a decision is made according to the pignistic transform. For rejected samples, an alternative processing is proposed using multi-stream HMM. As multi-stream HMM are more efficient, this improves the word recognition rate as well as the reliability of the recognition system. Moreover, as this alternative processing is conducted only for difficult words (those for which classification is difficult)

[1] In the experimental section, we compare the influence of the various rejection strategies (presented in the previous section) to select the best one to use in the final global system.

it does not cause significant delays in the recognition process (in spite of the computational burden of multi-stream HMM). The whole system is depicted on figure 3. In the following sections we will provide details of the preprocessing, feature extraction and training stages.

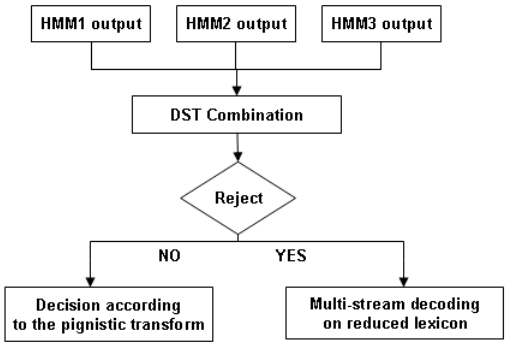

Figure 3: The global system description

5.1. Preprocessing

Preprocessing is applied to word images in order to eliminate noise and to get more reliable features less sensitive to noise and distortion.

- Normalization: In an ideal model of handwriting, a word is supposed to be written horizontally and with ascenders and descenders aligned along the vertical direction. In real data, such conditions are rarely respected. We use slant and slope correction so as to normalize the word image [40].
- Contour smoothing: Smoothing eliminates small blobs on the contour.
- Base line detection: Our approach uses the algorithm described in [2] based on the horizontal projection curve that is computed with respect to the horizontal pixel density (show Figure 4). Baseline position is used to extract baseline dependent features that emphasize the presence of descenders and ascenders.

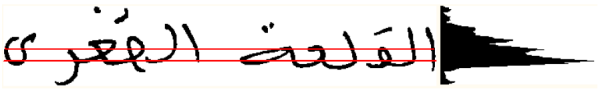

Figure 4: Base line detection

5.2. Features extraction

An important task in multi-stream combination is to identify features that carry complementary information. In order to build the feature vector sequence, the image is divided into vertical overlapping windows or frames. The sliding window is shifted along the word image from right to left and a feature vector is computed for each frame.

Two feature sets are proposed in this work. The first one is based on directional density features. This kind of features, initially proposed for latin script [40], has proved to be discriminative for arabic script [41]. The second one is based on foreground (black) pixel densities [4].

5.2.1. Densities features

The feature set inspired by [4], which has shown its efficiency in the 2009 ICDAR word recognition competition [3]. It is based on density and concavity features. From each frame 26 features are extracted for window of 8-pixel width (and 32 features for window of 14-pixel width). There are two types of features: features based on foreground (black) pixel densities, and features based on concavity. In order to compute some of these features (for example, f2 and f15 as described next) the window is divided into cells where the cell height is fixed (4 pixels in our experiments) as presented in Figure 5.

Let H be the height of the frame in an image, h be the fixed height of a cell, w the width of a frame (see figure 5). The number of cells in a frame n_c is equal to : $n_c = H/h$. Let $r_t(j)$ be the number of foreground pixels in the jth row of frame t, $n_t(i)$ the number of foreground pixels in cell i, and $b_t(i)$ the density level of cell i : $b_t(i) = 0$ if $n_t(i) = 0$ else $b_t(i) = 1$ Let LB the position of the lower baseline, UB the position of the upper baseline. For each frame t, the features are the following:

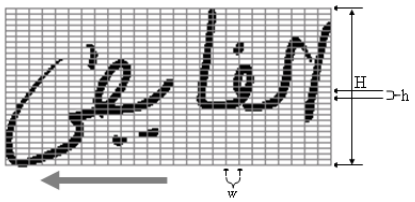

Figure 5: Word image divided into vertical frames and cells

- f_1 : density of foreground (black) pixels : $f_1 = \sum_{i=1}^{n_c} n_t(i)$.
- f_2 : number of transitions between two consecutive cells of different density levels : $f_2 = \sum_{i=2}^{n_c} |b_t(i) - b_t(i-1)|$.
- f_3 : difference in y position of gravity centers of foreground pixels in the current frame and in the previous one : $f_3 = g(t) - g(t-1)$ where $g(t) = \frac{\sum_{j=1}^{H} j.r_t(j)}{\sum_{j=1}^{H} r_t(j)}$.
- $f_4 - f_{11}$: densities of black pixels for each vertical column of pixels in each frame (note that the frames here are of 8-pixel width).

The next features depend of the base line position :

- f_{12} : vertical position of the center of gravity of the foreground pixels in the whole frame with respect to the lower baseline : $f_{12} = \frac{g(t)-LB}{H}$.
- $f_{13} - f_{14}$: density of foreground pixels over and under the lower baselines for each frame : $f_{13} = \frac{\sum_{j=LB+1}^{H} r_t(j)}{H.w}$, $f_{14} = \frac{\sum_{j=1}^{L-1} r_t(j)}{H.w}$
- f_{15} : number of transitions between two consecutive cells of different density levels above the lower baseline : $f_{15} = \sum_{i=k}^{n_c} |b_t(i) - b_t(i-1)|$ Where k is the cell that contains the lower baseline.
- f_{16} : zone to which the gravity center of black pixels belongs with respect to the upper and lower baselines (above upper baseline, a middle zone, and below lower baseline).

- $f_{17} - f_{26}$: five concavity features in each frame and another five concavity features in the core zone of a word, that is, the zone bounded by the upper and lower baselines. They are extracted by using a 3×3 grid as shown in Figure 6.

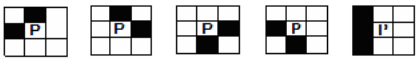

Figure 6: Five types of concavity configurations for a background pixel P

5.2.2. Contour features

These features are extracted from the word contour representation. Each word image is represented by its lower and upper contours (see Figure 7). A sliding window is shifted along the word image, two parameters characterize a window: window width (8 pixels) and window overlap between two successive positions (5 pixels). For each position of a window, we extract the upper contour points (similarly, the lower contour points). For every point in this window, we determine the corresponding Freeman direction and the directions points are accumulated in the directional histogram (8 features). In addition to the

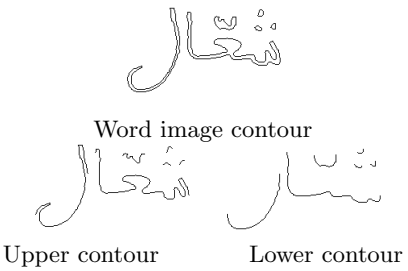

Figure 7: Word image contours

directional density features, a second feature set is computed at every point of the upper contour (similarly, it is done for every point on lower contour). The last (black) point (say, p^*) in the vertical black run started at an upper contour point (say, p) is considered and depending on the location of p^*, one of the four situations may arise. The point (p^*) can belong to a:

- Lower contour (see corresponding p points as marked red in Figure 8).
- Interior contour on closure (see blue points in Figure 8).
- Upper contour (see yellow points in Figure 8).
- No point found (see green points in Figure 8).

The black points in Figure 8 represent the lower contour.

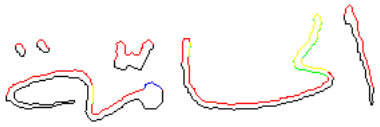

Figure 8: Contour feature extraction

The histogram of the four kinds of points is computed in each window. This second feature set provides additional information about structure of the contour like the loops, the turning points, the simple lines, and the end points on the word image (altogether, four different features).

The third feature set indicates the position of the upper contour (similarly, lower contour) points in the window. For this purpose, we localize the core zone of the word image. More precisely, we extract the lower and upper baselines of word images. These baselines divide the image into 3 zones: 1) a middle zone, 2) the lower zone, 3) the upper zone. This feature set (3 features) provides additional information about the ascending and the descending characters, which are salient characteristics for recognition of arabic script. Hence, in each window we generate a 15-dimensional (8 features from chain code, 4 features from the structure of the contour and 3 features from the position of the contour) contour (for upper or lower contour) based feature vector.

5.3. Character Models

In order to model the Latin characters, we have considered 72 models corresponding to lower case letters, capital letters, digits and accented letters. In the case of the Arabic characters, we built up to 159 character HMMs. An Arabic

character may actually have different shapes according to its position within the word (beginning, middle, end word position). Other models are specified with additional marks such as "shadda". Each character HMM is composed of 4 emitting states. The observation probabilities are modeled with Gaussian Mixtures (3 per state). Embedded training is used where all character models are trained in parallel using Baum-Welch algorithm applied on word examples. The system builds a word HMM by concatenation of the character HMM corresponding to the word transcription of the training sample.

6. Experiments and results

In this section, we evaluate the performances of the global system described above and we compare it to an equivalent technique in a probabilistic setting.

6.1. Datasets

Experiments have been conducted on two publicly available databases: IFN/ENIT benchmark database of arabic words and RIMES database for latin words. The IFN/ENIT [42] contains a total of 32,492 handwritten words (arabic symbols) of 946 Tunisian town/villages names written by 411 different writers. The sets a,b,c,d and e are predefined in the database for training and the set f for testing. In order to tune the HMM parameters, we performed a cross validation over sets a,b,c,d and e. The RIMES database [43] is composed of isolated handwritten word snippets extracted form handwritten letters (latin symbols). In our experiments, 36000 snippets of words are used to train the different HMM classifiers, 6000 word images are used for validation and 3000 for the test. At the recognition step, we use predefined lexicons composed of 2100 words in the case of IFN/ENIT database and 1600 words in the case of RIMES database.

6.2. Combination step

Table 1 provides the performances of each of the three HMM classifiers. In this table, not only the "best" class is given, but an ordered list of the TOP N best classes is considered. Then, for each value of $n \leq N$, a recognition

Table 1: Individual performances of the HMM classifiers.

	IFN/ENIT		RIMES	
	Top 1	**Top 2**	**Top 1**	**Top 2**
Upper contour	73.60	79.77	54.10	66.40
Lower contour	65.90	74.03	38.93	51.57
Density	72.97	79.73	53.23	65.83

rate is computed as the percentage of words for which the ground truth class is proposed in the first n elements of the TOP N list. We note that the reported results in table 1 are given without rejecting any sample.

It clearly shows that the two data sets are of heterogeneous difficulty. Moreover, the lower contour is always the less informative feature, and in the case of the RIMES database, it is really not informative. In Table 2, we present the performance of the combination of these HMM classifiers. We use the DST-based combination classifier presented in the previous sections and we compare it to the sum, the product and the Borda count rules.

Table 2: Accuracy rates of the various strategies on the two datasets.

	IFN/ENIT		RIMES	
	Top 1	**Top 2**	**Top 1**	**Top 2**
Product	80.07	83.23	64.80	73.10
Borda count	79.43	83.20	63.47	74.13
Sum	78.47	82.87	63.03	70.63
Proposed approach	**82.00**	**86.53**	**68.30**	**79.80**

we notice that the proposed combination approach improves the performance obtained with any of the single stream HMM. The gain is 8.4% on the IFN/ENIT

database and 14.2% on the RIMES database compared to the best single stream recognition rate. In addition, we notice that our evidential approach performs better than the product approach (which appears to be the best non evidential combination method) on the two databases with a gain of 1.93% on the IFN/ENIT database and 3.5% on the RIMES database.

Thus, the next point is to check whether the pairwise differences in the accuracy rates are significant or not. As addressed in [44], McNemars test can be used for determining whether one learning algorithm is better than another. If a difference is significant, it means that the first method is clearly better than the second one. On the contrary, if the difference is not statistically significant, then, the difference of performance is too small to decide the superiority of one method over another (as the results would be slightly different with other training/testing sets).

We first calculate the contingency table assuming there are two algorithms I and II, illustrated in Table 3, where :

- n_{00} is number of samples misclassified by both algorithms.
- n_{01} number of samples misclassified by algorithm I but not II.
- n_{10} number of samples misclassified by algorithm II but not I.
- n_{11} are correctly classified by both algorithms.

In our case, the null hypothesis assumes that the performance of two different strategies is the same. In Tables 4 and 5, we consider all the pairwise comparisons between two methods, and for each, we compute the p-value, i.e. the probability that the null hypothesis is true. The smaller the p-value, the more the difference of accuracy is likely to be significant. We may reject the null hypothesis if the p-value is less than 0.05.

The proposed approach is significantly different from the other combination methods on the two databases.

n_{00}	n_{01}
n_{10}	n_{11}

Table 3: 2×2 CONTINGENCY TABLE

Table 4: The p-values of McNemar's test for all the pairwise comparisons on the RIMES dataset. NA: not a number.

	S1	**S2**	**S3**	**S4**
S1: Proposed approach	NA	0.0041	1.22×10^{-9}	7.17×10^{-5}
S2: Product		NA	3.36×10^{-16}	2.2×10^{-6}
S3: Borda Count			NA	0.01948
S4: Sum				NA

Table 5: The p-values of McNemar's test for all the pairwise comparisons on the IFN/ENIT dataset.

	S1	**S2**	**S3**	**S4**
S1: Proposed approach	NA	1.89×10^{-11}	0.08283	0.00248
S2: Product		NA	5.76×10^{-13}	3.81×10^{-14}
S3: Borda Count			NA	0.12
S4: Sum				NA

6.3. Acceptance/rejection strategies

For comparison purpose with the rejection policies proposed in the literature, we have chosen the one proposed in [20] which provides the best result. It is sound to choose this strategy, as it shares the same philosophy as ours: it is based on the comparison of a simple measure computed for each test word to a fixed threshold, and it does not require extra classification process. Let ω_* be an unknown word from the test set, it is based on the following measure:

$$Diff(\omega_*) = \frac{\overline{m_\cap}(\omega_{(1)})}{\overline{m_\cap}(\omega_{(1)}) - \overline{m_\cap}(\omega_{(2)})} \tag{12}$$

where $\omega_{(1)}$ is the best word hypothesis and $\omega_{(2)}$ is the second best word hypothesis. The $Diff$ measure varies within $[0, 1]$. Thus, a threshold in $[0, 1]$ is selected on the validation set according to the expected Rejection Rate, and words with a $Diff$ measure greater than the threshold are rejected.

The acceptance/rejection strategies described in Section 4.1 and 4.2 have been applied to both databases. The considered measure is compared to a threshold, which has been determined on a validation set, in order to reach a particular Rejection Rate. Depending on the sign of the difference between the measure and the threshold, the test word is classified or rejected. Of course, our two motivations for the rejection (too much conflict or too few conviction) are supposed to be independent. In practice, as the classifiers are not completely independent, and as the scores provided by the classifiers are normalized (so that they add up to one whatever the conflict and the conviction), the conviction and conflict measures appeared as rather correlated in the preliminary tests. Hence, it makes sense to combine them, to stabilize the rejection performances. As advised in [36], we do so by simply rejecting a word if at least one of the two measures is beyond the threshold corresponding to the chosen Rejection Rate.

Rejection performance is evaluated using the Receiver Operating Characteristic (ROC) curve, which is a graphical representation of the trade-off between the True Rejection Rate (TRR) and the False Rejection Rate (FRR). The TRR (resp. FRR) is defined as the number of miss (resp. well) recognized words that

are rejected divided by the number of well (resp. miss) recognized words. Since we have a N-class problem, these rates are calculated as follows:

Let us consider a testing set of N_{test} words. We have:

$$\begin{aligned} N_{test} &= \overbrace{N_{rec} + N_{err}}^{N_{proc}} + \underbrace{N_{rejhit} + N_{rejmiss}}_{N_{rej}} \\ &= N_{hit} + N_{mis} \end{aligned}$$

where N_{rec} is the number of correctly classified words, N_{err} is the number of incorrectly classified words, and N_{rej} is the number of words which are not classified, as they have been rejected. The latter are divided into N_{rejhit}, the number of words that would have been correctly classified if not rejected, and $N_{rejmiss}$, the number of words that would have been misclassified if processed. Finally, N_{proc} is the number of words which have been processed (i.e. not rejected), and N_{hit} and N_{mis} corresponds to the number of words that would have been respectively correctly and incorrectly classified in case of absence of rejection strategies. Then, the following rates are classically defined:

$$\begin{aligned} \text{Recognition Rate} &= \frac{N_{rec}}{N_{test}} \\ \text{Error Rate} &= \frac{N_{err}}{N_{test}} \\ \text{Rejection Rate} &= \frac{N_{rej}}{N_{test}} = \frac{N_{rej}}{N_{rej} + N_{proc}} \\ \text{Reliability} &= \frac{N_{rec}}{N_{proc}} = \frac{\text{Recognition Rate}}{1 - \text{Rejection Rate}} \\ \text{True Rejection Rate} &= \frac{N_{rejmis}}{N_{mis}} \\ \text{False Rejection Rate} &= \frac{N_{rejhit}}{N_{hit}} \end{aligned}$$

The ROC curves, as well as the Error Rate, the Recognition Rate and Reliability with respect to the Rejection Rate are represented on Fig. 9. On the RIMES dataset, results are slightly better than with the reference strategy described above. Indeed, the value of the Area Under Curve (AUC)is 75.95% with the reference strategy, whereas it is 79.01% with ours. On the other hand, results on the IFN/ENIT dataset are by far better using our rejection strategy. In

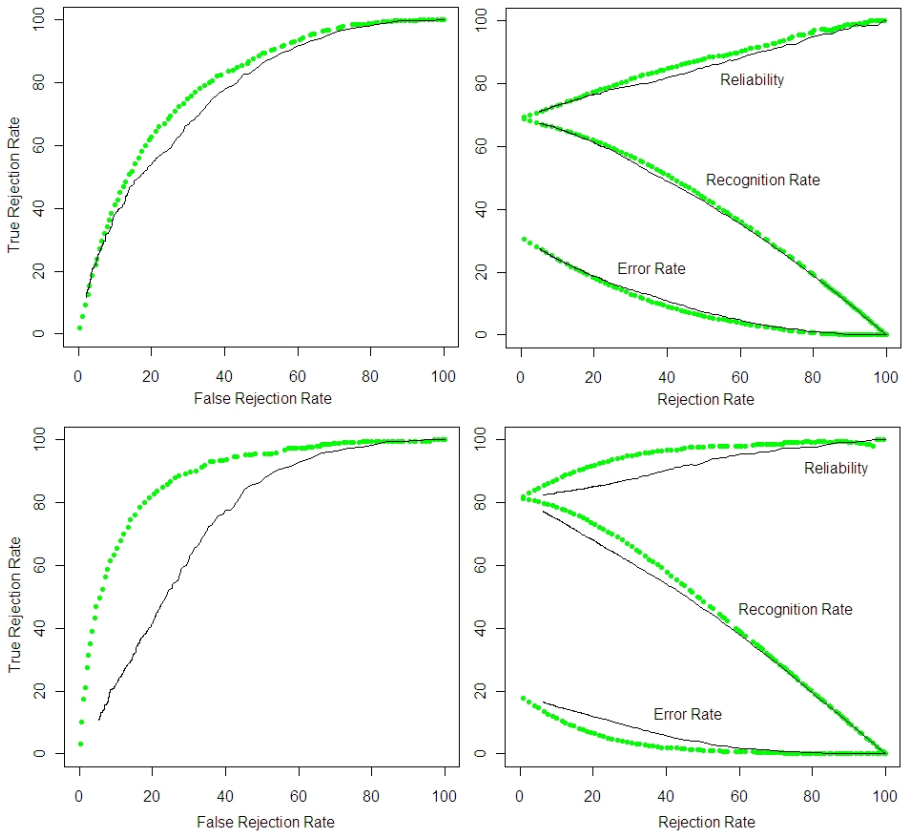

Figure 9: Comparison of the presented (dotted) and the reference (lined) methods for the RIMES (above) and IFN/ENIT (below) datasets. On the left, the ROC curve; on the right, the reliability, error and recognition rates.

fact, the AUC value is 72.79% with the reference strategy, whereas it is 88.05% with ours.

Moreover, we observe from this figure that for low Rejection Rates, the proposed rejection strategy produces interesting trade-offs between error and reject, which is the more important point in practical applications. Practically, the word Error Rates can be reduced from 18% to 6.37% on IFN/ENIT dataset and from 30.47% to 17.77% on RIMES at the cost of rejecting 20% of the input words.

Finally, these first series of experiments lead us to a use a logical OR on the thresholding induced by the Viction and the Flict strategies (displayed on Fig.10), and from that point on, the whole system is evaluated in this setting.

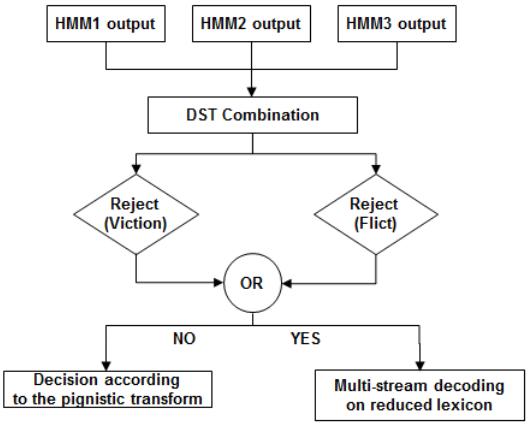

Figure 10: The global system description, refined according to the specification of the rejection module.

6.4. Treatment of the rejected samples

In the next evaluations, we apply to the words rejected by the selected acceptance/rejection strategy, a second classification level using the multi-stream HMM as described in [3]. To overcome the high complexity of the multi-stream decoding step, we use a small lexicon, so that, the delay introduced in the overall recognition process is almost negligible. The multi-stream HMM is tested using a lexicon composed of the 15 best word hypothesis that were also used to define the dynamic frame (Section 3.1). The rejection rate is tuned to 20% of the test set.

In table 6 we present the obtained results of the global system including the multi-stream HMM decoding for the rejected samples. The obtained results are compared to those of the system prior to any acceptance/rejection strategies (see Section 6.2), and they show that the second classification step based on the multi-stream HMM improves the performance of the global system in terms of

recognition rate while not increasing the recognition time. When compared to the reference system, the gain is 5.55% on IFN/ENIT database and 6.75% on RIMES database using the acceptance/rejection strategy.

In addition, we have used McNemar to determine if the two classification methods have significantly different recognition rates. The obtained p-value is equal to 2.22×10^{-5} which confirm that post-treatments of the rejected samples improve significantly the classification results.

Table 6: Accuracy rates of the various strategies

	IFN/ENIT		**RIMES**	
	Top 1	**Top 2**	**Top 1**	**Top 2**
Simple DST combination	82.00	86.53	68.30	79.80
Complete system	87.55	91.07	75.05	83.25

In order to compare our results to the most recent works presented in the literature, we report on table 7 the obtained results at the last international competition in Arabic handwriting recognition systems at ICDAR 2011 [45]. During this competition, 4 different handwriting recognition systems have been tested using IFN/ENIT database. We compare also our result to those of ICDAR 2009, ICFHR 2010 and ICDAR 2011 competitions. We notice that our system provides promising results, as it ranks in TOP 3 for all competitions, which is remarkable as our system does not contain any specific preprocessing adapted for the Arabic script (as it is initially proposed for the recognition of multi-script handwriting).

Table 7: Competition results comparison

System ID	Performance
ICDAR 2011 results [45]	
JU-OCR	63.86
CENPARMI-OCR	40.00
RWTH-OCR	92.20
REGIM	79.03
Results of the 3 best systems at ICFHR 2010 [46]	
UPV PRHLT	92.20
CUBS-AMA	80.32
RWTH-OCR	90.94
Results of the 3 best systems at ICDAR 2009 [47]	
MDLSTM	93.37
A2iA	89.42
RWTH-OCR	85.69
Proposed system	87.55

7. Conclusion

In this article, we have presented novel DST strategies to improve the performance and the reliability of a handwriting recognition system. The first contribution is the combination classifier based on Dempster-Shafer theory, which combines the outputs of several HMM classifiers. This combination classifier is interesting as (1) it can easily be generalized to other classifiers, as long as they provide a probabilistic output, (2) it improves the results with respect to classical probabilistic combination of HMM classifiers, (3) the complexity is kept under control in spite of the use of the DST, which is known for its computation cost (due to the manipulation of the power set). The second contribution is to propose a post-processing module based on different acceptance/rejection strategies, for reducing the Error Rate and improving the Reliability of the off-line handwritten word recognition system. The experimental results have shown through two different publicly available datasets (one with Latin script and the other with Arabic script) that the proposed system show significant improvement using DST strategies.

References

[1] S. Gunter, H. Bunke, Hmm-based handwritten word recognition: on the optimization of the number of states, training iterations and gaussian components, Pattern Recognition 37 (10) (2004) 2069 – 2079.

[2] A. Vinciarelli, S. Bengio, Writer adaptation techniques in hmm based off-line cursive script recognition, Pattern Recognition Letters 23 (8) (2002) 905–916.

[3] Y. Kessentini, T. Paquet, A. B. Hamadou, Off-line handwritten word recognition using multi-stream hidden markov models, Pattern Recognition Letters 30 (1) (2010) 60–70.

[4] R. Al-Hajj, C. Mokbel, L. Likforman-Sulem, Combination of hmm-based

classifiers for the recognition of arabic handwritten words, Proc. Int. Conf. on Document Analysis and Recognition (2007) 959–963.

[5] T.-H. Su, T.-W. Zhang, D.-J. Guan, H.-J. Huang, Off-line recognition of realistic chinese handwriting using segmentation-free strategy, Pattern Recognition 42 (1) (2009) 167 – 182.

[6] L. I. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, Wiley-Interscience, 2004.

[7] L. Xu, A. Krzyzak, C. Suen, Methods of combining multiple classifiers and their applications to handwriting recognition, IEEE Trans. Syst., Man, Cybern. (3) (1992) 418–435.

[8] N. Arica, F. T. Yarman-Vural, An overview of character recognition focused on off-line handwriting, IEEE Trans. Systems, Man and Cybernetics, Part C: Applications and Reviews (2) (2001) 216–232.

[9] M. Liwicki, H. Bunke, Combining diverse on-line and off-line systems for handwritten text line recognition, Pattern Recognition 42 (12) (2009) 3254 – 3263.

[10] R. Bertolami, H. Bunke, Early feature stream integration versus decision level combination in a multiple classifier system for text line recognition., in: Proc. Int. Conf. on Pattern Recognition, 2006, pp. 845–848.

[11] L. Prevost, C. Michel-Sendis, A. Moises, L. Oudot, M. Milgram, Combining model-based and discriminative classifiers : application to handwritten character recognition, Proc. Int. Conf. on Document Analysis and Recognition 1 (2003) 31–35.

[12] G. Shafer, A Mathematical Theory of Evidence, Princeton University Press, 1976.

[13] P. Smets, The transferable belief model, Artif. Intell. 66 (2) (1994) 191–234.

[14] C.-L. Liu, Classifier combination based on confidence transformation, Pattern Recognition 38 (1) (2005) 11 – 28.

[15] B. Quost, M.-H. Masson, T. Denoeux, Classifier fusion in the dempster-shafer framework using optimized t-norm based combination rules, International Journal of Approximate Reasoning 52 (3) (2011) 353–374.

[16] Y. Bi, The impact of diversity on the accuracy of evidential classifier ensembles, Int. J. Approx. Reasoning 53 (4) (2012) 584–607.

[17] M. Fontani, T. Bianchi, A. De Rosa, A. Piva, M. Barni, A framework for decision fusion in image forensics based on dempster shafer theory of evidence, Information Forensics and Security, IEEE Transactions on 8 (4) (2013) 593–607.

[18] A. Brakensiek, J. Rottland, G. Rigoll, Confidence measures for an address reading system, Proc. Int. Conf. on Document Analysis and Recognition 1 (2003) 294–298.

[19] G. Nikolai, Optimizing error-reject trade off in recognition systems, in: Proc. Int. Conf. on Document Analysis and Recognition, IEEE Computer Society, Washington, DC, USA, 1997, pp. 1092–1096.

[20] A. L. Koerich, R. Sabourin, C. Y. Suen, Recognition and verification of unconstrained handwritten words, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (2005) 1509–1522.

[21] P. Zhang, T. D. Bui, C. Y. Suen, A novel cascade ensemble classifier system with a high recognition performance on handwritten digits, Pattern Recognition 40 (12) (2007) 3415 – 3429.

[22] C. Chow, On optimum recognition error and reject tradeoff, IEEE Transactions on Information Theory 16 (1) (1970) 41–46.

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

[24] J. Rodrguez, G. Snchez, J. Llads, Rejection strategies involving classifier combination for handwriting recognition, in: Proc. Int. Conf. on Document Analysis and Recognition, Vol. 4478 of Lecture Notes in Computer Science, 2007, pp. 97–104.

[25] L. Guichard, A. H. Toselli, B. Couasnon, Handwritten word verification by svm-based hypotheses re-scoring and multiple thresholds rejection, Proc. Int. Conf. on Frontiers in Handwriting Recognition (2010) 57–62.

[26] L. Oliveira, A. Britto, R. Sabourin, Optimizing class-related thresholds with particle swarm optimization, in: Proc. Int.l Joint Conf. on Neural Networks, Vol. 3, 2005, pp. 1511–1516 vol. 3.

[27] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proc. Int. Joint Conf. on Neural Networks, Vol. 4, 1995, pp. 1942–1948 vol.4.

[28] L. R. Rabiner, A tutorial on hidden markov models and selected applications in speech recognition, in: Proceedings of the IEEE, 1989, pp. 257–286.

[29] Y. Kessentini, T. Paquet, A. Guermazi, An optimized multi-stream decoding algorithm for handwritten word recognition, in: Proc. Int. Conf. on Document Analysis and Recognition, IEEE, 2011, pp. 192–196.

[30] M. Grabisch, k-order additive discrete fuzzy measures and their representation, Fuzzy sets and systems 92 (2) (1997) 167–189.

[31] D. Dubois, H. Prade, P. Smets, New semantics for quantitative possibility theory, in: Proc. of the 6th European Conf. on Symbolic and Quantitative Approaches to Reasoning and Uncertainty, 2001, pp. 410–421.

[32] Y. Kessentini, T. Burger, T. Paquet, Constructing dynamic frames of discernment in cases of large number of classes, Proc 11th European Conf. on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (2011) 275–286.

[33] Y. Kessentini, T. Burger, T. Paquet, Evidential ensemble hmm classifier for handwriting recognition, in: Proc. Int. Conf. on Uncertainty Processing and Management, Vol. 6178, 2010, pp. 445–454.

[34] Y. Kessentini, T. Paquet, T. Burger, Comparaison des méthodes probabilistes et évidentielles de fusion de classifieurs pour la reconnaissance de mots manuscrits, in: CIFED, 2010.

[35] T. Burger, O. Aran, A. Urankar, L. Akarun, A. Caplier, A dempster-shafer theory based combination of classifiers for hand gesture recognition, Computer Vision and Computer Graphics - Theory and Applications, Lecture Notes in Communications in Computer and Information Science 21 (2008) 137–150.

[36] T. Burger, Y. Kessentini, T. Paquet, Dempster-shafer based rejection strategy for handwritten word recognition, in: Proc. Int. Conf. on Document Analysis and Recognition, 2011, pp. 528 – 532.

[37] W. Liu, Analyzing the degree of conflict among belief functions, Artificial Intelligence 170 (11) (2006) 909–924.

[38] S. Destercke, T. Burger, Revisiting the notion of conflicting belief functions, in: T. Denoeux, M.-H. Masson (Eds.), Belief Functions: Theory and Applications, Vol. 164 of Advances in Intelligent and Soft Computing, Springer Berlin Heidelberg, 2012, pp. 153–160.

[39] S. Destercke, T. Burger, Toward an axiomatic definition of conflict between belief functions, IEEE Trans Syst Man Cybern B 43 (2) (2013) 585–596.

[40] F. Kimura, S. Tsuruoka, Y. Miyake, M. Shridhar, A lexicon directed algorithm for recognition of unconstrained handwritten words, IEICE Trans. on Information & Syst. E77-D (7) (1994) 785–793.

[41] Y. Kessentini, T. Paquet, A. BenHamadou, A multi-lingual recognition system for arabic and latin handwriting, in: Proc. Int. Conf. on Document Analysis and Recognition, 2009, pp. 1196–1200.

[42] M. Pechwitz, S. Maddouri, V. Maergner, N. Ellouze, H. Amiri, Ifn/enit - database of handwritten arabic words, Colloque International Francophone sur l'Ecrit et le Doucement (2002) 129–136.

[43] E. Grosicki, M. Carre, J. Brodin, E. Geoffrois, Results of the rimes evaluation campaign for handwritten mail processing, Proc. Int. Conf. on Document Analysis and Recognition 0 (2009) 941–945.

[44] T. G. Dietterich, Approximate statistical tests for comparing supervised classification learning algorithms, Neural Comput. 10 (7) (1998) 1895–1923.

[45] V. Margner, H. Abed, Icdar 2011 - arabic handwriting recognition competition, in: Proc. Int. Conf. on Document Analysis and Recognition, 2011, pp. 1444–1448.

[46] V. Margner, H. Abed, Icfhr 2010 - arabic handwriting recognition competition, in: Proc. Int. Conf. on Frontiers in Handwriting Recognition, 2010, pp. 709–714.

[47] H. El Abed, V. Mrgner, Icdar 2009-arabic handwriting recognition competition, International Journal on Document Analysis and Recognition 14 (1) (2011) 3–13.