Using word graphs as intermediate representation of uttered sentences

Jon A. Gómez and Emilio Sanchis

Departament de Sistemes Informàtics i Computació,
Universitat Politècnica de València, Spain
{jon,esanchis}@dsic.upv.es
http://elirf.dsic.upv.es/

Abstract. We present an algorithm for building graphs of words as an intermediate representation of uttered sentences. No language model is used. The input data for the algorithm are the pronunciation lexicon organized as a tree and the sequence of acoustic frames. The transition between consecutive units are considered as additional units.
Nodes represent discrete instants of time, arcs are labelled with words, and a confidence measure is assigned to each detected word, which is computed by using the phonetic probabilities of the subsequence of acoustic frames used for completing the word.
We evaluated the obtained word graphs by searching the path that best matches with the correct sentence and then measuring the word accuracy, i.e. the oracle word accuracy.

Keywords: word graphs, word lattices, lexical tree, confidence measures

1 Introduction

Word graphs are directed acyclic graphs (DAGs) where each arc is labelled with a word and each node is labelled with a discrete time mark. Each arc represents a word detected between two instants of time and contains a confidence measure.

There are a lot of works using weighted word graphs as intermediate representation of uttered sentences in automatic speech recognition (ASR) systems or in spoken language understanding (SLU) systems [1–7]. Some authors use the concept of word lattice and other ones use the concept of word confusion network (WCN). There maybe differences at the implementation level, but the purpose is the same, that is to obtain a compact and efficient representation of the n-best recognized sentences. Then, word-graphs/word-lattices/WCN can be used as the input to modules operating at higher levels of knowledge, for example the understanding module in a spoken dialog (SD) system.

In [1], a bigram language model is used to obtain a word graph from each uttered sentence. The used algorithm is an extension of their one-pass beam search strategy using lexical trees. A variation of the approach presented in [1] is presented in [2]. The difference between both approaches resides in the strategy

used for exploring the search space. Several confidence measures to be used in weighted word graphs are presented and discussed in [3, 4].

A SLU system where the output of the ASR module is a lattice of word hypotheses is described in [5]. Finite state transducers (FST) are used in a translation process for generating hypotheses of conceptual constituents from the lattice of word hypotheses. In [6], word lattices are used to be converted into word confusion networks (WCN), the authors presented the *"pivot"* algorithm for reducing a word lattice and obtaining as a result a WCN. The main idea of this algorithm is to normalize the topology of input graphs.

In [7], it is presented a two-stage strategy for building word graphs: the first stage is "forward–decoding" and the second one is "backward–decoding". The first stage is an interesting extension of the Viterbi algorithm that stores several predecessor words for each word, then the second one explores backward the n-best final states until the initial state is reached. Word graphs are generated as a result of the backward-decoding phase.

Our approach does not use a language model for building word graphs, it explores the sequence of acoustic frames in order to complete phonetic sequences corresponding to words. The pronunciation lexicon with all the possible phonetic sequences of each word is organized as a tree. This way of representing the pronunciation lexicon has been used by many researchers [1]. The confidence measure of each detected word is computed using only the phonetic probabilities estimated by the acoustic models. In other words, our approach for the construction of word graphs only uses the following sources of knowledge: acoustical, phonetic and lexical. The syntactic knowledge provided by language models can be used in a decoder that takes word graphs as starting point for recognizing. In [9], the understanding module combines two sources of knowledge, syntactic and semantic, for detecting semantic units related with the concepts defined as goals in a dialogue system.

The rest of the paper is organized as follows. Section 2 describes the details of the lexical tree used in our approach. Section 3 presents and explains the algorithm for building word graphs. Section 4 describes the experiments and presents the results. Finally conclusions are discussed in Section 5.

2 Lexical tree or *Trie*

Figure 1 shows an example of the lexical tree or *Trie* used in our system for representing the phonetic transcriptions of all the words in the vocabulary. Terminal nodes are grey coloured and labelled with the words which are completed when reaching these nodes. In fact, each terminal node points to a list with all the words that share the same phonetic transcription.

Table 1 presents an example of file containing the words in the vocabulary with their possible phonetic transcriptions. The process of building the *Trie* takes into account the set of phonetic units (phonemes + silence) and the transitions between two consecutive phonetic units. Transitions are considered as additional units and are represented with a plus sign between the labels of two consecutive

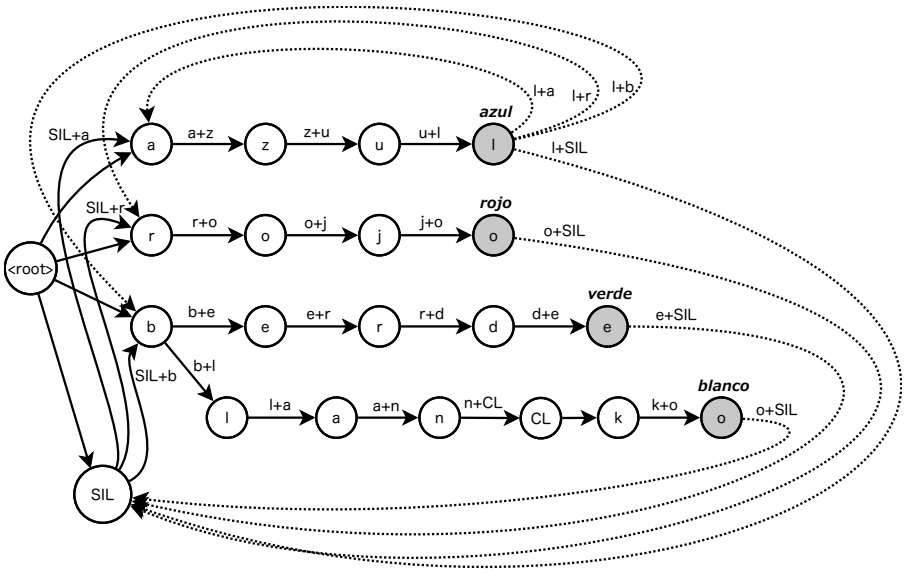

Fig. 1. *Example of lexical tree o Trie for representing the phonetic transcriptions of all the words in the vocabulary. Arcs are labelled with units representing transitions between consecutive phonemes. Dotted arcs are special arcs for connecting terminal nodes with the ones representing word-initial phonemes. For clarity, not all the arcs of this kind are represented, only the arcs leaving from the terminal node corresponding to the word* "azul" *and the ones reaching the silence.*

units. Not all possible transitions are used, only those which appear frequently enough in the set of training sentences are considered.

Nodes represent phonetic units. Arcs are labelled with the transition between two phonetic units when the transition is used, otherwise arcs are λ labelled. An example is shown in Figure 1, where the arc connecting nodes `CL` and `k` of word *"blanco"* is not labelled because the transition `CL+k` is not allowed. `CL` represents the closure before plosive consonants. Root node doesn't represent any unit. `SIL` node represents all possible silences in the recording of an utterance: initial silence, final silence and pauses between words.

As Figure 1 shows, there are special arcs leaving from terminal nodes pointing to initial nodes. These arcs are drawn with dotted lines and their role is taking into account the transition between the ending phoneme of a word and the initial phoneme of the next one. The use of this kind of arcs simplifies the algorithm for building word graphs.

3 Algorithm for building word graphs

Our proposed algorithm for building word graphs will be easy to understand keeping in mind the described *Trie*, and putting special attention to arcs that

Table 1. Example of the list representing all the possible phonetic transcriptions of the words in the vocabulary. Each line contains a pair orthographic transcription – phonetic transcription. The different possible phonetic sequences of a word are in different lines.

Orthographic transcription	Phonetic transcription
azul	a z u l
rojo	r o x o
verde	b e r d e
blanco	b l a n CL k o

connect terminal nodes with nodes representing the word-initial phonemes. This algorithm explores the sequence of acoustic frames that represents each pronounced sentence, and maintains a list of word hypotheses that is updated for each incoming acoustic frame.

A hypothesis is defined as a 5-tuple, $hyp = \{n, a, t_0, t, score\}$, where n is a node in the *Trie*, a is an arc in the *Trie*, a hypothesis can only be located in either a node or an arc, no both. t is the current time, t_0 is the time instant this word hypothesis begins, and *score* is the sum of the logarithms of the phonetic probabilities corresponding to the acoustic frames from t_0 up to t.

Figure 2 shows the block diagram of our ASR system. Only modules related with the word graph builder are represented. Each module runs in an independent execution thread. FIFO queues are used in order to manage the data flow between each pair of connected modules. Due to this, no process becomes locked when sending its output data and the whole system takes advantage of microprocessors with multiple cores.

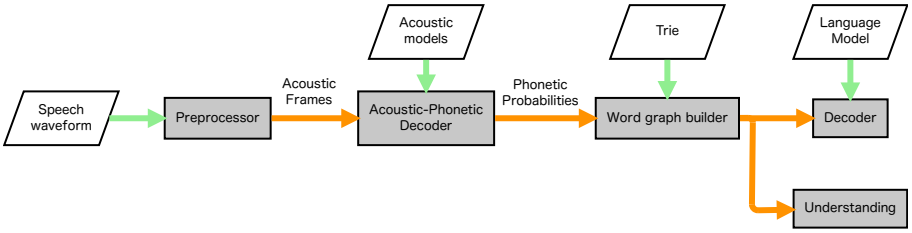

Fig. 2. *Block diagram of our ASR system. Each module runs in an independent execution thread and modules are connected by means of FIFO queues. Word graphs are sent to several modules (understanding, decoder, ...) depending on the selected configuration.*

The acoustic-phonetic decoder converts the sequence of acoustic frames into a sequence of vectors with phonetic probabilities [8]. The word graph builder processes sequentially the stream of vectors with phonetic probabilities. It is connected by means of a FIFO queue with the output of the acoustic-phonetic decoder. The process of building word graphs begins with a unique hypothesis

located at the root node of the *Trie*. The first incoming vector with phonetic probabilities is used to expand this initial word hypothesis in order to reach nodes corresponding to word-initial phonemes. The normal operation of the algorithm consists of expanding hypotheses in the list of hypotheses (LH) for each vector with phonetic probabilities extracted from the FIFO queue. Not all the hypotheses in LH are expanded, a beam search strategy is applied. A word is inserted into the graph every time that one hypothesis reaches a word-initial phoneme (or silence) from a terminal node.

Algorithm 1 Expanding of the list of hypotheses

Input: $logProb$: vector with phonetic probabilities estimated at frame t
Input: t : index frame representing time
Global variables:
$bestScore$: real
$BEAM$: real
LH : list of hypotheses expanded up to frame $t-1$
Local variables:
LH' : temporal list of hypotheses
nbs : real ▷ New best score

$LH' \leftarrow \{\}$
$nbs \leftarrow -\infty$
for all $h \in LH$ **do**
  if $score(h)$ is worse than $(bestScore + BEAM)$ **then**
    do nothing
  else if $node(h)$ is not $null$ **then**
    if $node(h)$ is a terminal node **and** h is in the last state of the acoustic model **then**
      insert into the graph all words completed at $node(h)$
    end if
    $u \leftarrow label(node(h))$ ▷ Hypothesis which continues in the same node
    $h' \leftarrow (node(h), null, t_0(h), t, score(h) + logProb[u])$
    $LH' \leftarrow LH' + \{h'\}$; $nbs \leftarrow \max(nbs, score(h'))$
    for all $a \in arcsFrom(node(h))$ **do**
      if $label(a)$ is not $null$ **then**
        $u \leftarrow label(a)$ ▷ New hypothesis in a labelled arc
        $h' \leftarrow (null, a, t_0(h), t, score(h) + logProb[u])$
      else
        $n' \leftarrow node(a)$
        $u \leftarrow label(n')$ ▷ New hypothesis in a node
        if n' is a word-initial node **then**
          $h' \leftarrow (n', null, t, t, score(h) + logProb[u])$
        else
          $h' \leftarrow (n', null, t_0(h), t, score(h) + logProb[u])$
        end if

```
            end if
            LH' ← LH' + {h'} ; nbs ← max(nbs, score(h'))
        end for
    else                                          ▷ Then arc(h) is not null
        a ← arc(h)
        u ← label(a)                ▷ Hypothesis which continues in the same arc
        h' ← (null, a, t0(h), t, score(h) + logProb[u])
        LH' ← LH' + {h'} ; nbs ← max(nbs, score(h'))

        n' ← node(arc(h))
        u ← label(n')
                    ▷ New hypothesis in the node pointed by the current arc
        if n' is a word-initial node then
            h' ← (n', null, t, t, score(h) + logProb[u])
        else
            h' ← (n', null, t0(h), t, score(h) + logProb[u])
        end if
        LH' ← LH' + {h'} ; nbs ← max(nbs, score(h'))
    end if
end for
LH ← LH' ;
bestScore ← nbs
```

Algorithm 1 shows the pseudo-code corresponding to the body of the loop that processes the stream of vectors with phonetic probabilities. Each iteration of this loop processes one vector for expanding all hypotheses in LH. The operation for inserting new hypotheses in LH' checks if the hypothesis is going to be pruned in the next iteration. A new hypothesis will be rejected if its score is worse than $(nbs + BEAM)$. As pointed out in [6, 7], word graphs are an efficient way of storing a high number of sentence hypotheses, but they can grow hugely if it is not applied a pruning strategy in the building process. Additionally, we use a histogram based pruning strategy for limiting the number of words between each pair of nodes. This value was adjusted empirically to five words in a trade-off between performance and word accuracy.

4 Experimentation

We used two Spanish speech corpora for measuring the quality of the obtained word graphs: *Albayzin* [10] and DIHANA [11].

Albayzin corpus is divided into two subcorpus, one phonetic and one geographical. We used the phonetic subcorpus: 6,800 uttered sentences obtained by making groups from a set of 700 distinct sentences pronounced by 40 different speakers. We used the suggested training/test subdivision.

DIHANA corpus is also divided into two subcorpus. One subcorpus has 3,594 uttered sentences. We used all these uttered sentences for training. The other

subcorpus has 6,277 uttered sentences corresponding to 900 human-machine dialogues regarding information about train timetables, fares, destinations and services. 4,928 uttered sentences were used for training and 1,349 for testing. Working with this corpus is a difficult task due to the spontaneity of the speech.

4.1 Evaluation Results

For evaluation purposes, it was extracted from each word graph the sequence of words corresponding to the path which best matches the reference sentence. The criterion used for evaluating the quality of word graphs consists of measuring the oracle word accuracy O-WA $= 100 \times \frac{Correct}{Correct+Insertions+Substitutions+Deletions}$.

Another parameter of interest in relation with word graphs is the branching factor, i.e. the average number of arcs leaving from nodes.

Table 2. Oracle word accuracy and branching factor estimated for both Spanish corpora *Albayzin* and DIHANA, and word accuracy for DIHANA obtained in [7].

Corpus	Oracle WA	Branching factor	Oracle WA in [7]
Albayzin	98.8%	367	
DIHANA	88.8%	189	86.9%

Table 2 shows the results obtained by evaluating the graphs generated by using our ASR system. Our algorithm has a high rate for detecting words, as the obtained results confirm. We have to point out that DIHANA corpus is composed by spontaneous speech dialogues acquired by telephone. It is considered a very difficult task. As a reference, the O-WA obtained in [7] when working with DIHANA corpus and using a language model for building graphs is also shown in Table 2.

The values of the branching factor may seem too high, specially in the case of *Albayzin* corpus, but it is important to point out that the obtained word graphs have a lot of arcs labelled with the same word. This happens because we allow several alternatives for each detected word, each alternative begins and ends at different nodes. This feature allows to simplify the algorithms to be used for working with these graphs.

5 Conclusions

We have presented an algorithm for building word graphs as an intermediate representation of uttered sentences. The main difference with respect to previous works is that our algorithm does not use a language model, therefore, the strategy for detecting words is quite different. The rules used for deciding whether a word has to be inserted into the graph are also different. Furthermore, words are inserted into the graph just when their phonetic sequence is completed, in a way different from other approaches, that generate the graphs in a backward process

once the end of the sentence is reached. Usually, these other algorithms build the graph from the lattice obtained in the Viterbi decoding process or from the n-best recognized sentences.

The obtained oracle word accuracy shows that our algorithm has a high capacity for detecting words, but there are some aspects we have to improve in order to obtain less dense graphs.

Acknowledgments

This work was supported by the Spanish MICINN under contract TIN2011-28169-C05-01 and the Vic. d'Investigació of the UPV under contract 20110897.

References

1. Ortmanns, S., Ney, H. and Aubert, X.: A word graph algorithm for large vocabulary continuous speech recognition. In Computer Speech and Language (1997), vol. 11, pp. 43–72.
2. Ney, H., Ortmanns, S. and Lindam, I.: Extensions to the word graph method for large vocabulary continuous speech recognition. In proceedings of IEEE ICASSP-97. vol. 3, pp. 1791–1794, Munich, Germany (1997).
3. Wessel, F., Schlüter, R., Macherey, K. and Ney, H.: Confidence Measures for Large Vocabulary Continuous Speech Recognition. In IEEE Transactions on Speech and Audio Processing, vol. 9, no. 3, march 2001. pp. 288–298.
4. Ferreiros, J., Segundo, R. San, Fernández, F, D'Haro, L.-F., Sama, V., Barra, R. and Mellén, P.: New word-level and sentence-level confidence scoring using graph theory calculus and its evaluation on speech understanding. In proceedings of INTERSPEECH-2005, pp. 3377–3380, Lisbon, Portugal (2005).
5. Raymond, C., Béchet, F., De Mori, R. and Damnati, G.: On the use of finite state transducers for semantic interpretation. In Speech Communication (2006), vol. 48, pp. 288–304.
6. Hakkani-Tür, D., Béchet, F., Riccardi, G. and Tur, G.: Beyond ASR 1-best: Using word confusion networks in spoken language understanding. In Computer Speech and Language (2006), vol. 20, pp. 495–514.
7. Justo, R., Pérez, A. and Torres, M.I.: Impact of the Approaches Involved on Word-Graph Derivation from the ASR System. In proceedings of IbPRIA 2011, LNCS, vol. 6669, pp. 668–675, Springer-Verlag, Berlin, Heidelberg (2011).
8. Gómez, Jon A. and Calvo, Marcos: Improvements on Automatic Speech Segmentation at the Phonetic Level. In proceedings of CIARP 2011, LNCS, vol. 7042, pp. 557–564, Springer-Verlag, Pucón, Chile, November 15–18 (2011).
9. Calvo, M., Gómez, J.A., Sanchis, E. and Hurtado, L.F.: An algorithm for automatic speech understanding over word graphs. In Procesamiento del Lenguaje Natural (2012), no. 48, pp. (accepted, pending of publication).
10. Moreno, A., Poch, D., Bonafonte, A., Lleida, E., Llisterri, J., Mariño, J. B. and Nadeu, C.: Albayzin Speech Database: Design of the Phonetic Corpus. In: Proceedings of Eurospeech, 1993, volume 1, pages 653–656. Berlin (Germany), September 1993.
11. Benedí, J.M., Lleida, E., Varona, A., Castro, M., Galiano, I., Justo, R., López, I., Miguel, A.: Design and acquisition of a telephone spontaneous speech dialogue corpus in Spanish: DIHANA. In: Proc. of LREC 2006, Genova, Italy (2006).