S10.16

An unanchored matching algorithm for lexical access.

Alexander I. Rudnicky Zong-ge Li Joseph H. Polifroni Eric Thayer
Julia L. Gale

Department of Computer Science, Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213 USA

Abstract

This paper describes the lexical access component of the CMU continuous speech recognition system. The word recognition algorithm operates in a left to right fashion, building words as it traverses an input network. Search is initiated at each node in the input network. The score assigned to a word is a function of both arc phone probabilities assigned by the acoustic phonetic module and knowledge of expected phone duration and frequecy of occurence of different word pronunciations. The algorithm also incorporates knowledge-based strategies to control the number of hypotheses generated by the matcher. These strategies use criteria external to the search. Performance charactristics are reported using a 1029 word lexicon built automatically from standard pronunciation baseforms by context-dependent phonetic rules. Lexical rules are independent of specific lexicons and are derived by examination of transcribed speech data. The lexical representation now includes juncture rules that model specific inter-word phenomena. A junction validation module is also described, whose task is to evaluate the connectivity of words in the word hypothesis lattice.

1 Overview of CMU speech recognition system

The CMU system is designed to perform speaker-independent, continuous speech recognition. Its design is modular, and includes three major components: *Acoustic-Phonetics*, *Word Hypothesization*, and *Parsing*. The organization is that of a data-flow machine: low-level hypotheses are fed to higher levels of the system, with no feedback from higher levels to lower levels. The purpose of the system is to serve as a framework for the study of issues at the three majors levels of analysis represented by the modules.

The Acoustic-Phonetic level produces a symbolic representation of the speech signal, in the form of a phonetic network, representing alternate segmentation and labeling of the speech signal. A more detailed description of the Acoustic-Phonetic module is given in [1].

The Word Hypothesizer traverses the acoustic-phonetic network, matching it against a stored representation of the task vocabulary, the *lexicon*, producing a *lattice* of word hypotheses. The Word Hypothesizer is described in greater detail in the next section. A sample output, showing acoustic-phonetic and word-hypothesizer levels is shown in Figure 1

The final stage of recognition is performed by the Parser that traverses the word lattice provided by the Word Hypothesizer and produce one or more strings that span the input utterance. The Parser makes use of two constraints in constructing spanning strings: acoustic/phonetic information about the legality of word junctures and grammatical information about the likelihood of word sequences. Juncture information is always used, but the CMU parser can operate either with grammars or without. The Parser module is described in greater detail in [3].

2 The Word Hypothesizer

Word recognition has two essential components: The search algorithm and the representation. A feature of the CMU system is the use of a completely automatic lexicon generation approach. The lexical representation is generated from a a set of baseforms and a collection of ordered phonological rules. The reason for this approach is two-fold: Automatic generation greatly simplifies movement between tasks, particularly those that involve large vocabularies (1000 or more words). Second, it forces the development of an explicit model of lexical phenomena, something that is lacking in purely statistical approaches to recognition.

2.1 Lexicon Generation

A lexicon is created in steps from phonemic baseforms and phonological rules. In the first step, a baseform is selected for each word in a task vocabulary. Thus, for the Resource Management task, nominally consisting of 1000 words, a set of 1029 baseforms were created. (The additional entries are accounted for by the inclusion, as separate lexicon entries, of variants such as `[d ey t ax]` and `[d ae t ax]` for the word DATA, as well as compound forms for certain function word combinations, e.g., IN–THE.) A baseform consists of a single standardized pronunciation, and includes stress information. The (linear) baseforms are transformed into networks specifying alternate pronunciations using an ordered set of phonological rules. Each of the rules in the lexicon accounts for a specific phonological process and is sensitive to context and stress. Rules also specify penalties for certain alternatives based on the frequency of occurrence of the phonological process that the rules encodes, to penalize less-frequent variants. Rules are coded and penalties assessed after examination of transcription data.

Phonological rules operate on phonetic segments, inserting alternative paths into individual word networks. All are modeled using transcription data, as well as general knowledge of phonological processes in English. The majority of lexical rules deal with vowels, liquids and nasals. Approximately half of the rules for sonorants are deletion rules, with about 40% performing transformations and another 10% inserting sonorant segments into the networks. These statistics are consistent with our data and are dependent on such phenomena as speech rate and dialect. Deletions are especially common in sonorant stretches within words where entire syllables may be deleted. For example, in 7 of 22 instances of the word CARRIER in our database, the penultimate syllable (`[-iy-]`) was deleted. Substitutions within the lexicon account most often for dialectal variation. The talkers in our database represent seven major dialect regions of American English. Vowels among these speakers vary greatly but all the variations are rule governed (e.g., raising of mid-front vowels to high-front in Southern American English: `[p eh n]` ⇒ `[p ih n]`) and are easily accounted for in the lexicon.

Among the rules that affect stops, fricatives, and closures, the largest group (approximately 40%) deal with deletions. Roughly one-third of

CH2561-9/88/0000-0469 $1.00 © 1988 IEEE

Authorized licensed use limited to: Carnegie Mellon Libraries. Downloaded on December 11, 2009 at 11:35 from IEEE Xplore. Restrictions apply.

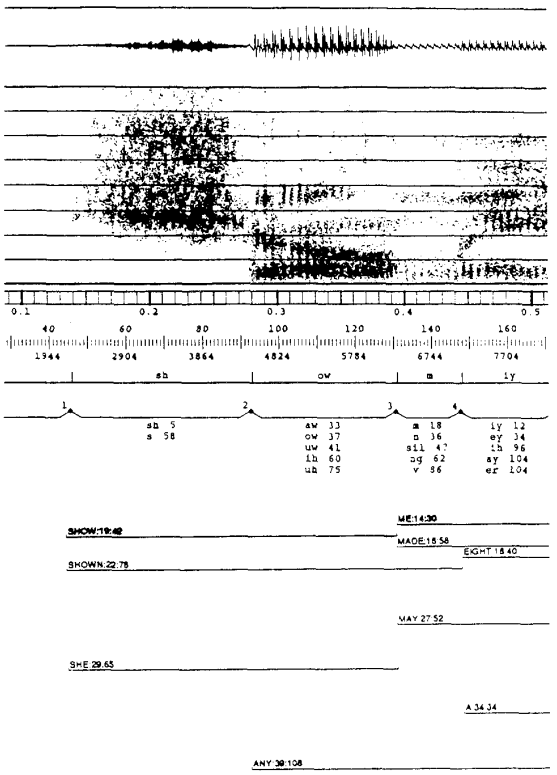

Figure 1: Example of system output.

these rules deal with transformations such as voicing, while the remainder deal with epenthetic insertions. Although the rules that deal with consonantal segments and closures also account for dialectal variation, the majority of these rules address system characteristics, i.e., they model the output of the acoustic-phonetic labeler. For example, certain speech events, such as stop bursts preceding fricatives, are difficult to detect; their absence must be modelled in the lexicon. Labeling errors modelled by the lexicon are commonly due to insufficient training at the acoustic-phonetic level or to non-optimal signal processing.

Individual word networks are created by rule for each of the 1029 words. Juncture rules then apply and model such phenomena as flapping or closure deletions across word boundaries. The word networks are then merged into a single network, collapsing common initial subsequences. (This step is taken for purely computational reasons, to eliminate duplicate search. We are also working with unmerged networks, as those may allow more detailed modeling of intra-word phenomena.) In the merged network presently being using for the 1029-word Resource Management task, there are 9957 nodes and 44514 arcs. Note that self-transitions, if desired, can be generated implicitly during matching, and are not present in the representation itself.

2.2 Matcher Algorithm

Graph search is performed using a stack-driven search algorithm. The acoustic-phonetic network is searched from left to right, absorbing arcs in strict ascending time order, sorting by begin node, then end node. Search is controlled by a single pruning parameter, the beam width. Beam search uses a cutoff score calculated from the score of the best existing path, plus some delta value. The smaller the delta, the narrower the search. Beam search appears to have desirable properties for speech recognition, constraining search for high-quality input and widening it for regions of uncertainty. Due to the nature of the input (i.e., a network), separate thresholds are maintained at each node of the acoustic-phonetic network. Scores are based on likelihoods, calculated from acoustic-phonetic probabilities using the formula $-25 \cdot \log p$ (25 is a scaling factor). Likelihood values range from 0 to 255, effectively infinity (corresponding to a probability of about $4 \cdot 10^{-5}$). We have found that a normalized, time-weighted likelihood is best for controlling search, although a likelihood sum is reported as the score for a word.

Unlike the syllable-anchored matcher described previously [2], the current implementation begins a search (i.e., injects the initial node of of the lexicon graph) at every node in the input network. Although this increases the size of the search and the number of hypotheses generated (the latter by a factor of about 3), the new algorithm does not have the flaw of missing words due to syllabification failure. Nevertheless, the increase in the number of hypotheses was problematic, as it represented additional noise at the parser level. As a result, we developed a number of strategies for evaluating the goodness of a hypothesis and are currently able to discard a significant proportion of false hypotheses. The next section describes these strategies.

2.3 Hypothesis Filtering Techniques

Graph search creates problems in correctly treating local pruning decisions since a locally bad decision may prove to have been correct in a more global context defined by subsequent search. Graph search is therefore biased towards the inclusion of path segments that have no supporting evidence in the input representation. This section describes several techniques that can be used to evaluate hypotheses that contain such segments ("hallucinations") and reject those hypotheses that can be identified, on a global level, as ill-formed.

The effectiveness of a filtering technique is measured in terms of the wisp to word ratio[1]. The wisp-to-word ratio is the ratio of hypotheses per actual word in an utterance. For example, if for a 10 word utterance, the matcher generates 100 hypotheses (not all of them correct, naturally), the wisp-to-word ratio is said to be 10 (100/10). From a parsing perspective, small ratios are desirable, since they reduce the computation necessary to produce a parse and in fact allow a more thorough examination of alternatives.

The following is an example of hallucination: The lexicon graph may specify the vowel `[iy]`, but the current arc may only have a `[eh]` label. The arc may still be traversed as an `[iy]`, but the score will be 255 (the worst score, corresponding to complete lack of evidence). Hallucinations occur for various specific reasons and are in some cases beneficial: segments may be mislabelled in the input, the system should have the flexibility to continue a search in such cases. Hallucinations occur in cases where a path has accumulated a score sufficiently good to be able to absorb one or more transitions with no positive evidence. While this is useful in some cases, in other cases it will lead to excess search as well as the generation of erroneous hypotheses.

To reduce the incidence of hypotheses incorporating hallucinations, the following modifications to the search algorithm were investigated:

- Allow only a certain percentage of the segments in a word to be hallucinated ("PROP-255"). Since pruning is based on a score derived from a weighted average of component segments, a few long but high scoring segments can permit the inclusion of a significant number of hallucinated segments. The present technique compensates for this bias by calculating the proportion of such segments in an utterance. We have found that allowing only 1 out

[1] WISP: word-index + score + phones, an acronym for the information contained in a word hypothesis data structure

Authorized licensed use limited to: Carnegie Mellon Libraries. Downloaded on December 11, 2009 at 11:35 from IEEE Xplore. Restrictions apply.

of 4 segments to be hallucinated reduced the number of hypotheses produced by 30-40%, at no cost in accuracy. The formula is amended for hypotheses of fewer that 4 segments to allow one "free" hallucination for short words (i.e., those of fewer than 4 segments).

- Block self-transitions that absorb arcs of score 255 ("SELF-255"). All nodes in the lexicon graph have implicit self-transitions; the purpose of this modification is to block self-transitions that have no supporting acoustic evidence. It does not block cases where a single hallucination is produced, but it blocks its perpetuation. The advantage is seen primarily as one of reduced search.
- Block hypotheses whose final segment has a score of 255. ("FINAL-255"). Search may continue past the point where a word ends, creating false hypotheses. An example of this is words with singular and plural forms, such as *speed* and *speeds*. Every (true) match of *speed* will bring with it a companion *speeds* with a hallucinated final `[s]`. This blocking prevents such occurrences.

The three techniques described above were evaluated using three types of input data: **pt**: hand transcribed speech; **spectra**: blind labeled speech; and **ap**: automatically transcribed speech. Each type of data presents its own problems and interacts differently with various system attributes. A dataset consisting of 100 utterances from the resource management task (ten each by 10 different talkers, for a total of 876 word tokens) was used for the evaluation.

The results are presented in Tables 1, 2, and 3. Table 1 shows that there is little loss of accuracy using any of the filtering techniques, except for END-255. The loss, however, is minimal (or actually a gain the the case of spectra data) for the higher ranks. We have found that in practical terms, parser performance is governed by matcher accuracy measured in terms of words ranked third or better; correct words below that rank have minimal effect. It also appears that many correct words with low rank are in some way degenerate, and are poor candidates given the acoustic evidence present in the input.

In terms of the goals for filtering, the techniques make the following contributions: proportional filtering reduces the number of hypotheses by a factor of at least 3. Self-transition filtering, although it has minimal impact on matcher accuracy or number of hypotheses generated, reduced execution time by about 35 to 40%, presumably by eliminating dead-end paths from the search. End-filtering reduces the number of hypotheses by an average of 45% (once proportional filtering has been performed).

Table 1: Performance (percent correct) using different filters.

	standard	PROP-255	SELF-255	END-255	all
pt					
1^{st}	83	84	82	84	84
3^{rd}	**97**	**97**	**97**	**96**	**96**
10^{th}	99	99	97	97	97
all	99	99	99	97	97
spectra					
1^{st}	72	73	73	73	73
3^{rd}	**89**	**90**	**90**	**90**	**90**
10^{th}	98	98	98	98	98
all	99	99	99	99	99
ap					
1^{st}	44	44	44	45	44
3^{rd}	**67**	**65**	**65**	**64**	**64**
10^{th}	84	81	81	79	80
all	91	87	86	83	83

Note: Rows indicate the *rank* of the correct word (using score) within a window defined by the hand-labeled position of the correct word. *all* indicates presence in the lattice.

Table 3: Execution times (sec. per utterance).

	standard	PROP-255	SELF-255	END-255	all
pt	7.9	8.9	9.1	4.9	5.7
spectra	85.6	88.5	92.0	49.0	59.9
ap	90.6	93.2	94.4	66.4	56.6

Table 2: Wisp to word ratio using different filters.

	standard	PROP-255	SELF-255	END-255	all
pt	325	93	88	44	44
spectra	3156	1041	1005	637	645
ap	2878	680	394	256	248

3 Juncture Validation

While the parser has a lattice of word hypotheses available, it also needs information that specifies which words in the lattice may be joined together, and at what cost. From a scoring perspective, a string hypothesis is describable by the cost of traversing the input acoustic-phonetic network along a given path. Word hypotheses indicate the cost of the sub-paths they cover. The cost of transitions between hypotheses is determined separately, on an as-needed basis, by the *juncture verifier*. The juncture verifier determines the acoustic-phonetic cost of a transition, which is separate from cost based on grammatical constraint, obtained from the parser's knowledge base.

Five types of junctures must be considered. The first and simplest is the case where one word hypothesis abuts another. In this case, since one word hypothesis begins where the other leaves off, a legal juncture is allowed by definition. Depending on parametric biases within the parsing algorithm, about half to two-thirds of all junctures are abuts. The is no cost associated with an abut. The remaining juncture types, gaps, overlaps, and end-points, require a more complex decision process.

3.1 Gap Juncture Verification

Gaps, and end-point junctures (a special case of gap) include the greatest number of possible variants. The gap juncture algorithm uses four input structures: two word hypotheses, the acoustic phonetic network (AP), and the gap validation network. The gap validation network is an encoded form of the rules for determining valid gap junctures. The algorithm's goal is to determine the path through the AP network that will satisfy the conditions of: *Connectivity* (it connects the two word hypotheses), *Validity* (it satisfies the gap juncture rules), and *Optimality* (the best-scoring path is chosen). The score of a path is defined to be the sum of the likelihoods of the labels on the AP net arcs along the path. Although it is possible to associate penalties with given alternatives, this is not presently done.

Gap juncture rules are encoded into the *gap validation network*. This structure is a finite state machine (FSM) similar to those used in many pattern matching problems, but has been elaborated due to the requirements of the the gap juncture problem. Gap rules are specified declaratively and are compiled into executable code prior to use in the system.

Figure 2 shows a version of the gap validation network in graphic form. The network includes two special nodes (START and ACCEPT) as well as some number of non-special nodes. The start and accept nodes serve to provide singular begin and end points to the network. The non-special nodes can be divided into two sets of nodes: null and test. Null nodes simply serve as a means of providing connectivity between test nodes. They may have any in-degree and any out-degree. Test nodes specify a predicate that must be satisfied in order for traversal to occur. If no path exists through the network, then the gap fails to satisfy the juncture rules.

Authorized licensed use limited to: Carnegie Mellon Libraries. Downloaded on December 11, 2009 at 11:35 from IEEE Xplore. Restrictions apply.

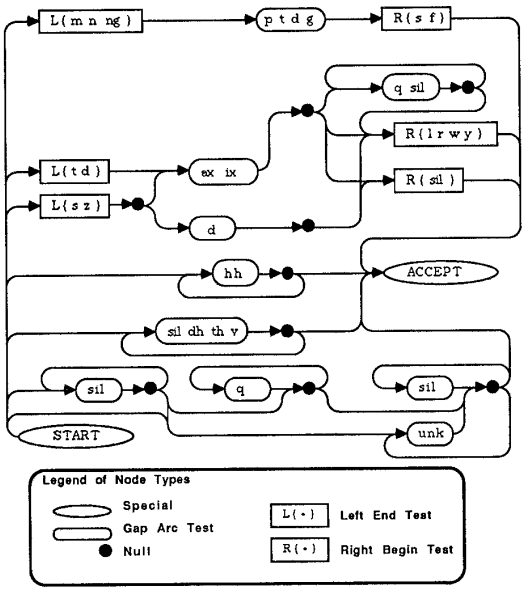

Figure 2: Graph Validation Network.

There are two types of test nodes: end/start test nodes and gap arc test nodes. A word hypothesis includes the phonetic labels hypothesized as the first and last segments of the word. An end/start test node specifies a predicate that is applied to the word's end-point phone. For example, if the left word ended in `[s]` and the predicate tested the phonetic label at the end of the left word for membership in the set `{s z}`, the predicate would be true.

Gap node arcs place conditions on the AP net arcs in a gap path. There are only two conditions that may be applied: *label identity* and *duration constraint*. The labeling condition is satisfied if an arc, when labeled with one or more of a specified set of phonetic labels, yields a score that is better than a predefined threshold value. For instance, if an arc when labeled with `[s]` or `[z]` yields scores of 23 and 255 respectively and the threshold for scores is 115, the predicate would be true. The duration constraint is satisfied when the arc has a duration within a specified range. The duration constraint may be applied in conjunction with the above labeling condition.

If it is possible to traverse the gap network from START to ACCEPT a legal gap has been found. Since a gap region may be a sub-net of the acoustic-phonetic network, the actual procedure used must be more sophisticated. A search , in some ways similar to that for word hypothesization, is performed in order to guarantee that the best-scoring gap has been identified.

3.2 Overlaps

Overlaps constitute a simpler class of junctures than gaps. The algorithm can handle one-segment overlaps as well as some two-segment overlaps. One-segment overlaps are checked for segment compatibility. If the juncturing words both hypothesize the same label for a segment, say `[n]`, the juncture is allowed. Similar segments may also be allowed, say `[s]` and `[z]`. Permissible combinations are determined by table lookup. Two-segment overlaps are allowed in the case of closure-release sequences. At this point the algorithm is unable to process junctures that overlap (or gap, for that matter) along different paths of the input network. Such junctures are rejects. Overlap scoring insures that the overlapping segments are not counted twice; a word score is adjusted appropriately in the case of permissible overlap junctures.

3.3 Performance

Juncture validation performance was evaluated for the three types of data described previously. The best instance of the correct word hypotheses from a word lattice is selected and junctures are attempted between them. The error rate for **pt** was 1%, for **spectra** 4%, and for **ap**, 12%.

4 Conclusion

Our current experience with lexical access reinforces the conclusion reached in [2]: strategies that minimize the number of chained steps produce greater robustness. Elimination of a syllable-detection stage was beneficial. More generally, strategies that defer non-reversible decisions to a level where better, global constraints can be applied produce higher quality, more accurate output. We have, in the filtering strategies, a technique that quantifies the concept of hypothesis "sensibility" on a global level and can be used to effectively guide decisions.

The division of utterance recognition into word hypothesization and parser-driven juncturing also reflects the strategy of *delayed commitment*: words are linked together only in the context of information about higher-level structures, such as are available at the parsing level of a system. Delayed commitment may be viewed as a way of achieving the effect of top-down control within a cleaner, more robust control structure.

References

[1] B. Chigier & R.A. Brennan. Broad class network generation using a combination of rules and statistics for speaker independent continuous speech recognition. *ICASSP*, 1988.

[2] A.I. Rudnicky, L.K. Baumeister, K.H. DeGraaf, & E. Lehmann. The lexical access component of the CMU continuous speech recognition system. *ICASSP*, 1987, pp. 376-380.

[3] W.H. Ward, A.G. Hauptmann, R.M. Stern, & Th. Chanak Parsing spoken phrases despite missing words. *ICASSP*, 1988.

Authorized licensed use limited to: Carnegie Mellon Libraries. Downloaded on December 11, 2009 at 11:35 from IEEE Xplore. Restrictions apply.