KEY-PHRASE SPOTTING USING AN INTEGRATED LANGUAGE MODEL OF N-GRAMS AND FINITE-STATE GRAMMAR

Qiguang Lin *Dave Lubensky* *Michael Picheny* *P. Srinivasa Rao*

IBM Watson Research Center - Human Language Technologies Group
P.O. Box 218, Yorktown Heights, NY 10598, USA
email: qlin@watson.ibm.com

Abstract

This paper describes a new algorithm for key-phrase spotting applications. The algorithm consists of three processes. The first process is to synergistically integrate N-grams with Finite-State Grammars (FSG) – the two conventional language models (LM) for speech recognition. All the key phrases to be spotted are covered by the FSG component of the recognizer's LM, while the N-grams are used for decoding surrounding non-key phrases. Secondly, selective weighting is proposed and implemented. The weighting parameters independently control the triggering and completion of FSG on top of N-grams. Finally, the third process involves a word confirmation and rejection logic which determines whether to accept or reject a hypothesized key phrase. The proposed algorithm has been favorably evaluated on two separate experiments. In these experiments, only the FSG part of the LM need be updated for different application tasks while the N-gram part can remain unchanged.

1 Introduction

Language modeling is one of the important components of contemporary, statistical-based speech recognition systems. Given a (partial) list of immediately preceding words, the language model predicts what words are most likely to follow. The final decoding result is determined by the combined likelihood scores of acoustic and language modeling [1]. A dichotomy has evolved among the existing schemes for language modeling. In one class of schemes, the current word is inferred based on the previous $(N-1)$ words. This method is generally referred to as N-grams. Most speech recognition systems employ trigrams $(N=3)$, backing off to bigrams $(N=2)$ and unigrams $(N=1)$. In the other class of language modeling, formal grammars (usually implemented in terms of finite-state networks) are used. Both classes have been shown to be effective for speech recognition. The major limitation of trigrams is the lack of long-range dependency. Use of a large N could preserve the long-range dependency but would also require much more text data to adequately train the N-grams. On the other hand, the major drawback of grammar approaches is relatively narrow coverage of language phenomena.

Motivated by these relative advantages and disadvantages, attempts have been made to synergistically integrate N-grams and FSG for key-phrase spotting (i.e., to detect the occurrence of some preselected words or phrases buried in continuous speech). Usually, key phrases are limited and stable for a given application (such as telephone numbers). Therefore, they are most effectively modeled by FSG to take advantage of prior structure information. On the other hand, surrounding carrier phrases may vary substantially and are best modeled by N-grams to take advantage of wide coverage and easy training (cf. [4]).

Significant progress in keyword spotting has been achieved in recent years. A prevailing method is to construct an acoustic filler or garbage model which provides an effective means for likelihood score normalization or a confidence measure (see, e.g., [3, 7, 9]). For a given observation $O = \{o_1, ..., o_T\}$, let $Pr(O|\lambda_0)$ denote the likelihood that O has been produced by a keyword model λ_0, and $Pr(O|\lambda_1)$ the likelihood that O has been produced by the filler model λ_1. The following likelihood ratio then decides whether keyword, w_i, is present in O:

$$L_i = \frac{Pr(O|\lambda_0)}{Pr(O|\lambda_1)} \qquad (1)$$

The filler model, λ_1, can be key-word dependent (also known as anti-model) or independent, and can be based on modeling whole words or subword units [7, 9]. Both acoustic fillers and lexical fillers have been investigated [3]. Filler models are usually trained on all nonkeywords using the maximum likelihood method. Some recent studies have implemented discriminative training methods for keyword spotting, to maximize the "contrast" between a particular model and its anti-model and thereby elevate the performance of the spotter [6, 8].

In the present study, no explicit acoustic nor lexical fillers are used. Instead, the N-grams have an analogue function to the filler to provide a means for likelihood score normalization. For each potential key phrase, two competing likelihood scores are obtained. One is from N-grams, $Pr(O|Ngrams)$ and the other is from FSG, $Pr(O|FSG)$. A key-phrase putative hit is detected if the ratio

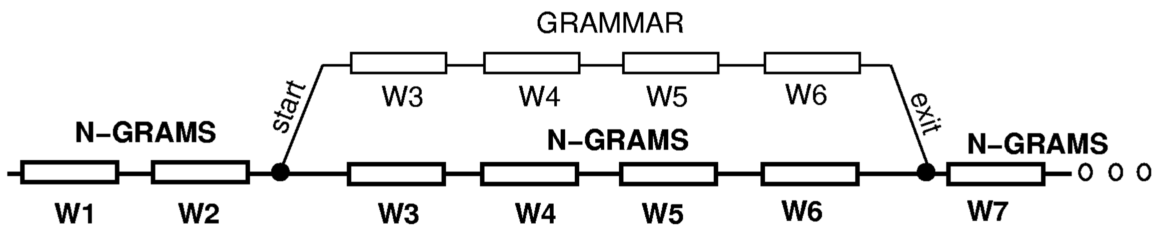

Figure 1: *Block diagram of word sequences in a sentence, illustrating the synergy of N-grams and grammar.*

$$L = \frac{Pr(O|FSG)}{Pr(O|Ngrams)} \tag{2}$$

is above a given threshold. It is noted that eqs. (1) and (2) are similar.

A hybrid LM combining N-grams and phrase-structure grammar has previously been devised to improve speech recognition performance [4]. In the present study, it is used for key-phrase spotting applications. A novel approach for incorporate FSG into N-grams is suggested. The approach ensures proper initialization and completion of the FSG, independent of the task-dependent N-grams. Consequently, the algorithm tends not to miss a key phrase (error type I). As for error type II (false alarm), word confirmation logic is applied to the spotted key phrase. The logic determines whether to accept or reject the spotted candidate.

2 Combining N-grams/FSG

For a LM with integrated N-grams and FSG, the N-grams can be trained with a label being used to replace all phrases in the text corpora which are covered by the grammar. The resultant N-gram model therefore contains a vocabulary of terminal words (not related to grammar) and non-terminals (grammar labels). During recognition, each non-terminal is replaced by the FSG. For a word not included in the grammar, only the N-gram portion of the language model is utilized. The grammar is triggered when the decoder sees legitimate words that start the grammar, see Figure 1. Once triggered, the finite-state network is propagated in parallel with the N-grams, hence generating two competing likelihood scores useful for the confidence measure purpose, as previously mentioned. If the finite-state network can be successfully completed and its associated likelihood score is higher than the N-gram counterpart, a putative key-phrase event is found. Otherwise, there is no key phrase present in the utterance.

3 Selective weighting

It is apparent from the previous section that the trained N-grams are dependent upon applications. When the application changes such as digits to non-digits, the N-grams need to be retrained. It is always desirable to alleviate such application dependency constraints as much as possible. In the present study, a novel approach to incorporating FSG into N-grams is suggested. More specifically, we have introduced an initial boost factor and a selective penalty factor to ensure proper initialization and propagation of the FSG.

1. *initial boost α:* This parameter independently controls how easy or how difficult the grammar is started off on top of the N-gram model. For the example shown in Figure 1, we have the N-gram score $Pr(w_3|w_2, w_1)$, and the elevated FSG score $\alpha \cdot Pr(FSG|w_2, w_1)$.

2. *selective penalty β:* Once the finite-state network is entered, it is desirable that the complete network path can be successfully propagated and finished at the exit node. (A completed path is a key phrase in a key-phrase spotting task.) The selective penalty parameter punishes early exit from the network, depending on the depth of the network which has been visited:

$$PENALTY = \prod_{i=1}^{I} \beta \tag{3}$$

   where β is the penalty after the grammar network is first visited, and I denotes the visited depth in terms of the number of words. I is reset as soon as the FSG network is completed successfully (so as to handle multiple key phrases in an utterance). The penalty of eq. (3) is computed for every extended word and added to the N-gram path of Figure 1. Alternatively, the penalty can be translated into "reward" and added to the FSG path.

Compared with prior approaches [4, 5], the present system with suggested selective weighting has several advantages. For example, it is particularly well suited for key-phrase spotting applications. Each of the key phrases to be spotted can be directly modeled in the (phrase) grammar, and the above new features jointly ensure detection of each key phrase when it occurs. It is possible that the new features may increase more false alarms (type II error, non-key phrases decoded as key phrases). This problem can be effectively solved by applying some additional rejection logic.

Furthermore, the N-gram model is used to decode non-key phrases only. It is thus possible that

LM used	phone number string error rate	word error rates (for all words)
trigram	38%	15.3%
trigram+FSG	28%	9.2%
trigram+FSG +weighting	14%	4.8%

Table 1: *Phone number string error rate (key-phrase spotting related scores) and word error rate for different language models.*

a simple N-gram model will lead to satisfactory spotting performance, independent of task domain. In other words, it is necessary to only update the grammar part and the above boosting/penalty parameters when changing to a new application. The N-gram model may remain unchanged or preferably become augmented by a small amount of task-specific data.

4 Experimental results

Two experiments have been conducted to evaluate the proposed algorithm. The results are summarized below. Note that a trigram model, trained on general-purpose English text, is used in both experiments, except as otherwise specified.

4.1 Experiment 1

The task of Experiment 1 is detection/recognition of 7-digit telephone numbers embedded in sentences of read, discrete speech. The testing speech data is from a male speaker and has 61 sentences with 63 strings of phone numbers. The total number of words (phone numbers and others) in the test set amounts to 1529. The speech is sampled at 11 KHz. The front-end process involves (i) preemphasis; (ii) computing frame energy and FFT spectra every 10 ms using a 25 ms Hamming window; (iii) converting the mel-band output of the spectra to 12-dimensional cepstral coefficients, MFCC's (excluding C_0); (iv) removing sentence-wise means of MFCC's and normalizing the frame energy; (v) computing first-order and second-order derivatives of MFCC's and frame energy. The final acoustic vector for each frame thus constitutes 39 elements. The recognizer is a speaker-independent large vocabulary HMM recognizer based on IBM VoiceType technology. Each of the 52 phones is modeled with 3 HMM arcs and rank distribution histograms are used for computing likelihood scores (cf. [1]). The recognizer has approximately 20k Gaussian mixtures and a vocabulary of over 26k words.

The recognition results are tabulated in Table 1 for three different LM conditions. It is seen that the integrated N-grams and FSG is capable of driving the word recognition error rate (for phone numbers and other words) from 15.3% to 9.2%, a reduction of 40%. The suggested weighting further reduces the word error rate to 4.8% which corresponds to a 69% reduction from the baseline (trigram alone).

The center column in Table 1 gives the string error rate of the phone number. This measure is more relevant to spotting tasks. It is clear that the proposed approach, *trigram+FSG+weighting*, yields the best string error rate of 14%. Without selective weighting, the string error rate is doubled to 28%.

4.2 Experiment 2

This experiment is a telephony application with field recordings of inquiries about business names. The corpus contains 460 speakers and 2890 testing utterances. A set of 85 key-phrases were chosen to cover approximately 2/3 of the utterances in the corpus. (1897 utterances each contain a key phrase, and the remaining 993 contain no key phrases.) Basically, the recognizer is similar to the one used in Experiment 1, but it has now a smaller vocabulary of 751 words.

Again, a number of different LM's are utilized in this experiment. First, the trigram for the test data is computed (often called a cheating trigram) and used to establish the upper-bound performance. Then an FSG alone is used to see how well an FSG would serve telephony inquiry application. The results are shown in Table 2 in terms of *correct hit %* and *false alarm %*. *correct hit %* is defined as the ratio of the number of the key phrases correctly recognized to the total number of utterances containing a key word, and *false alarm %* is defined as the ratio of the number of non-key phrases misrecognized as key phrases to the number of the total utterances containing non-key phrases. From Table 2 it is seen that the cheating trigram and the FSG produce similar putative hit rates, but the latter generates 5 times more false alarms.

The next step is to experiment with a hybrid LM integrating N-grams and FSG. As in Experiment 1, the same trigram model for general-purpose English text is used here. The obtained results, also given in Table 2, show (1) that when the weighting is not applied, the spotter tends to miss key phrases resulting in a low hit rate as well as false alarm; (2) that the weighting helps spot more key phrases at the expense of increased false alarms; (3) that the combination of *trigram+FSG+weighting* is better than FSG alone.

Because many false alarms are generated using either *trigram+FSG+weighting* or *FSG* alone, we decide to incorporate a word confirmation and rejection logic immediately after key-phrase spotting. The block diagram in Figure 2 illustrates the rejection logic we implement and use in Experiment 2. As shown in Table 2, the combination of "*trigram+FSG+weighting plus word rejection*" outperforms the combination of "*FSG plus word rejection*".

The combination of "*trigram+FSG+weighting plus*

LM used	correct hit %	false alarm %
cheating trigram	88.7	3.8
FSG	85.6	19.6
trigram+FSG	**66.2**	**3.6**
trigram+FSG+weighting	85.9	14.8
FSG *PLUS WORD REJECTION*	81.4	8.9
trigram+FSG+weighting PLUS WORD REJECTION	**84.6**	**7.6**

Table 2: *Correct detection of key phrases and false alarm, in percentage, obtained with different LM's.*

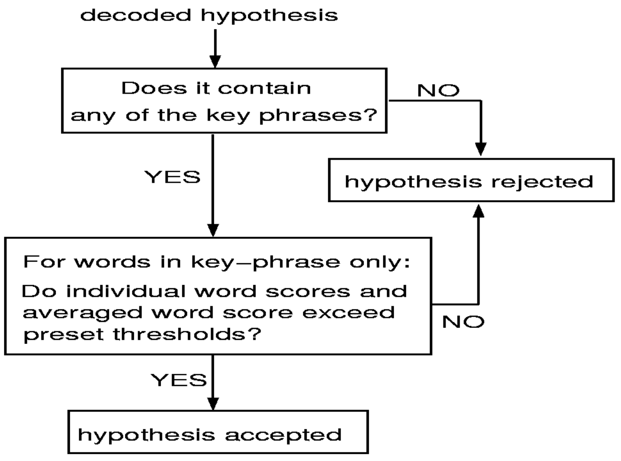
decoded hypothesis
Does it contain any of the key phrases?
NO
YES
hypothesis rejected
For words in key–phrase only: Do individual word scores and averaged word score exceed preset thresholds?
NO
YES
hypothesis accepted

Figure 2: *Block diagram of the decision logic which determines whether to accept or reject a decoded hypothesis.*

word rejection" also gives a performance competitive to that achieved by the cheating trigram. It should be noted that the telephony inquiry application is much different from general-purpose English text. In other words, the trigrams for the general-purpose English text have been successfully used in a domain-independent manner in the experiment.

5 Conclusion

This paper has presented a technique for key phrase spotting. The algorithm involves three processes. The first process is to synergistically integrate N-grams with FSG. All the key phrases to be spotted are covered by the FSG component of the decoder's LM, while the N-grams are used for decoding surrounding non-key phrases. Secondly, selective weighting is implemented. The weighting parameters independently control the triggering and completion of FSG independent of the used N-grams, and hence, help detect putative key-phrase hits. The weighting may at the same time lead to an increased false alarm rate. Therefore, in the final process a confirmation and rejection logic is utilized to reject a misrecognized hypothesis. The rejection is based on the comparison of the likelihood scores of words in the key phrase and a preset threshold.

Two separate experiments have been conducted to evaluate the proposed technique, and good key-phrase spotting performance has been obtained. It should be reminded that Experiment 2 is for spontaneous telephone speech with a sizable number of key phrases to be spotted. Often, these factors make the difficult task of phrase spotting even more challenging!

Compared with filler models, the described system has an advantage that its acoustic training is independent of the key-phrase vocabulary. Furthermore, the suggested selective penalty alleviates the domain- or task-dependency of the trained N-grams as shown in the above experiments. Currently, the weights are determined experimentally based on several sentences. We expect to develop an automatic method for estimating the weights in the future. We also expect to further improve the present word rejection scheme including the use of word-specific anti-models [6, 8].

References

[1] Bahl, L., de Souza, P., Gopalakrishnan, P., Nahamoo, D., and Picheny, M., "Robust methods for using context-dependent features and models in a continuous speech recognizer," *ICASSP94*, pp. 533-536.

[2] Mark, K., Miller, M., Grenander, U., and Abney, S., "Parameter estimation for constrained context-free language models," *Proc. Speech and Language Workshop*, San Mateo, CA. pp. 146-149.

[3] Meliani, R. and O'Shaughnessy, D.: "Accurate keyword spotting using strictly lexical fillers," *ICASSP97*, pp. 907-910.

[4] Meteer, M. and Rohlicek, R., "Statistical language modeling combining N-gram and context-free grammar," *ICASSP93*, pp. II-37-40.

[5] Periera, F., and Schabes, E.,"Inside-outside reestimation from partially bracketed corpora," *Proc. Speech and Language Workshop*, San Mateo, CA. 1992, pp. 122-127.

[6] Rahim, M., Lee, C.-H., and Juang B.-H.: "Discriminative utterance verification for connected digits recognition," *Proc. Eurospeech95*, pp. 529-532.

[7] Rose, R. and Paul, D.: "A hidden Markov model based keyword recognition system," *Proc. ICASSP90*, pp. 129-132.

[8] Sukkar R. and Lee, C.-H.: "Vocabulary independent discriminative utterance verification for non-keyword rejection in subword based speech recognition," *Trans. Speech Audio Proc.* **40**, pp. 420-429.

[9] Wilpon, J., Rabiner, L, Lee, C.-H., and Goldman, E.: "Automatic recognition of keywords in unconstrained speech using hidden Markov models," *Trans. ASSP* **38**, pp. 1870-1990.