An approach to obtain weighted graphs of words based on phoneme detection

Jon Ander Gómez, María José Castro and Emilio Sanchis
Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia, Valencia (Spain)
{jon,mcastro,esanchis}@dsic.upv.es

Abstract
In this paper, we present an approach for phoneme detection and phonetic classification that can be used as a basis for different speech processes, such as phoneme boundary detection, acoustic-phonetic decoding or word-graph construction with acoustic confidence scores.

The phonetic classifier that has been developed is based on a phase of acoustic vector clustering in the space of acoustic characteristics, and on a second phase for the association of the acoustic classes with the phonetic units by means of conditional probabilities.

We also present methods to build graphs of linguistic units (phonemes, words or semantic units). Phoneme graphs can be applied to acoustic-phonetic decoding tasks or as a preceding step to obtain word graphs. Word graphs can be used in recognition tasks, or they can be converted into graphs of semantic units to be used for understanding tasks. Some recognition experiments and understanding experiments are also presented for a restricted-semantic task about geographical queries.

1 Introduction
In this paper, we present an Automatic Speech Recognition (ASR) system which uses graphs of linguistic units to represent utterances. These graphs can be of phonemes, words or semantic units depending on the knowledge level. Graphs have the capacity to reflect uncertainty by allowing several hypotheses, and to assign a confidence score for each detected linguistic unit. The confidence score is calculated from phonetic probabilities.

Our ASR system is a decoupled bottom-up system based on distinct modules that are connected serially. These modules operate sequentially, so a good way to represent utterances to transfer information from one module to another is needed. We use graphs of linguistic units to achieve this end. There are some works that also use graphs to represent utterances [1, 2, 3].

Word graphs are the crucial point of our ASR system; in fact, phoneme graphs are intermediate representations to construct them. Phoneme graphs are constructed by detecting phonemes in an utterance based on the phonetic probabilities provided by a phonetic classifier, which computes the probability of each phonetic unit at every acoustic frame. The word-graph construction process consists of exploring the phoneme graph in order to complete words that belong to the vocabulary. Every time a word is completed, a new node is created and inserted into the word graph.

Word graphs can be converted into graphs of semantic units in a way similar to the way they are constructed from phoneme graphs. In this case, a vocabulary of semantic units is used.

Recognition and understanding processes are performed using the same algorithm, which finds the most likely path on a graph. In recognition mode it uses the word graphs and a language model based on words; in understanding mode it uses the graphs of semantic units and a language model based on semantic units.

Next, in Section 2, we present an overview of our ASR system. Some details about the phonetic classifier and the phoneme graphs are described in Sections 3 and 4, respectively. In Section 5, we describe the word-graph construction process. In Section 6, the results of several experiments are presented: first, the ones obtained by measuring the capacity of the system to detect words and phonemes; second, the ones obtained by performing recognition tasks; and last, the ones obtained by performing understanding tasks. Finally, the conclusions are presented in Section 7.

2 System Description
A basic overview of our ASR system is shown in Figure 1. The first step is carried out by the preprocessor, which transforms the speech waveform into a sequence of acoustic vectors. The phonetic clas-
The phoneme-graph constructor obtains a phoneme graph as a representation of an utterance. This module creates nodes to be inserted into the phoneme graph. These nodes represent phonetic units detected in a delimited time interval. The detection of each phonetic unit is performed separately by searching for subsequences of acoustic frames where the phonetic unit is considered to be probable enough. Two consecutive nodes are connected by an arc in the time direction if they fulfill a set of restrictions [5].

When a phoneme graph is available, the word-graph constructor builds a word graph based on the exploration of the phoneme graph to complete words that belong to the vocabulary. The vocabulary is represented by a tree, where arcs are labeled with phonetic units and terminal nodes are labeled by the orthographic transcription of the words. By exploring the phoneme graph, we can make phonetic sequences which are used to find words in the vocabulary tree.

Once a word graph is available, the recognizer finds the most likely path between the initial node and the final one. Then, the recognized sentence proposed by the system is made by concatenating the words that label the nodes of the most likely path. The criterion used to find the most likely path properly combines the confidence scores of words with the probabilities provided by a language model.

Furthermore, we can also consider the whole word graph as a way to supply multiple recognition hypotheses to other modules, such as understanding or dialog modules.

If we use our system for understanding purposes, word graphs are converted into graphs of semantic units by using the same strategy employed to obtain word graphs from phoneme graphs. In this case, a vocabulary of semantic units is used, where arcs are labeled with words and terminal nodes are labeled with semantic units. The most likely path in graphs of semantic units is found with the same algorithm used in recognition mode; obviously, the language model used here is based on semantic units.

3 Acoustic-Phonetic Classification

In this section, we present the phonetic classification which is based on the unsupervised learning of acoustic classes and their association to phonemes by means of conditional probabilities. This phonetic classification is used in our ASR system [4].

The phonetic classifier computes the a posteriori probabilities of each phonetic unit $p_{th}$ given an acoustic frame $x_t$, $Pr(p_{th}|x_t)$, by combining the probabilities of acoustic classes, which are estimated sifiers, such as neural networks.
from a clustering procedure on the acoustic feature space, and the conditional probabilities of each acoustic class with respect to each phonetic unit.

Clustering of acoustic classes. It is assumed that acoustic classes can be modeled by means of Gaussian distributions. Parameters of each Gaussian distribution are estimated by using the unsupervised version of the Maximum Likelihood Estimation procedure [6]. Given a number of acoustic classes \( C \), it is possible to estimate the probability of each acoustic class \( w_c \) given an acoustic vector \( x_t \), \( \Pr(w_c|x_t) \), from the mixture of Gaussian distributions. Nevertheless, as we need the probability of each phonetic unit \( ph_f \) given an acoustic vector \( x_t \), \( \Pr(ph_f|x_t) \), a set of conditional probabilities is estimated in order to calculate the phonetic probabilities from the acoustic ones.

Association of acoustic classes with phonetic units. The use of conditional probabilities allows us to compute the phonetic-conditional probability densities \( p(x_t|ph_f) \) as follows [4, 5]:

\[
p(x_t|ph_f) = \sum_{c=1}^{C} p(x_t|w_c) \cdot \Pr(w_c|ph_f) \quad (1)
\]

where \( C \) is the number of acoustic classes, \( p(x_t|w_c) \) is the acoustic class-conditional probability density, computed as the Gaussian probability density function, and \( \Pr(w_c|ph_f) \) is the conditional probability that acoustic class \( w_c \) has been manifested when phonetic unit \( ph_f \) has been uttered.

Then, applying the Bayes rule, we obtain the phonetic probabilities as:

\[
\Pr(ph_f|x_t) = \frac{\sum_{c=1}^{C} p(x_t|w_c) \cdot \Pr(w_c|ph_f)}{\sum_{j=1}^{F} \left( \sum_{c=1}^{C} p(x_t|w_c) \cdot \Pr(w_c|ph_j) \right)} \quad (2)
\]

where \( F \) is the number of phonetic units.

4 Phoneme Graphs

Phoneme graphs are a more complex representation of utterances than sequences of phonetic probabilities. In graphs of this kind, nodes represent the detected phonetic units, and arcs connect those phonetic units which are close in time, specifying what nodes are accessible from another one [5]. Phonetic units are detected separately by searching time intervals where each one is considered to be probable enough.

Phoneme graphs are the lowest-level representation of utterances in terms of linguistic units, where several alternative paths are represented. Furthermore, an important aspect to be pointed out is that each phoneme is weighted by a confidence score computed from the phonetic probabilities.

The information that is related to detected phonetic units is stored in the nodes of the graph. Let \( n \) be a node of a phoneme graph; then, we define the following information items:

- \( ph(n) \), the phonetic unit of \( n \)
- \( t_i(n) \), the frame where the segment of \( ph(n) \) begins
- \( t_f(n) \), the frame where the segment of \( ph(n) \) ends
- \( score_{ph}(n) \), the confidence score that \( ph(n) \) was uttered at interval \([t_i(n), t_f(n)]\) calculated under acoustic-phonetic criteria as follows:

\[
score_{ph}(n) = \sum_{t=t_i(n)}^{t_f(n)} - \log \Pr(ph(n)|x_t) \quad (3)
\]

where \( \Pr(ph(n)|x_t) \) is the a posteriori probability to be pronounced \( ph(n) \) when \( x_t \) appears, calculated as equation (2).

When the nodes representing phonetic units detected are connected by arcs, the best boundary between two consecutive nodes is computed in order to maximize the probability of the sequence. Then, an adjustment for each arc is computed taking this boundary into account. The role of the adjustment is to make sure that any path in the phoneme graph takes into account the entire sequence of acoustic frames, and to make sure that each frame is computed only once [5].

5 Word Graphs

The phoneme graph and the vocabulary are used to build the word graph. Basically, the word-graph constructor builds a word graph by exploring a phoneme graph in an attempt to complete words belonging to the vocabulary. Once a word is completed, a new node representing that word is created and it is added to the word graph.

The vocabulary is represented by a tree which is generated from a list of pairs \{phonetic transcription, orthographic transcription\} for each word belonging to the vocabulary. The phonetic sequence of each word is automatically obtained from its orthographic transcription using a grapheme-to-phoneme converter [7].

Figure 2 presents a simple example. The nodes with the words “la” and “casa” are created from
the phonetic sequence “l-a-k-a-s-a” because these words appear in the vocabulary.

**Deletions, insertions and substitution errors** of phonetic units are allowed in the word-graph construction process in order to repair the acoustic-level errors that appear in phoneme graphs. These operations enrich word graphs by adding word hypotheses.

Figure 2 also presents an example of a substitution error: the node labeled with the word “casa” is also inserted into the word graph from the phonetic sequence “l-a-k-a-s-a”. In this case, a path in the vocabulary which needs the phoneme “o” is allowed to be continued when the phoneme “a" appears in the phoneme graph.

Each word added to the word graph is obtained by completing words of the vocabulary according to the successors of the phoneme graph. Nodes are processed according to a branch-and-bound strategy. It manages a list of active nodes of the phoneme graph and inserts into this list as they are reached by new hypotheses. The nodes are processed according to time criteria.

A hypothesis represents an incomplete word and is composed by the pair \((n, v)\), where \(n\) is a node of the phoneme graph and \(v\) is a node of the tree that represents the vocabulary. New hypotheses are created by expanding towards the successors of \(n\) in the phoneme graph according to the successors of \(v\) in the vocabulary.

We now summarize the actions to be carried out to expand each hypothesis \((n, v)\). We use \(n'\) to denote a successor of \(n\), and \(v'\) to denote a successor of \(v\).

1) **Coincidences \((n, v) \rightarrow (n', v')\)**

A new hypothesis is created for each coincidence among the successors of \(n\) and the successors of \(v\). A coincidence occurs when the phonetic unit of \(n'\) and the phonetic unit of \(v'\) are the same.

2) **Substitutions \((n, v) \rightarrow (n', v')\)**

A new hypothesis is created for each successor of \(v\). This hypothesis is the one obtained by exploring all the possible successors of \(n\) that do not match the phonetic unit of \(v'\). The substitution selected is the one with the best confidence score.

3) **Insertions \((n, v) \rightarrow (n', v)\)**

An insertion is the operation that advances towards one of the successors of \(n\) in the phoneme graph and stays on the same node \(v\) of the vocabulary. This operation is done under restrictions: it must be followed by a coincidence or the phonetic unit of \(v'\) is a silence.

4) **Deletions \((n, v) \rightarrow (n, v')\)**

Deletions are also allowed to generate new hypotheses, but only when the vocabulary node \(v'\) has not been achieved by a coincidence operation. Furthermore, deletions are restricted to the following conditions: they must be followed by a coincidence operation, or they are the last step to complete a word.
A new node representing a word is added to the word graph every time a word belonging to the vocabulary is completed while the phoneme graph is being explored. Then, the confidence score of the completed word, which is calculated by using equation (4), is stored in the node. The word hypotheses are unfinished words whose confidence scores are also computed by using the same equation.

To avoid the exponential proliferation of hypotheses, a pruning criterion is used to reject those hypotheses with the worst confidence scores. There is a list of hypotheses in each node of the phoneme graph: when a new hypothesis \((n', v')\) is created, it is inserted to the list of \(n'\). The list of each node is ordered by the normalized confidence score and is pruned if it exceeds a size limit. This limit is a heuristic that is empirically adjusted to trade off between recognition accuracy and response time.

The normalized confidence score of a word hypothesis is calculated by dividing its confidence score by the number of frames it comprises.

### 6 Experiments and Results

In this section, we present results obtained by several experiments. First, we present the ones that were obtained in automatic segmentation tasks. Next, we present the ones that were obtained measuring the capacity of our system to detect phonemes and words. In the third subsection, we present the ones that were obtained measuring the capacity of our system to detect phonemes throughout an utterance.

In all the experiments, one partition was used for training and the other was used for testing. In the case of the PHONETIC subcorpus, the training represents 85% of the corpus and the test 15%. In the case of the BDGEO subcorpus the training represents 65% and the test 35%.

#### 6.1 Automatic Segmentation

Automatic segmentation of speech databases can be performed easily if the phonetic sequence that was pronounced for each sentence is known. A DTW algorithm is used in order to align the frame sequence with the phonetic transcription using the phonetic probabilities mentioned above [4, 5].

Table 1 shows the results of automatic segmentation. The same segmentation task was carried out using the HTK toolkit [10] with the “flat-start” option activated. It can be observed that our automatic system performed slightly better than HTK.

#### 6.2 Word and phoneme detection

Evaluating phoneme detection is carried out by computing the Levenshtein distance between the correct phonetic sequence and its closest derivation in the phoneme graph. The calculation of the Levenshtein distance produces the number of coincidences (C), insertions (I), deletions (D) and substitutions (S) as output [11]. Then, the phoneme detection rate is defined as \(\frac{C}{C+I+D+S}\).

The phoneme-graph construction process uses two empirically adjusted thresholds: \(U_d\) and \(U_a\). The first one is used to detect phonemes throughout an utterance. The second one is used to fix the boundaries once each phoneme has been detected.

<table>
<thead>
<tr>
<th>ALBAYZN</th>
<th>PF</th>
<th>PB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10 ms</td>
<td>20 ms</td>
</tr>
<tr>
<td>Automatic</td>
<td>83.8</td>
<td>75.1</td>
</tr>
<tr>
<td>HTK</td>
<td>82.8</td>
<td>72.9</td>
</tr>
</tbody>
</table>

Table 1: Percentage of frames (PF) matching both segmentations (manual and automatic), and percentage of correctly fixed boundaries (PB) with tolerance intervals expressed in milliseconds obtained with the phonetic subcorpus of the ALBAYZN speech database.

Figure 3: Phoneme detection rate using the phonetic subcorpus of the ALBAYZN database.
Table 2: Word detection rate using the BDGEO subcorpus of ALBAYZIN database.

<table>
<thead>
<tr>
<th>Word-acceptance threshold</th>
<th>Word detection rate</th>
<th>Response time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0%</td>
<td>99.6%</td>
<td>4.3</td>
</tr>
<tr>
<td>2.5%</td>
<td>99.0%</td>
<td>3.4</td>
</tr>
<tr>
<td>5.0%</td>
<td>97.7%</td>
<td>2.4</td>
</tr>
<tr>
<td>7.5%</td>
<td>96.6%</td>
<td>1.8</td>
</tr>
<tr>
<td>10.0%</td>
<td>93.3%</td>
<td>1.6</td>
</tr>
<tr>
<td>15.0%</td>
<td>84.1%</td>
<td>1.4</td>
</tr>
</tbody>
</table>

Figure 3 shows the phoneme detection rate for various combinations of detection and extension thresholds. There is one aspect to be taken into account in order to select the best combination: the insertion of phonemes. Small improvements in the detection rate imply a significant increase in the number of phonemes that are erroneously detected. A balanced combination is the one which uses $U_d = 0.08$ and $U_a = 0.03$. The phoneme detection rate for this combination is around 94%.

On the other hand, evaluating word detection is carried out in a way similar to the one that was done in phoneme detection evaluation. The correct sentence is searched in the word graph to obtain its closest derivation. Then, the same process is again used to obtain the word detection rate. In this case, the Levenshtein distance is obtained between sequences of words, the rate is defined by the same formula, and the counters mentioned above refer to coincidences, insertions, deletions and substitutions at word level.

Completed words in the word-graph construction process are filtered previously the creation of new nodes to be inserted into the word graph. This filter consists of a threshold used to reject those completed words whose normalized confidence scores are worse than the threshold.

Table 2 shows the word detection rate for different values of the word-acceptance threshold. In this table, the word-acceptance threshold is expressed as a percentage in terms of probability. Table 2 also shows the effect of the word-acceptance threshold on system-response time. In the case that acceptance threshold is 5%, it means that a sentence that is one-second in length is processed in 1.4 seconds, so the output of the system is obtained 2.4 seconds after the sentence began to be pronounced.

6.3 Recognition results

When our system operates in recognition mode, the most likely path in word graphs is searched, and then, the recognized sentence proposed by the system is formed by concatenating the words that label the nodes of this path.

Table 3: Word Accuracy (WA) obtained when our system operates in recognition mode using the BDGEO subcorpus of the ALBAYZIN database.

<table>
<thead>
<tr>
<th>Language model</th>
<th>WA</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-grams</td>
<td>84.6%</td>
</tr>
<tr>
<td>3-grams</td>
<td>89.2%</td>
</tr>
<tr>
<td>4-grams</td>
<td>89.5%</td>
</tr>
<tr>
<td>5-grams</td>
<td>89.3%</td>
</tr>
</tbody>
</table>

Finding the most likely path in word graphs is carried out by a branch-and-bound algorithm, which takes into account the probability that a word is preceded by a sequence of $n$ words provided by a language model. The output of this algorithm is a list containing the $N$-best paths.

The language model used was the $n$-gram model, which was computed using the CMU-Cambridge toolkit [12]. The Witten Bell discounting back-off technique was used to permit unseen events.

In order to evaluate our system in recognition mode, we use the Word Accuracy (WA) defined as the Word detection rate ($\frac{C}{C+I+D+S}$); however, in this case the Levenshtein distance is computed between the correct sentence and the recognized one.

Table 3 shows the Word Accuracy (WA) obtained when our system operates in recognition mode for several complexity levels of the language model. Our best result was a WA of 89.5%, which can be compared with the best one obtained by a system based on Hidden Markov Models (HMM) [13]. The best result of this system with the same experiment was a WA of 91.4%.

Figure 4 shows the Correct-Sentence Rate (CSR) obtained by testing whether the correct sentence was among the $N$-best ones. A significant improvement...
6.4 Understanding results

In order to test our system in understanding mode, we have used the MINIGEO subtask, which is defined as a subset of the BEDGE subcorpus of the ALBAYZIN database [14]. This subtask was based on three objects: river, sea and city, and on some relations among them: belongs to, washes, etc.

As mentioned above, when our system works in understanding mode, word graphs are converted to graphs of semantic units. Then, the most likely path in these graphs is found in order to obtain a sequence of semantic units, which can be used to make database queries.

The best way to represent queries is to use a formal language such as SQL [15]. However, in order to simplify the problem, we used an intermediate semantic language which allows us to sequentially translate the sentences into sequences of semantic units [14].

Table 4 shows the Semantic Unit Accuracy (SUA), which is equivalent to the Word Accuracy used to evaluate the system when it operates in recognition mode. It can be observed that these results are significantly lower than the ones obtained in recognition mode. Nevertheless, we must point out that the understanding module has only recently been developed, and that no error model was used in the construction of the graphs of semantic units.

![Figure 5: Correct-Sentence Rate searching for the correct sentence among the N-best ones in the MINIGEO task.](image)

7 Conclusions

In this paper, we have presented several results at different knowledge levels (acoustic, phonetic, syntactic and semantic). The role of our work is to obtain graphs of linguistic units (phonemes, words or semantic units) as representations of utterances, where each linguistic unit is weighted with a confidence score.

As has been presented, to build these graphs we need a phonetic classifier which provides the probability of each phonetic unit given an acoustic vector or frame. Then, a sequence of acoustic vectors, which is the output of the preprocessor, is converted by the phonetic classifier into a sequence of vectors with phonetic probabilities.

For any given sentence, the phonetic boundaries can be fixed using the correct phonetic sequence and the sequence of vectors with phonetic probabilities. It is also worth noting that our approach can be used satisfactorily to automatically segment speech databases for several purposes, such as training other acoustic models or preparing a database of phonetic segments to be used in speech synthesis [16, 17].

Nevertheless, the sequence of phonetic probabilities is specially suitable for detecting segments where a phoneme is considered to be probable enough. Then, a phoneme graph is obtained as a representation of an utterance.

The purpose of phoneme graphs is to be the starting point for building word graphs. By analyzing phoneme graphs, we evaluate the capacity of the system for detecting phonemes. The results obtained are good from the point of view of phoneme detection, but there is one important aspect that must be taken into account. This is the high number of inserted alternatives, which shows that the phoneme detection is not as precise as it should be for the word-graph construction process.

Once a phoneme graph is available as a representation of an utterance, a word graph is constructed in an attempt to complete words belonging to the vocabulary. The capacity of the system to detect

<table>
<thead>
<tr>
<th>Language model</th>
<th>SUA</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-grams</td>
<td>79.1%</td>
</tr>
<tr>
<td>3-grams</td>
<td>81.1%</td>
</tr>
<tr>
<td>4-grams</td>
<td>81.1%</td>
</tr>
<tr>
<td>5-grams</td>
<td>81.2%</td>
</tr>
<tr>
<td>6-grams</td>
<td>81.5%</td>
</tr>
</tbody>
</table>

Table 4: Semantic Unit Accuracy (SUA) obtained when our system operates in understanding mode applied to MINIGEO task.
words is evaluated by looking for the correct sentence in the word graph. We consider this capacity to be good (around 96% of words are detected); however, in our system too many alternative words were inserted into the word graph which were not really pronounced.

Results at the recognition and understanding levels have been presented in order to show the usefulness of word graphs. However, it must also be observed that the Correct-Sentence Rate improves significantly when the correct sentence is searched for taking into account the $N$-best paths for values of $N$ greater than 5. This is confirmation that the capacity of our system to detect words or semantic units is high, but it also confirms that the insertion of linguistic units must be reduced.

Finally, we can say that the methods and the techniques presented here can be used for several purposes in speech recognition tasks, and that our decoupled ASR system obtains results that can be compared with the ones obtained by HMM-based systems [13].

As immediate future work, we plan to improve the error model used in the word-graph construction process, and we will try to reduce the uncertainty at the phonetic level. This last item implies the use of other phonetic classifiers; for example, one based on supervised learning which uses manually segmented and labeled sentences.

References


