Hybrid Speech Recognition for Voice Search: a Comparative Study

Evandro Gouvêa

Carnegie Mellon University, Pittsburgh, USA

evandro.gouvea@alumni.cmu.edu

Abstract

We compare different systems for use in information retrieval of items by voice. These systems differ only in the unit they use: words, a subwords, a combination of these into a hybrid, and phones. The subword set is derived by splitting words using a Minimum Description Length (MDL) criterion. In general, we convert an index written in terms of words into an index written in terms of these different units. A speech recognition engine that uses a language model and pronunciation dictionary built from each such an inventory of units is completely independent from the information retrieval task, and can, therefore, remain fixed, making this approach ideal for resource constrained systems. We demonstrate that recognition accuracy and recall results at higher OOV rates are much superior for the hybrid system than the alternatives. On a music lyrics task at 80% OOV, the hybrid system has a recall of 82.9%, compared to 75.2% for the subword-based one and 47.4% for a word system.

Index Terms: information retrieval by voice, subword units, minimum description length, hybrid systems

1. Introduction

Information retrieval by voice is already a very important market. Several applications, specially involving smartphones, use speech as the preferred input modality for making queries to search engines, sometimes in combination with other modes, such as typing or pointing.

A prototypical system for spoken query retrieval is shown in Figure 1, depicting the system's two main components: an automatic speech recognition (ASR) front-end and an information retrieval (IR) back-end. The ASR front-end produces an N-best list of recognition hypotheses from an input spoken query. The N-best list is then submitted to the IR back-end, which retrieves the top-K relevant documents for that query.

Ideally, the language model (LM) used by the ASR is built from the entries in the database to be indexed. The drawback of this approach is that if the set of documents in this database changes, the LM has to change. Particularly, new databases may have words not present before. Therefore, changes in the databases to be indexed require that we re-prune or re-compress the LMs.

A more practical approach is to build a system where the LM remains fixed as the set of documents to be indexed changes. This is particularly important in systems with low resources, or when the LM becomes too large. Such a system, however, cannot predict every word contained in any possible document. The system necessarily needs to handle out of vocabulary (OOV) words such that the retrieval performance degrades gracefully as the OOV rate increases.

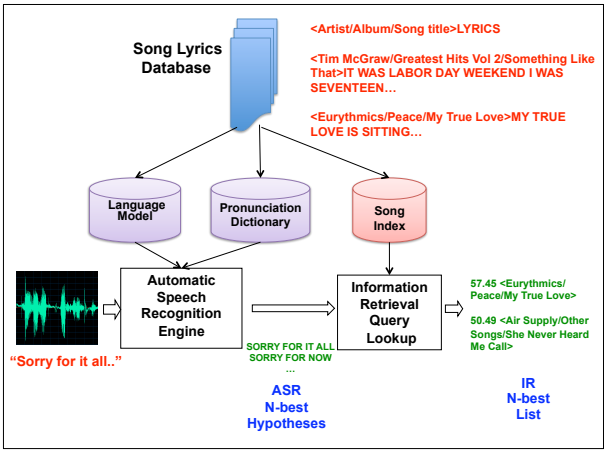

Figure 1: Overview of an Information Retrieval by Voice for a Song Lyric Task

Subword units have been successfully used to handle OOVs in ASR systems [1][2][3], mainly with the goal of improving speech recognition accuracy. As such, subword units are normally converted to a form that is more amenable to a human reader.

In IR tasks, subword units can be used directly, without a need for reconversion. In a previous work [4], we presented a system where the front end ASR knowledge base, *i.e.* the pronunciation and language model, and the back end IR index were entirely subword-based. Importantly, these phonetic subword units were vocabulary independent: the LM and lexicon used by the ASR engine remained the same even if we changed the set of documents to retrieve. Novel databases were simply rewritten in terms of the subword unit inventory. Therefore, we satisfied our goal of dissociating the text used to build the ASR knowledge base from the IR index.

In [5] we extended this previous work by studying the effects of OOV words in the information retrieval task. The results were quite satisfactory, but there still were

HOURGLASS	AW_R + G_L_AE_S
HOUSE	HH_AW_S
HOUSES	HH_AW_S + IH_Z
HOUSES(2)	HH_AW + Z + AH + Z

Table 1: Examples of words rewritten in terms of subwords. Note that some words with alternate pronunciations have multiple subword representations.

conditions where a word based system outperformed our subword based one.

OOVs are handled by hybrid systems in an increasing range of IR tasks, such as spoken term detection [6][7] and question answering [8]. We considered the benefits of using it also for voice search. A hybrid system has at least one clear benefit in comparison with a purely subword based one, as follows. The subword system maps each pronunciation alternate to a different sequence of subword units. Therefore, words that have multiple pronunciations end up having its alternates being treated as separate entities. In contrast, in a hybrid system, at least the most frequent terms are kept as words, possibly with multiple alternate pronunciations.

The hybrid system can also be designed in a way that, through a combination of words and subword units, a dictionary and language model are built and remain fixed as the IR task changes. Therefore, it has the advantage of the subword-based system, *i.e.* the system does not have to be rebuilt for every change in the IR database, and the most frequent units can have multiple pronunciations.

We study the effect of increasing OOV rates on the recall rate and recognition error rate as we compare the baseline word-based system with our previous subword-based one, a hybrid system, and a purely phonetic one, as an extreme case.

In Section 2 we summarize the main points of the method we use to obtain subword units. In Section 3 we describe the experimental setup, and in Section 4 we present and discuss the results, concluding in Section 5.

2. MDL Subword Unit Inventory

Table 1 helps to clarify our definition of a subword unit. A word, *e.g.* HOURGLASS, is rewritten as a sequence of subword units AW_R and G_L_AE_S, where the subword units are sequences of *phonemes*. A subword unit may also span an entire word, as with HOUSE. Our algorithm rewrites a database I in terms of a subword unit inventory U given the set of pronunciations Q of words found in I.

The subword unit inventory algorithm utilizes the Minimum Description Length (MDL) principle [9] to search for an inventory of units U which minimize the sum of two terms, $L(Q|U)$ and $L(U)$:

$$\arg\min_{U} \{\lambda L(Q|U) + (1-\lambda)L(U)\} \qquad (1)$$

where $0 \leq \lambda \leq 1$ is chosen by the user to achieve the desired number of subwords M. The MDL principle finds the *smallest model* which also *predicts the training data well*. Smaller models generalize better to unseen data.

The Model Representation Cost $L(U)$ measures the number of bits needed to store the inventory U itself. it is computed over all the units in U from the probability $p(phoneme)$, estimated from the frequency counts of each phoneme in Q:

$$L(U) = \sum_{u \in U} \sum_{phoneme \in u} -\log p(phoneme) \qquad (2)$$

The Model Prediction Cost $L(Q|U)$ measures the bits needed to represent Q with the current inventory U:

$$L(Q|U) = \sum_{q \in Q} \sum_{u \in tokens(q)} -\log p_u \qquad (3)$$

where $tokens(q)$ maps a pronunciation onto a sequence of subword units.

To find the optimal subword inventory U and segmentation $tokens(q)$, we utilize a greedy, top-down, depth-first search algorithm described in [4][5]. Words are split as long as the overall cost decreases, until the desired number of units is reached. Given a new set of words with their pronunciations, the Viterbi algorithm is used to segment each of its pronunciations into subword units from the inventory U, with smallest cost $\sum_{i=1}^{n} -\log p_{u_i}$.

To rewrite a database I in terms of subword units, the words are scanned sequentially. Each word is mapped to a subword unit sequence. If a word has multiple pronunciations, one mapping is chosen randomly. Once a database has been rewritten in terms of subword units, the LM is trained on the rewritten database.

3. Experimental Design

3.1. Dataset Description

The dataset used in this work is the same as the one used by [1]. It is a collection with data from 35,868 songs. Each entry contains a song title, artist name, album name, and the song lyrics. A unique ID is created for each song by merging the song title, artist name, and album name. Figure 1 shows examples for several songs.

1000 songs were selected randomly from the song database, and divided into groups of 50. Twenty subjects (13 males and 7 females) were instructed to listen to 30-second snippets of 50 songs each, and to utter any portion of the lyrics that they heard. Subjects were also prompted to transcribe what they said, which served as reference transcripts (for calculating phone error rates). The song title was also kept.

The ground truth for the IR experiments is the set of songs with the same title as the query song. The song title as a key addresses the retrieval of covers, as well as songs re-recorded by the same artist. An exception table is used, however, to handle cases when songs have different lyrics but similar titles, *e.g. Angel* by *Jimi Hendrix* or by *Dave Matthews Band*. This exception table was built by hand.

In these experiments, we worked with two subsets of the database. The smallest lyric set, ls2000, contains 1989 songs that serve as ground truth to the test set utterances. The largest set, ls36000, contains all the songs.

3.2. ASR

The core architecture of experiments follows the prototypical system shown in Figure 1, and comprises an ASR front-end and an IR back-end.

As ASR front-end, we used the CMU Sphinx-3 ASR system. Sphinx-3 is used to generate the 7-best hypotheses for each spoken query, which are then submitted to the IR back-end for retrieval. The input spoken query is converted into standard MFCC. The acoustic models used by the decoder are triphone HMM, trained from Wall Street Journal data resampled to 8kHz. The word pronunciations are obtained from the CMU dictionary when available, or NIST's `addttp` (G2P tool) when not. Finally, the LMs are trigrams with Witten-Bell smoothing, built using the CMU SLM toolkit. All of these components are available as open source.

The ASR is evaluated based on the *Phone Error Rate (PER)*, the sum of substitutions, insertions, and deletions made by the ASR engine at the phone level. We used PER because we do not have the subword-level references.

3.3. Information Retrieval

The IR back-end uses a vector space model approach for retrieval. Each song document forms a multidimensional feature vector v. The query also forms a vector q in the same feature space. A score $Score(q, v)$ measures the similarity between q and v. The songs with the top 7 scores are submitted for our recall analysis.

After evaluating different feature spaces and scoring methods, we selected as features the counts of unique unigrams, bigrams, and trigrams present in documents and query, which we call *terms*. We selected the score:

$$Score(q, v) = \sum_{\forall t} \delta(t)\text{IDF}(t) \qquad (4)$$

where $t \in \{terms(q) \bigcup terms(v)\}$, $\delta(t)$ is 1 if term t appears in both query and document, 0 otherwise, and IDF(t) is the inverse document frequency of term t. No document length normalization was performed. Similarly to question answering tasks [10], here the documents are too short to accurately estimate the probability distributions of words. Direct matches between words in the query and in the songs, *i.e.* the presence or absence of n-grams, are therefore better measure of similarity than query likelihood.

The baseline system is a *word* system, in which the LM and index are comprised of words as base units. This architecture is compared with three others: a *subword* system, where the LM and the index base units are subwords, a *hybrid* system, where the LM and the index base units are a mix of words and subwords, and a *phone* system, where the LM is comprised of words but the index is phone-based, and the ASR output is converted to phones. The IR accuracy metric is the *k-call-at-n*, where the information need is considered satisfied if *at least k* correct retrievals appear in the top *n*. The *1-call-at-7* measures the percentage of test utterances for which the IR back-end retrieves *at least one* of the ground truth songs in the top 7 results.

3.4. Out of Vocabulary Rates

We simulated a range of OOV rates by pruning the dictionary and language model used by the recognizer or by the MDL algorithm. In the case of words, we built the LM from the set of songs we wanted to index. We simulated an OOV rate by pruning the dictionary based on word frequency computed in the index data. For an OOV rate of $N\%$, we pruned the dictionary so that $N\%$ of the words in the test set are removed, as well as all words less frequent than these. The minimum OOV rate is 5%. As the phone system uses the output from the word system and simply converts it to phones, or equivalently uses the phone-level recognition from a word-based recognition, the OOV rate for the phone system is similar.

In the case of subwords, we used the pruned dictionary as described above for building the subword unit inventory. We mapped `ls2000` (*cf.* Section 3.5) from words to subwords using this inventory. The mapping from words to subwords is induced by the Viterbi algorithm, as described in [5]. `ls2000`, mapped to subwords, was used to create an LM. The subword dictionary trivially maps a subword unit to its constituent phones. The LM and dictionary remained fixed for all recognition experiments regardless of the set of songs to index.

The hybrid system was created from the subword system. Therefore, the LM and dictionary also remain fixed for all recognition experiments. In the list of mappings from words to subwords, we looked for entries where words were mapped to no more than one subword. These words were added to the dictionary with their alternate pronunciations, and they were not changed in the text from which the LM was built. The remaining entries from the word to particle mapping, *i.e.* those cases where a word maps to a sequence of subwords, were treated as in the subword system: words were replaced by their corresponding subword sequence in the text used to create the LM, and each subword was added to the dictionary. This way, both the dictionary and the LM contain a mix of words and subwords.

3.5. Subword Unit Inventory Sizes

In our previous work [4], we found that building the inventory from the smallest set was better than from the largest one, even generalizing better. Here, we use the smallest set, `ls2000`, to build inventories of various sizes up to 4800 units. For a given size and OOV rate, we ran recall experiments using indices of different sizes. We built each index by inducing a mapping from words in the songs to subword units. We assumed that it is much less expensive to generate pronunciations than to build an LM for each index. Therefore, at index-build time, we used a full pronunciation dictionary. All words used to build the IR are induced from the inventory built from `ls2000`.

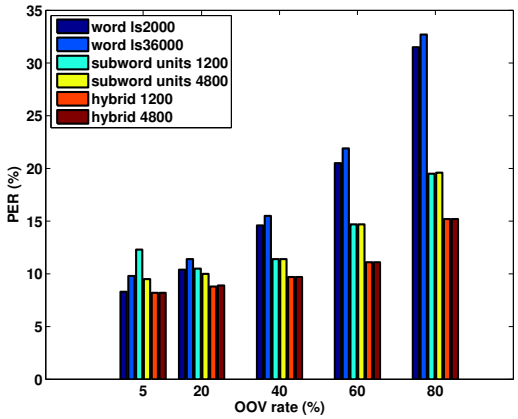

Figure 2: Phone Error Rate as OOV rate changes. *Word* systems built from different subsets of the database, *Subword* systems with various inventory sizes, and the equivalent *Hybrid* .

4. Results and Discussion

Figure 2 shows recognition accuracy (in PER) as a function of OOV rate. We show two word-based systems built from `ls2000` and `ls36000`, the smaller having a more constrained language model. We also show subword-based systems built with different number of units, and the hybrid systems built from each of these. The PER for the phone system is the same as the word system so it is not shown. As expected, the PER degrades much more gracefully for the subword systems as the OOV rates increases. But interestingly, the hybrid system has an accuracy that is as good as the word based for low OOV rate, and it degrades much more gracefully than the subword system as the OOV rate increases. Words with multiple pronunciations end up as different elements in the subword system. Therefore, the LM entries for these words end up being split, hurting recognition. These words may remain untouched in the hybrid system. This explains the better performance of the hybrid system compared to the subword one. The better performance of the hybrid compared to the word one possibly results from the subwords present in the hybrid system.

Figure 3 displays the retrieval performance as a function of OOV rates comparing all four systems, *word*, *subword*, *hybrid*, and *phone*. The figure shows results with the indices built from `ls36000`. The recall for the word system, as expected, degrades as the OOV rate. The recall for the phone system is much lower, even if its PER is the same as the word system, showing it is not a good unit for recall in this type of databases. The recall for the subword system remains at a reasonable level. The really encouraging result is that the hybrid system outperforms the subword one everywhere, reaching almost the same level as the word system at very low OOV rates. This result was achieved by assuming that the LM, used by the ASR system, is fixed, but the pronunciation dictionary, used to induce a subword mapping, can change. This assumption is reasonable in cases where rebuilding an LM

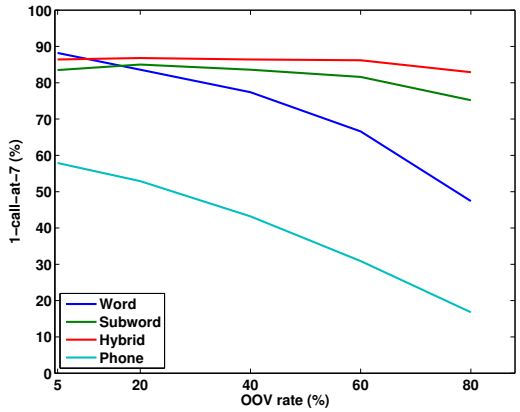

Figure 3: Recall as OOV rate changes for systems with different units. Indices built from `ls36000`. The subword unit inventory has 1200 units. Hybrid built from equivalent subword system. Phone extracted from word recognition results.

can be prohibitively costly, such as embedded systems or very large LMs.

5. Conclusion

Novel words or unexpected spellings can drive the OOV rate to high levels. In these situations, we have demonstrated that a hybrid voice search system outperforms a word-, subword-, or phone-based. A hybrid system brings the best of all worlds. Words with multiple pronunciations are not split as separate entries, as is the case with a pure subword system. Hybrid systems can absorb unknown words via its subset of subword units, making it much more robust to OOVs. A hybrid system also lends itself well to the construction of a LM and dictionary that do not change if the IR index changes.

6. References

[1] G. Choueiter, "Linguistically-motivated sub-word modeling with applications to speech recognition," Ph.D. dissertation, MIT, 2009.

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

[3] G. Zweig and P. Nguyen, "Maximum mutual information multiphone units in direct modeling," in *Proc. Interspeech*, Sep. 2009.

[4] E. Gouvêa, T. Ezzat, and B. Raj, "Subword unit approaches for retrieval by voice," in *SpokenQuery Workshop on Voice Search*, 2010.

[5] E. Gouvêa and T. Ezzat, "Vocabulary independent spoken query: a case for subword units," in *Proc. Interspeech*, Sep. 2010.

[6] R. Rose *et al.*, "Subword-based spoken term detection in audio course lectures," in *Proc. ICASSP*, 2010.

[7] C. Parada, A. Sethy, M. Dredze, and F. Jelinek, "A spoken term detection framework for recovering out-of-vocabulary words using the web," in *Proc. Interspeech*, Sep. 2010.

[8] T. Mishra and S. Bangalore, "Speech-driven query retrieval for question-answering," in *Proc. ICASSP*, 2010.

[9] M. Creutz and K. Lagus, "Unsupervised morpheme segmentation and morphology induction from text corpora using Morfessor 1.0." Helsinki University of Technology, Tech. Rep., Mar. 2005.

[10] V. Murdock and W. B. Croft, "Simple translation models for sentence retrieval in factoid question answering," in *Proc. SIGIR*, 2004.