HAL
open science

Using the Web to create dynamic dictionaries in handwritten out-of-vocabulary word recognition

Cristina Oprean, Laurence Likforman-Sulem, Adrian Popescu, Chafic Mokbel

To cite this version:

Cristina Oprean, Laurence Likforman-Sulem, Adrian Popescu, Chafic Mokbel. Using the Web to create dynamic dictionaries in handwritten out-of-vocabulary word recognition. 12th International Conference on Document Analysis and Recognition (ICDAR), 2013, Aug 2013, Washington DC, United States. pp.989 - 993. hal-00948980

HAL Id: hal-00948980

https://hal.science/hal-00948980v1

Submitted on 18 Feb 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Using the Web to create dynamic dictionaries in handwritten out-of-vocabulary word recognition

Cristina Oprean
Institut Mines-Telecom/
Telecom ParisTech and
CNRS LTCI, 46 rue Barrault,
75013 Paris (France)
oprean@telecom-paristech.fr

Laurence Likforman-Sulem
Institut Mines-Telecom/
Telecom ParisTech and
CNRS LTCI, 46 rue Barrault,
75013 Paris (France)
likforman@telecom-paristech.fr

Adrian Popescu
CEA, LIST,
LVIC,
91190 Gif-sur-Yvette, France
adrian.popescu@cea.fr

Chafic Mokbel
University of Balamand,
Faculty of Engineering,
P.O. Box 100 Tripoli (Liban)
chafic.mokbel@balamand.edu.lb

***Abstract*—Handwriting recognition systems rely on predefined dictionaries obtained from training data. Small and static dictionaries are usually exploited to obtain high in-vocabulary (IV) accuracy at the expense of coverage. Thus the recognition of out-of-vocabulary (OOV) words cannot be handled efficiently. To improve OOV recognition while keeping IV dictionaries small, we introduce a multi-step approach that exploits Web resources. After an initial IV-OOV sequence classification, external resources are used to create OOV sequence-adapted dynamic dictionaries. A final Viterbi-based decoding is performed over the dynamic dictionary to determine the most probable word for the OOV sequence. We validate our approach with experiments conducted on RIMES, a publicly available database. Results show that improvements are obtained compared to standard handwriting recognition, performed with a static dictionary. Both domain-adapted and generic dynamic dictionaries are studied and we show that domain adaptation is beneficial.**

I. Introduction

Speech and handwriting recognition systems heavily rely on linguistic resources, such as dictionaries (word lists) or language models (n-grams for instance). When dictionaries are used, the performance of recognition systems is tributary to an appropriate choice of dictionary size. Small dictionaries generate poor results due to the fact that only few words are modeled. For large dictionaries, a large number of confusions will equally lead to low accuracy. Moreover, the recognition complexity increases with the dictionary size and the use of large dictionaries is cumbersome in real world applications.

Recognition systems are often implemented using Hidden Markov Models (HMMs) which match observed sequences to dictionary words. Word-based HMM modeling is performed by concatenating the sequences' compound HMM character models. In this approach, a close world assumption can be followed to assign all tested sequences to dictionary words. Alternately, a part of the tested sequences are classified as OOV words which are fed into an ergodic character-based HMM, called filler model. The performances of filler models are generally poor and methods for improving OOV recognition are needed.

In this paper, we introduce a handwritten recognition approach that makes use of the Web to improve the recognition of OOV words. Figure 1 provides an overview of the approach. Each word is decoded with word-based and filler models and is classified as IV or OOV based on the difference of the log-likelihood. At this point, sequences classified as IV are assigned to one of the words from the static dictionary. Sequences classified as OOV are compared to Web resources (Google or Wikipedia) in order to create dynamic dictionaries. Finally, a second Viterbi is run to retrieve the element of the dynamic dictionary that is most similar to the tested sequence. Thus a good compromise between dictionary size and method coverage is obtained.

The Web is the largest repository one can exploit to produce language resources, including dictionaries for OOV recognition in speech processing or spell checking [1]. The main differences between our work and [1] come from the much higher difficulty of handwritten OOV recognition compared to misspelling detection and from the innovative way devised to create dynamic dictionaries from Wikipedia. In [2] and [3] local context is exploited to retrieve Web documents that are used to augment the lexicon. Our system is word oriented and we do not exploit the local context. In addition, we propose a new adaptation method to retain only the part of an external corpus that is most similar to our training dataset.

The paper is organized as follows. Section II presents our context-independent HMM recognition system. In Section III, the classification of IV and OOV words is presented. The creation of dynamic dictionaries from the Web is described in Section IV. Finally, Section V is devoted to word recognition experiments on the Rimes publicly available database.

II. System overview

We perform experiments with a context-independent HMM-based, segmentation-free system that uses a sliding window approach [4], [5]. The input of the system is a sequence of windows (frames), divided into a fixed number of subsets of pixels (cells). A set of 37 features is extracted for each frame, augmented with their first order derivatives. The extracted features include: *2 features* representing foreground/background transitions; *12 features* for concavity configurations; *3 features* for the gravity center position - the first feature gives the position w.r.t. the baselines, the second one is the distance in number of pixels to the lower baseline, and the last one represents the difference between the gravity centers of two neighboring windows; $w = 9$ features corresponding to the density of pixels in each column (w is the width of the window); *3 features* for the density of pixels within the sliding window, above and below the baselines and *8 directional*

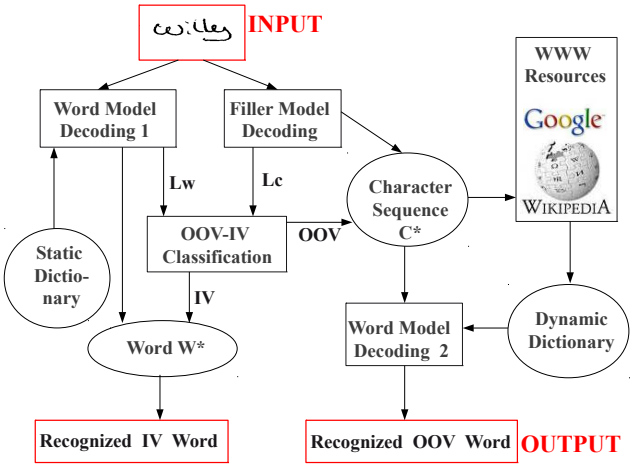

Fig. 1. Overview of the proposed approach.

features corresponding to the histogram of gradients for the 8 orientations from 0 to 7π/4, with a π/4 step. Two consecutive sliding windows have a shift of $\delta = 3$ and each of them is divided in 20 cells. Each character is modeled by 12 states and special symbols such as (, -, /) are modeled by 4 states. Each state is defined by a Gaussian Mixture Model with 20 components. Since the system is context-independent, a character HMM model corresponds to a letter. Therefore a number of 79 models, corresponding to case sensitive French characters, was considered. The Baum-Welch algorithm is used for training and the Viterbi algorithm for sequence decoding.

III. OOV AND IV WORDS PROCESSING

The use of word models to recognize sequences of handwritten characters yields good quality results for IV words if a good compromise between coverage and discriminative power is found. For the remaining OOV words (i.e. low frequency words, named entities, codes) alternative methods are needed. An efficient classification of IV and OOV words is a prerequisite for obtaining high overall system performances. In this section we briefly present our log-likelihood approach for IV-OOV classification.

A. OOV/IV Word Modeling

There are many approaches in the literature to cope with OOV detection in speech or handwriting recognition. They are based on filler models, hybrid sub-words models [6], [7] or confidence scores [8], [9]. Handwriting recognition systems were developed to detect OOV words within sentences or texts [10], [11], [12] by combining filler models and confidence scores.

OOV words are represented by character networks composed of independent characters that are delimited by space models (fig. 2-a). IV words are represented using more sophisticated word networks built over a static predefined dictionary (fig. 2-b). The HTK Toolkit [13] is used to decode visual frame sequences extracted from word images. For a given sequence, the likelihood of the observed feature vector is computed by using Viterbi algorithm [14] (top 3 decodings are retained).

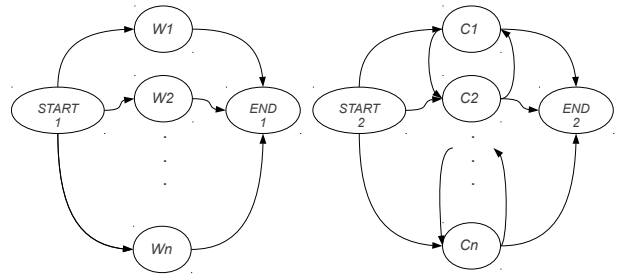

Fig. 2. a word network (a) a character network (b).

B. IV-OOV Word Classification

Sequences are classified as IV or OOV by using the log-likelihoods of the frame computed using with word and filler models. Let $X = x_1, x_2, ...x_N$ be the observed frame sequence, with N the number of frames. Two likelihood values, Lc and Lw, are obtained. $Lc = log\ P(X|C^*) = \max_C \log\ P(X|C)$ where C^* is the best character sequence for the frame sequence X. $Lw = log\ P(X|W^*) = \max_W \log\ P(X|W)$ where W^* is the best dictionary word for X.

The normalized difference between the two log-likelihoods, $diff(W^*, C^*)$ (1), is used to classify the decoded sequence as OOV or IV.

$$diff(W^*, C^*) = \frac{Lw - Lc}{N}. \quad (1)$$

If $diff(W^*, C^*) \geq thre$ the candidate sequence is assigned to IV, assuming that it belongs to the static dictionary. Otherwise it is considered an OOV word. The value of $thre$ is optimized using a subset of the validation dataset (Section V).

IV. DYNAMIC DICTIONARY CREATION FOR OOV WORD RECOGNITION

We hypothesize that, due to the poor performances of filler models, the creation of dynamic dictionaries that contain strings similar to a detected OOV sequence improves its recognition. When a sequence is classified as OOV, the output of the filler model is compared to an adapted dynamic dictionary in order to find the most probable word that corresponds to the sequence. For instance, when decoding *signalais*, *experiences*, *secteurs* the system returns *sinnxhsas*, *ciperierces* and *secterers*. To mitigate errors produced by character networks, we propose two dynamic dictionary creation methods that exploit external knowledge contained in the Google search index and in Wikipedia respectively. The two methods are similar in spirit since candidates are obtained using string similarities. The main differences arise from the size of the underlying corpus, much bigger for the Google-based method, the costly access to Google via API queries vs. immediate access to the Wikipedia resource and the number of suggestions - fixed by the Google API but easy to tailor for Wikipedia.

A. Google-based dynamic dictionary

Here the Google API is used to build dynamic dictionaries for OOV recognition. Once an OOV is decoded, the

sequence of characters obtained with the character network is run through the Google API which provides results, but also proposes corrected word spellings in some cases. The list of suggestions obtained for each sequence of characters associated to the OOV is roughly sorted by the number of times they appear in web texts [15]. This list is used as a dynamic dictionary that constitutes a new word network associated to the OOV sequence and is exploited by a second Viterbi decoding.

Google alternative spelling proposals dynamically reduce the candidate space by retaining strings that are similar to the initially decoded sequence. The main advantage of this solution is that it enables the processing of OOV words without the use of a static dictionary and, consequently, a generalization of the word recognition process is obtained. As we show in Section V the most important limitation of the method comes from the reduced number of alternative spellings proposed. Its utility is reduced for high confusions that are often produced by filler models. Additionally, the number of queries that a user can freely issue on the Google API is drastically limited (100 per day), generating a non-neglectable cost if Google would be used for large scale corpora.

B. Wikipedia-based dynamic dictionary

Wikipedia is a comprehensive encyclopedia that describes a large number of concepts and is thus fitted for creating dictionaries that provide a good language coverage. The dump of French Wikipedia from September 2012 was used in this paper. It included 410,482 articles that contain at least 100 distinct words. One of our objectives was to test the creation of a domain-adapted dictionary that was built using the RIMES training corpus. We then used the cosine similarity between the TF-IDF representation of the RIMES training corpus and the TF-IDF representation of each Wikipedia article to rank these articles based on their similarity to the RIMES domain. The domain and generic dictionaries were built by taking the 20,000 and respectively 200,000 most similar articles and then computing each word's document frequency (i.e. number of different documents in which the word appears) from these corpora. Domain dictionaries favor domain-related terms and are based on a smaller corpus. Generic dictionaries do not favor domain terms but are built using a larger corpus. We illustrate term selection with the word *facture* (bill). This word is highly characteristic of the RIMES collection and has documents frequencies of 1440 and 1459 in the domain and the generic dictionaries.

Levenshtein distance [16] is a classical measure of the difference between two sequences of characters. It computes the number of edits necessary for one sequence to turn into the other. The most similar words w.r.t. an entry sequence are grouped based on their Levenshtein distance. At equal distance, words are sorted using their document frequency. Only the k most frequent words for which the difference between their length and that of the decoded sequence is at most l are retained in the dynamic dictionary. k and l values are determined by statistical analysis of the number of correctly recognized words that are included in the dictionary and of the difference between word-sequence lengths. The authors of [17] report that 80% of the misspellings have an edit distance of one. We computed the Levenshtein distance

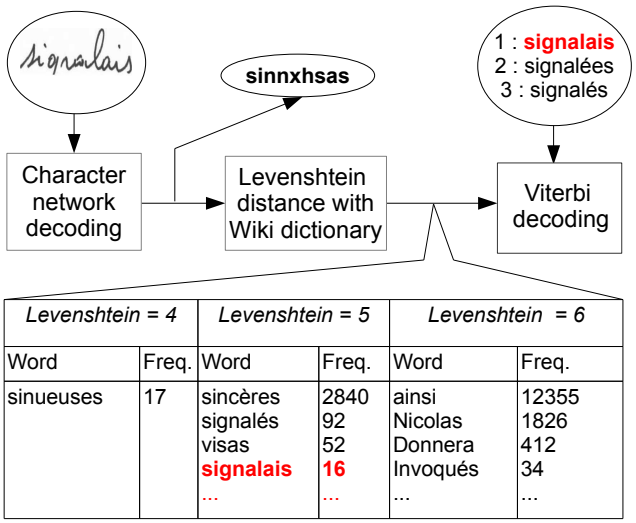

Fig. 3. Wikipedia-based OOV word recognition process.

between the OOV sequence and the real word of and it averages at 5.48. Moreover, the overall percentage of erroneous character recognitions is 61%. These results show that the problem tackled here is much more difficult than misspelling correction and it is necessary to retain a relatively high number of similar words in the dynamic dictionary. In the experiments, the dictionary size is k = 100, 200, 500 and l = 5.

The Levenshtein-based groups are used to create the word-adapted dictionaries. For instance, *signalais* was initially decoded as *sinnxhsas*. By computing the Levenshtein distance with the entire Wikipedia dictionaries, we obtain the groups represented in fig. 3. Note that the true word *signalais* is found at a Levenshtein distance 5. Even though the word does not have a high document frequency in Wikipedia and it does not have the minimum Levenshtein distance, the word can still be retrieved through the use of a second Viterbi decoding.

V. Experiments

To assess the effectiveness of dynamic dictionary creation presented in Section IV, we conduct experiments on the RIMES word database [18]. The metric used throughout all experiments is accuracy (i.e. the number of correctly recognized words divided by the test set size). RIMES was created by asking volunteers to write letters related to scenarios such as bank account information or modification, damage declarations or payment. Writers were asked to use black ink on white paper and to write freely. Documents were then scanned in gray level. From these documents, blocks, text-lines and words were extracted and labeled. Word and text-line recognition competitions have been organized using this database in 2009 and 2011 [18]. RIMES 2011 is divided into training, validation and test sets composed of 51,739, 7,464 and 7,776 word images respectively. Corresponding dictionaries include 4,279, 1,588 and 1,612 unique words respectively.

A. OOV detection

IV-OOV classification based on log-likelihood difference ($diff$) score was presented in Section III-B. The training set

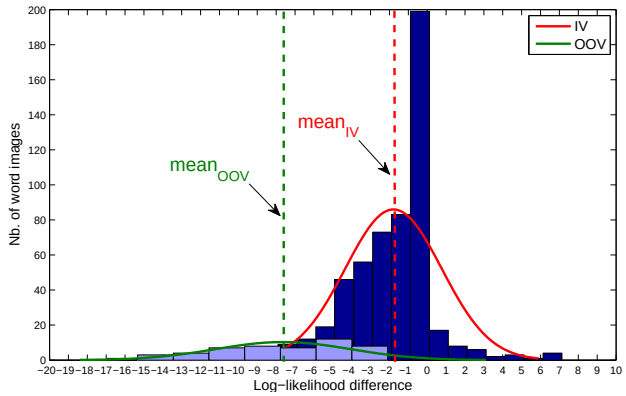

Fig. 4. Log-likelihood distributions for IV and OOV words

is used to learn models and the validation set to evaluate performances. Around 5% of validation words are not present in the training set and are OOVs. 378 OOVs are obtained after pruning special sequences (codes, dates, telephone numbers etc.) that are outside the scope of this paper. In order to re-use our system for completely new words, we determine a threshold $thre$ (Section III-B) that classifies words as OOV and IV on a subset of the validation dataset. To approach the overall distribution of words in the validation set, 50 OOVs and 550 IVs are used for this task and excluded from further experiments. Both word sublists are decoded by using the Viterbi algorithm and the character and word networks (Section III-A). The word network is built on the RIMES 2011 training dataset. The $diff$ score distributions for the selected OOV and IV words are shown in fig. 4. The distributions are centered on two distinct means $mean_{OOV}$ and $mean_{IV}$ but show a significant overlap. The threshold value $thre$ that separates OOV and IV words can be chosen between the two means.

A $thre$ value near $mean_{IV}$ minimizes the number of real OOV words assigned to the IV class (false negatives) but wrongly classifies a large number of IV as OOV. Inversely, with a $thre$ value close to $mean_{OOV}$, the number of IVs mistaken for OOVs is minimized while a larger proportion of OOVs are classified as IVs. In an initial experiment performed with 120 sequences (80 IV and 40 OOV), the global IV-OOV classification accuracies are 50% with $thre = mean_{IV}$ and 44.1% with $thre = mean_{OOV}$. We include both these thresholds in the larger scale word recognition experiments presented in the next section.

B. OOV Recognition

A preliminary experiment was devoted to the comparison of Google (already used for misspelling correction [1] and other tasks) and Wikipedia-based recognitions. The generic Wikipedia dictionary, with top 100 nearest Levenshtein neighbors was used. First, 40 OOV words were randomly chosen, out of which 1 and respectively 20 sequences were correctly recognized with Google and Wikipedia. While fitted for simpler tasks as misspelling correction during Web searches, the Google-based approach does not provide enough word variants to cope with the high confusions generated by the filler model. Due to this poor performance of Google, larger scale experiments were performed only with Wikipedia.

A second large-scale experiment was conducted on 6,864 words from the RIMES 2011 validation set (the 600 words used for tuning the system were left out during this experiment). To examine the behavior of the system in different conditions, we have considered $thre$ as $mean_{IV}$ and $mean_{OOV}$. When $thre$ = $mean_{IV}$ is used a number of 3,562 word images were identified as OOV, compared to 329 when $thre$ = $mean_{OOV}$. Words detected as IV will be decoded with the train vocabulary. The words classified as OOV (their $diff$ score $< thre$) are processed with the method described in Subsection IV-B. As we mentioned, Wikipedia domain and generic dictionaries are based on 20,000 and 200,000 articles respectively. Thresholds on document frequency were used (minimum 12 and 40 occurrences) and the two dictionaries contain 76,566 and 137,200 words respectively. Different values for the number of articles and document frequency thresholds were used in initial tests. These tests showed that the reported values ensure a good compromise between accuracy and computational complexity.

We compare the hybrid IV-OOV correction approach to a pure IV approach in which sequences are matched to words from the static dictionary. Domain and generic Wikipedia dictionaries are used to build dynamic word-adapted dictionaries and then to recognize words initially classified as OOV via a second Viterbi applied to the dynamic dictionary. In table I, results for $thre = mean_{IV}$ and $thre = mean_{OOV}$ for both Wikipedia dictionaries are presented with variable size of the dynamic dictionary ($k = 100, 200, 500$). The results

Dictionary type	**Domain**			**Generic**		
Levenshtein neighbors	100	200	500	100	200	500
$thre = mean_{OOV}$ [%]	46.03	47.25	**51.22**	41.15	41.76	**43.59**
$thre = mean_{IV}$ [%]	28.9	29.24	**29.58**	25.51	25.85	**26.22**

TABLE I. OOV RECOGNITION ACCURACY ON THE RIMES 2011 DATABASE.

in table I show that domain-adapted dictionaries outperform generic ones for all tested dynamic dictionary sizes. For a size of 500 words with $thre = mean_{OOV}$, the accuracy of the domain-adapted dictionary is 51.22%, against only 43.56% for the generic one. Equally important, the recognition accuracy increases with the size of the dynamic dictionaries. The best results are obtained when up to 500 similar words are retained for both types of dictionaries. This result shows that, given the poor quality of filler model-based OOV decoding (the average Levenshtein distance is 5.48), a large number of similar words are needed for efficient recognition. Beyond improving OOV

Recognition approach	**Accuracy** [%]
Pure IV	70.57
IV-OOV Filler	70.51
IV-OOV Levenshtein	71.05
IV-OOV Viterbi	**72.84**

TABLE II. OVERALL RECOGNITION ACCURACY ON THE RIMES 2011 DATABASE.

recognition, a more general aim is to propose approaches that combine word and character-based recognition. We present global accuracy results obtained in table II using $thre = mean_{OOV}$ and dynamic dictionary size $k = 500$. *Pure IV* stands for the exclusive use of word networks in which all sequences are assigned to a predefined dictionary. *IV-OOV*

Filler combines word models for IV detection (performed on a static dictionary that excludes sequences detected as OOV) and a raw filler model OOV decoding (2.4% of OOV correctly recognized). *IV-OOV Levenshtein* adds a direct assignment of OOV to the nearest neighbor from the dynamic dictionary to *IV-OOV Filler* (10.4% of OOV correctly recognized). *IV-OOV Viterbi* is the complete configuration presented in fig. 3 (51.22% of OOV correctly recognized). This last configuration has the best overall performance, with 2.27% improvement over *Pure IV*. The raw use of filler models in *IV-OOV Filler* has a very small negative impact on overall performances, confirming the poor behavior of filler models. On the contrary, even a simple assignment of the top neighbor from the dynamic dictionary to OOV sequences used in *IV-OOV Levenshtein* has a slight beneficial impact on overall results (0.5%). The performance gain of *IV-OOV Viterbi* over IV-OOV Levenshtein supports the introduction of a second decoding step that minimizes the cost of the transition between the OOV sequence and its most probable match from the dynamic dictionary.

VI. Conclusion and future work

A common challenge in handwriting recognition is to cope with OOV words. We introduce methods for detecting and correcting such words that are based on large scale external resources. The detection method takes the difference of log-likelihood of IV and OOV word lists and calculates a threshold that classifies a list of unknown new words in one of the two categories. Google and Wikipedia were tested in order to create dynamic dictionaries and the latter resource was shown to be more adapted to the given task. Given the sequences initially classified as OOV, dynamic dictionaries are created from Wikipedia words that are contained in generic and domain-adapted large-scale dictionaries. These correction variants are then fed into a second Viterbi decoder . Our method correctly recognized up to 51.22% OOV sequences, against only 2.4% words recognized when using the filler model. More generally, the introduction of dynamic dictionaries in complement to word networks contributes to the improvement of overall system performances compared to the exclusive use of word networks (72.84% vs. 70.57%).

A first contribution of the proposed OOV words recognition is the innovative use of Wikipedia in a handwriting recognition task that copes with unknown words. The introduction of an efficient domain adaptation constitutes a second important innovation of this work. OOV recognition accuracy is 51.2% for domain dictionary and 43.6% for generic dictionary respectively. A third innovation concerns the introduction of a second Viterbi detection step in order to improve the recognition process compared to the choice of the most similar Wikipedia word predicted with the Levenshtein distance (51.2 vs. 10.4%). Although the introduction of the second Viterbi decoding makes the recognition process more complex, this complexity increase is largely compensated by the obtained performance gain. Last but not the least, the proposed method is easily reproducible since we exploit publicly available resources, and could be applied to other tasks that involve unreliable sequence decodings. Equally important, due to the multilingual dimension of Wikipedia, the method can be straightforwardly adapted to a large number of languages.

The obtained results are very promising and we will pursue work in three important directions. Firstly, the recognition of special cases of OOV words (codes, dates, telephone numbers, etc.) is not treated here since Web resources are not adapted to this task. Dedicated classifiers will be added to the system in the future to further improve performances. Secondly, while effective and simple, the similar words selection (Levenshtein distance, word document frequency, length difference with the OOV sequence) can be improved through the addition of character confusion probabilities. Clustering will be applied to the words initially selected in order to reduce the Viterbi search space while surfacing words that are more likely to be confused with the initially decoded sequence. Thirdly, the number of OOV words in real-world datasets is much higher than in RIMES 2011, a characteristic that reduces the performances of pure word-based models. Consequently, we will verify the hypothesis that the improvements introduced here are even higher for real-world databases. Finally, the execution time for the creation of a dynamic dictionary corresponding to a candidate character sequence takes in average 10.9s and 19.6s for the specific and the generic dictionaries. However, here the emphasis was not on optimizing the computation of the Levenshtein distance. We plan to use an optimized distance computation for the practical deployment of the system.

References

[1] C. Whitelaw and al., "Using the web for language independent spellchecking and autocorrection," in *EMNLP*, 2009, pp. 890–899.

[2] S. Oger and al., "On-demand new word learning using world wide web," in *ICASSP*, 2008, pp. 4305–4308.

[3] C. Parada and al., "A spoken term detection framework for recovering out-of-vocabulary words using the web," in *INTERSPEECH*, 2010.

[4] R. El-Hajj, "Reconnaissance hors ligne de mots manuscrits cursifs pas l'utilisation de systemes hybrides et de techniques d'apprentissage automatique," Ph.D. dissertation, Télécom ParisTech, 2007.

[5] A.-L. Bianne-Bernard, "Reconnaissance de mots manuscrits cursifs par modèles de markov cachés en contexte: application au français, à l'anglais et à l'arabe," Ph.D. dissertation, Télécom ParisTech, 2011.

[6] I.Bazzi and J. R. Glass, "Modeling out-of-vocabulary words for robust speech recognition," in *INTERSPEECH*, 2000, pp. 401–404.

[7] M. Bisani and H. Ney, "Open vocabulary speech recognition with flat hybrid models," in *INTERSPEECH*, 2005, pp. 725–728.

[8] F. Wessel and al., "Confidence measures for large vocabulary continuous speech recognition," *IEEE TSAP*, vol. 9, no. 3, pp. 288–298, 2001.

[9] H. Sun and al., "Using word confidence measure for oov words detection in a spontaneous spoken dialog system," in *INTERSPEECH'03*.

[10] S. Quiniou and É. Anquetil, "Use of a confusion network to detect and correct errors in an on-line handwritten sentence recognition system," in *ICDAR*, 2007, pp. 382–386.

[11] A. Fischer and al., "HMM-based word spotting in handwritten documents using subword models," in *ICPR*, 2010, pp. 3416–3419.

[12] H. Cuayhuitl and B. Serridge, "Out-of-vocabulary word modeling and rejection for spanish keyword spotting systems," in *MICAI'02*, 2002.

[13] S. Young, "The HTK hidden markov model toolkit: Design and philosophy," *Cambridge University, UK, Tech. Rep. TR*, vol. 153, 1993.

[14] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," in *Proceedings of the IEEE*, 1989.

[15] A. Kilgarriff and G. Grefenstette, "Introduction to the special issue on the web as corpus," *Computational Linguistics*, no. 3, pp. 333–347.

[16] V. Levenshtein, "Binary Codes Capable of Correcting Deletions, Insertions and Reversals," *Soviet Physics Doklady*, vol. 10, p. 707, 1966.

[17] F. Damerau, "A technique for computer detection and correction of spelling errors," *Commun. ACM*, vol. 7, pp. 171–176, 1964.

[18] E. Grosicki and H. El-Abed, "ICDAR 2011: French handwriting recognition competition," in *ICDAR*, 2011.