A New Smoothing Method for Lexicon-based Handwritten Text Keyword Spotting

Joan Puigcerver, Alejandro H. Toselli, and Enrique Vidal

Pattern Recognition and Human Language Technology Research Center
Universitat Politècnica de València
Camí de Vera s/n, 46022 València, Spain
{joapuipe,ahector,evidal}@prhlt.upv.es

Abstract. Lexicon-based handwritten text keyword spotting (KWS) has proven to be a very fast and accurate alternative to lexicon-free methods. Nevertheless, since lexicon-based KWS methods rely on a predefined vocabulary, fixed in the training phase, they perform poorly for any query keyword that was not included in it (i.e. out-of-vocabulary keywords). This turns the KWS system useless for that particular type of queries. In this paper, we present a new way of smoothing the scores of OOV keywords, and we compare it with previously published alternatives on different data sets.

Keywords: keyword spotting, lexicon-based, word-graph, smoothing, handwritten text recognition

1 Introduction

The aim of handwritten text keyword spotting (KWS) is to determine, given a predefined confidence level, in which documents or image regions, a given keyword is present. This work, focuses on the case where queries are presented to the system as strings typed by the user (known as Query-by-String) [8].

In the recent past, word-graphs (WG) have been proposed for KWS in handwritten text images [11]. This approach takes the statistical morphological, lexical and language models, previously trained for a handwriting text recognition task, and uses an extension of Viterbi algorithm to generate a word-graph for each text-line, containing a set of possible transcriptions of that particular line and its likelihood. From these WG, line-level confidence scores are computed for each keyword and indexed line, to allow for a fast lookup, with a confidence threshold given by the user.

This method provides much faster searches than traditional lexicon-agnostic methods, such as the HMM-filler method [1], or KWS methods based on bidirectional recurrent neural networks (BLSTM) [2]. Moreover, the Precision–Recall performance provided by the lexicon-based method gives better results than the traditional HMM-filler and comparable with that based on BLSTM.

Unfortunately, an important problem of lexicon-based methods is how to deal with out-of-vocabulary (OOV) queries: since this methods rely on a fixed

vocabulary, determined during the training phase of the system, they will give a null score for any keyword not included in this training lexicon. In order to cope with this problem, this paper presents a new technique for smoothing the scores given by a lexicon-based system, and we compare it with previous publications.

We will show that our proposal improves the ones presented in [6]. In both cases, a similarity metric between an OOV keyword and the indexed ones is used to approximate the score of the OOV events. We also compare our method with [5], which is a combination of lexicon-based and HMM-filler systems, which gives excellent performance results. However the use of the HMM-filler significantly dampers the speed of the system, even when fast search approaches are used [10], as we show in the experiments.

The paper is organized as follows: section 2 introduces basic concepts of the KWS framework used in this work, section 3 briefly describes the existing smoothing methods and presents the proposed method in detail, section 4 presents the experiments conducted to evaluate this work and compare it to previously published works, and final conclusions are drawn in section 5.

2 Keyword Spotting Framework Review

The KWS method used in this work was originally presented in [11]. For each query keyword v and each text line, represented by $\mathbf{x}$, the system tries to model a score $S(\mathbf{x}, v)$ which measures how likely is the event "keyword v is written in $\mathbf{x}$", or re-phrased as "text line $\mathbf{x}$ is relevant for keyword v". Following [11], we define the score as:

$$S(\mathbf{x}, v) \stackrel{\text{def}}{=} \max_{1 \leq i \leq n} P(v \mid \mathbf{x}, i) \tag{1}$$

$P(v \mid \mathbf{x}, i)$ is known as the *frame-level word posterior*, which is the probability that the word v is present in the line image $\mathbf{x}$ at position i. As shown in [11], this posterior can be directly approximated from the WG of the line image obtained as a byproduct of recognizing the image with a HTR system based on optical models such as HMMs and n-gram language models.

Observe that the previous score is equivalent to the probability of a Bernoulli distribution over a random variable R, measuring how likely is the event "text line $\mathbf{x}$ is relevant for keyword v".

$$P(R \mid \mathbf{x}, v) \stackrel{\text{def}}{=} \begin{cases} S(\mathbf{x}, v) & R = 1 \\ 1 - S(\mathbf{x}, v) & R = 0 \end{cases} \tag{2}$$

3 Out-of-Vocabulary Queries

In the introduction we briefly explained one drawback of the lexicon-based KWS method, related to the out-of-vocabulary keywords. Since this KWS approach uses a n-gram LM which assigns a null probability to any keyword not seen

during its training, the frame-level word posterior $P(u \mid \mathbf{x}, i) = 0$, for all keyword $u \notin V$, on any line image $\mathbf{x}$.

In this section, we will briefly explain some of the solutions that have recently been introduced in the literature to mitigate this problem, along with our contribution.

3.1 Line-level Smoothing with Levenshtein Distance

In [6], a first attempt to solve this problem is presented. Here the score of an OOV keyword u is smoothed using the scores of the indexed keywords $v \in V$ and the Levenshtein distance $d(u, v)$ between both strings, using:

$$S(\mathbf{x}, u) \stackrel{\text{def}}{=} \max_{v \in V} S(\mathbf{x}, v)^{1-\alpha} \cdot e^{-\alpha d(u,v)} \tag{3}$$

The parameter α is tuned using a validation set and is intended to balance the contribution of the Levenshtein distance on the score of the OOV event.

3.2 Frame-level Word Posterior Smoothing

In the previous work [6], a frame-level smoothing is also presented. This method, first smooths the frame-level word posterior $P(u \mid \mathbf{x}, i)$ of a keyword $u \notin V$, following:

$$\tilde{P}(u \mid \mathbf{x}, i) = \frac{\sum_{v \in V} P(v \mid \mathbf{x}, i) \cdot f(u, v)^{\alpha}}{1 + \sum_{v \in V} P(v \mid \mathbf{x}, i) \cdot f(u, v)^{\alpha}} \tag{4}$$

The parameter α is tuned again using a validation partition and the function $f(u, v)$ is a similarity measure based, on a stochastic error correcting approach, indicating how similar are the keywords $u \notin V$ and $v \in V$. Finally, the score $S(\mathbf{x}, u)$ is computed as in Eq.1, but using $\tilde{P}(u \mid \mathbf{x}, i)$ instead.

3.3 Lexicon-based and HMM-Filler combination

This approach, presented in [5], is actually a combination of two different systems, each of them with different strengths and issues. When a query keyword v is indexed (i.e. not an OOV), it can be immediately honored using the precomputed scores given by the lexicon-based approach. Otherwise, a lexicon-free HMM-filler model is used to serve the query. The resulting score is obtained using:

$$S(\mathbf{x}, v) = \begin{cases} S_G(\mathbf{x}, v) & v \in V \\ \exp(S_F(\mathbf{x}, v))^{\eta} & v \notin V \end{cases} \tag{5}$$

Here, $S_G(\mathbf{x}, v)$ are the scores given by the lexicon-based system and $S_F(\mathbf{x}, v)$ are the scores given by the HMM-filler. The exponentiation of the HMM-filler scores and the parameter η are required because this model does not give properly normalized scores, which have to be be scaled to an appropriate range.

3.4 New Proposed Line-level Smoothing

The proposed line-level score smoothing is based on the probabilistic interpretation of the scores obtained from the lexicon-based KWS approach. As we mentioned before, the score $S(\mathbf{x}, v)$, given by this approach, can be interpreted as $P(R \mid \mathbf{x}, v)$. Then, we can marginalize this distribution for a particular $u \notin V$, with all $v \in V$, resulting in:

$$P(R \mid \mathbf{x}, u) = \sum_{v \in V} P(R, v \mid \mathbf{x}, u) = \sum_{v \in V} P(R \mid \mathbf{x}, u, v) \cdot P(v \mid \mathbf{x}, u) \tag{6}$$

Then, we make two independence assumptions: First, assume that v is conditionally independent of $\mathbf{x}$, given u, i.e. $P(v \mid \mathbf{x}, u) \approx P(v \mid u)$. Then, we assume that $P(R \mid \mathbf{x}, u, v) \approx P(R \mid \mathbf{x}, v)$, which implies that the fact that a line $\mathbf{x}$ is relevant for a pair of keywords $u \notin V$ and $v \in V$, actually depends only on the keyword $v \in V$. After these assumptions, Eq.6 is approximated as:

$$P(R \mid \mathbf{x}, u) \approx \sum_{v \in V} P(R \mid \mathbf{x}, v) \cdot P(v \mid u) \tag{7}$$

$P(v \mid u)$ is a similarity probability distribution which has to normalized across all $v \in V$, which differs from the one introduced in [6], which had to be normalized across all $u \in \Sigma^*$. Since the distribution is over a finite set of elements, it can be defined in arbitrary ways that do not exhibit the problems of [6] (it was affected by the length of the keywords and had estimation issues). Particularly, we choose a distribution based on the Levenshtein distance $d(u, v)$:

$$P(v \mid u) \stackrel{\text{def}}{=} \frac{\exp(-\alpha\ d(u, v))}{\sum_{v' \in V} \exp(-\alpha\ d(u, v'))} \tag{8}$$

In the same way that the previous smoothing methods did, we introduce an parameter α to tune the contribution of the similarity measure.

4 Experiments

In order to compare the proposed method with previous works, several experiments were conducted on different corpora. Performance assessment, corpora, experimental setup and results are explained below.

4.1 Performance assessment

Assessment is measured using the average precision (AP) metric [7], based on the recall and precision measures. As is usually done, we use the interpolated precision in order to smooth plain precision [4]. AP is a scalar summary of the precision and recall, which are functions of a threshold used to determine whether the score $S(\mathbf{x}, v)$ is high enough to assume that v is relevant in $\mathbf{x}$. Additionally,

we report the mean average precision (mAP), which is also widely adopted in the literature. It is computed by averaging the AP of each keyword. We decided to optimize our parameters based on the AP metric.

Finally, we also compared the number of seconds required to compute the scores of an OOV query, in each corpus. We only considered the OOV keywords for this comparison, since the scores of in-vocabulary keywords can be precomputed and the lookup speed becomes asymptotically constant.

4.2 Corpora

The "Cristo-Salvador" (CS) dataset is a small XIX century single-writer Spanish manuscript. We used exactly the same partitioning as [10, 6]. Since the CS corpus is quite small, we ignored capitalization and diacritics to build the lexicon and the LM, and performed cross-validation to tune all parameters.

Regarding IAM, it consists of English handwritten texts from many writers. We used the same data partitions used in previous KWS experiments [1, 2, 10, 5]. In addition to the text in the line images, we used three external text corpora (LOB, Brown and Wellington), which were used to build a 20K-word lexicon and train a LM (test lines were excluded from LOB).

In both cases, we used the lexicon of the test lines as the query set. In the case of IAM, we subtracted from the query set all stop words, as in [5]. Tab.1 summarizes the most important information of the corpora.

Table 1: Tables summarizing the corpora used for experimentation.

(a) Basic statistics of the selected databases.

	CS		IAM		
	Train	Test	Train	Valid.	Test
Running Chars	35 863	26 353	269 270	39 318	39 130
Running Words	6 223	4 637	47 615	7 291	7 197
# Lines	675	497	6 161	920	929
Char Lex. Size	78	78	72	69	65
Word Lex. Size	2 236	1 671	7 778	2 442	2 488
OOV Lex. Size	—	1 051	—	435	437

(b) Details of the selected query sets for the test partition in each dataset.

	CS	IAM
# Line images: N	497	929
# Query words: M	1 671	2 209
# Line-query events: $M \cdot N$	830 487	2 052 161
# OOV Line-query events	522 347	405 973
# Relevant line-query events	4,346	3 446
# Relevant OOV line-query events	1,341	496

4.3 Experimental setup

In each corpus, we trained a a left-to-right HMM model with GMM distributions on the states, for each character included in the training set. The standard embedded Baum-Welch training algorithm was used [12]. Details about preprocessing, feature extraction, number of states and mixtures in the GMM, etc. can be found in [11] for the CS dataset, and [1] for IAM.

A bi-gram LM was used to build the lexicon-based system. In the case of CS, the transcripts of the training set were used, converting lowercase characters to uppercase. For IAM, the LM was trained using the external LBW corpus, restricted to the 20K most frequent words. Finally, the standard Kneser-Ney back-off was used to smooth the probabilities of unseen bi-grams [3].

The approach described in [11] was used, in order to speedup the search using the HMM-filler. In a *preparatory phase*, character-lattices (CL) are obtained using the "filler", for each of the test lines. The maximum node input degree (NID) of these was set to 30. Then, during the *search phase*, the scores of each query are computed using the CL. For the lexicon-based method, WG were generated using the standard HTK software, with the described language models, with a maximum NID value equal to 40, and not using any pruning technique during the decoding. Further details about the lexicon-based method can be found in [6, 5].

Concerning the tuning of parameters, we used the same values as the reference papers [6, 5], for the existing methods. As for our method (section 3.4), we used the IAM validation set and performed cross-validation on CS. We finally set α to 4, for both corpora.

Experiments were conducted on a Intel Core 2 Quad Q9550 CPU at 2.83GHz, running Ubuntu Linux 14.04. All custom software was implemented in C++.

4.4 Results

Table 2 summarizes the main results obtained on the test set of each corpora.

Table 2: Line-level Average Precision (AP) and Mean Average Precision (mAP) provided by each smoothing method on the test set of the corpora. Results tagged with † and ‡ were presented in [6, 5], respectively. Query time (Qtime) is the average time required to serve the OOV queries, expressed seconds.

	CS			IAM		
Method	AP	mAP	Qtime	AP	mAP	Qtime
No smoothing	55.6	29.0	—	69.1	68.8	—
Line Max. (sec. 3.1)	†57.8	†45.0	0.44	69.8	76.0	8.78
Posteriorgram (sec. 3.2)	†58.8	†46.7	27.21	70.2	76.1	42.48
WG + HMM-Filler (sec. 3.3)	72.5	76.6	177.10	‡76.9	‡82.2	58.16
This work (sec. 3.4)	59.5	46.0	0.52	71.3	76.0	9.96

Compared to the method described in 3.1, for both datasets, our proposal improves the previous AP results (about 1.7 points of absolute improvement), while maintaining similar computational costs. This is because the asymptotic running time of both algorithms is the same and the new method successfully uses more information from the index: the contribution of all in-vocabulary keywords is considered, instead of a maximum.

Moreover, the proposed work slightly improves the AP of the method described in 3.2 (about 0.9 points of absolute improvement), and is able to obtain the smoothed scores in much faster times (about 50 times faster in CS and 4 times faster in IAM). It is important to notice that the running time of method 3.2 depends both on the size of V, and the average number of edges per node in the line WG, which can be interpreted as the *perplexity* of the WG. In the CS case, the perplexity of the WG is much higher than in the IAM case (35.8 vs 23.6), since fewer training data was available for both the HMM and LM. However, the size of the vocabulary is much higher for IAM. On the other hand, the perplexity of the WG does not affect the speed of our method, which works directly with the index scores. The interaction between these two factors explains why method 3.2 is slower in IAM than in CS, but is faster in IAM than in CS when it is *relatively* compared to our proposal.

It is worth pointing out that the computation of the Levenshtein distances has been done naively in this work, with an asymptotic cost of $O(|q| \cdot L \cdot |V|)$, where q is the query string, $L = \max_{v \in V} |v|$, and $|s|$ gives the length of string s. Nevertheless, this computation can enormously reduced by using *tries* for indexing the vocabulary, or limiting the maximum number of errors allowed [9].

Finally, we show that the combination of the Lexicon-based and the HMM-filler methods gives better AP results than any of the proposed methods, but at a much higher computational cost: our method is 340 times faster in CS and 6 times faster in IAM, when compared to 3.3. The reason explaining the differences in the speedup is the same as explained above.

5 Conclusions

We presented a new method for smoothing the scores given by the lexicon-based system, based on the similarity between the OOV keyword and the indexed vocabulary. Futhermore, we performed a detailed comparison of different alternatives that try to alleviate the problem caused by OOV queries when using a lexicon-based KWS system for handwritten text images.

The presented smoothing method gives better results than most of the compared alternatives, and it is comparable in speed to the fastest of the previous methods, also based on similarity measures.

Only the combination of a lexicon-based system and a HMM-filler surpasses our proposal, among the studied methods. Nevertheless, it comes with a time cost several orders of magnitude bigger than our proposal, specially when the CL are big or have a high average number of edges per node, which may result impractical in some real scenarios (i.e. Cristo-Salvador corpus).

In the future, we are planning to investigate ways of effectively indexing open-lexicon systems and combine them with lexicon-based approaches, thus allowing for faster and more accurate searches for both in-vocabulary and out-of-vocabulary queries.

Acknowledgments

This work was partially supported by the Spanish MEC under FPU grant FPU13/06281 and under the STraDA research project (TIN2012-37475-C02-01), by the Generalitat Valenciana under the grant Prometeo/2009/014, and through the EU 7th Framework Programme grant tranScriptorium (Ref:600707).

References

1. Fischer, A., Keller, A., Frinken, V., Bunke, H.: Lexicon-free handwritten word spotting using character HMMs. Pattern Recognition Letters 33(7), 934 – 942 (2012), special Issue on Awards from ICPR 2010
2. Frinken, V., Fischer, A., Manmatha, R., Bunke, H.: A Novel Word Spotting Method Based on Recurrent Neural Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence 34(2), 211 –224 (Feb 2012)
3. Kneser, R., Ney, H.: Improved backing-off for N-gram language modeling. In: International Conference on Acoustics, Speech and Signal Processing (ICASSP '95). vol. 1, pp. 181–184. IEEE Computer Society, Los Alamitos, CA, USA (1995)
4. Manning, C.D., Raghavan, P., Schtze, H.: Introduction to Information Retrieval. Cambridge University Press, New York, NY, USA (2008)
5. Puigcerver, J., Toselli, A.H., Vidal, E.: Word-Graph and Character-Lattice Combination for KWS in Handwritten Documents. In: 14th International Conference on Frontiers in Handwriting Recognition (ICFHR). pp. 181–186 (2014)
6. Puigcerver, J., Toselli, A.H., Vidal, E.: Word-Graph-based Handwriting Keyword Spotting of Out-of-Vocabulary Queries. In: 22nd International Conference on Pattern Recognition (ICPR). pp. 2035–2040 (2014)
7. Robertson, S.: A new interpretation of average precision. In: Proc. of the Intl. ACM SIGIR conference on Research and Development in Information Retrieval (SIGIR '08). pp. 689–690. ACM, New York, NY, USA (2008)
8. Rodriguez-Serrano, J.A., Perronnin, F.: Handwritten word-spotting using hidden markov models and universal vocabularies. Pattern Recognition 42(9), 2106–2116 (2009), http://www.sciencedirect.com/science/article/pii/S0031320309000673
9. Shang, H., Merrettal, T.: Tries for approximate string matching. Knowledge and Data Engineering, IEEE Transactions on 8(4), 540–547 (1996)
10. Toselli, A.H., Vidal, E.: Fast HMM-Filler approach for Key Word Spotting in Handwritten Documents. In: Proceedings of the 12th International Conference on Document Analysis and Recognition (ICDAR). pp. 501–505 (2013)
11. Toselli, A.H., Vidal, E., Romero, V., Frinken, V.: Word-Graph Based Keyword Spotting and Indexing of Handwritten Document Images. Tech. rep., Universitat Politcnica de Valncia (2013)
12. Woodland, P., Leggetter, C., Odell, J., Valtchev, V., Young, S.: The 1994 HTK large vocabulary speech recognition system. In: Intl. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '95). vol. 1, pp. 73 –76 vol.1 (may 1995)