A Hierarchical Lexical Representation for Bi-directional Spelling-to-Pronunciation / Pronunciation-to-Spelling Generation

Author: Helen Meng,[1]
Department of Systems Engineering and Engineering Management,
The Chinese University of Hong Kong,
Shatin, N.T., HONG KONG

Abstract: We propose a hierarchical framework for integrating a variety of linguistic knowledge sources of lexical representation in English, in order to facilitate their concurrent utilization in language applications. Our unified lexical representation encompasses information including morphology, stress, syllabification, phonemics and graphemics. Each linguistic knowledge source occupies a distinct stratum in the in hierarchy. The merits of the proposed framework is demonstrated on the test bed of bi-directional spelling-to-pronunciation / pronunciation-to-spelling generation. Constraints from the multiple linguistic knowledge sources are administered in parallel during generation, by means of a probabilistic parsing paradigm. This paper extends previous work on spelling-to-pronunciation generation as reported in [28], by presenting our full results on *bi-directional* generation which includes pronunciation-to-spelling generation. We also introduce a *robust parsing* technique which is aimed for maximizing the coverage of our parser for generation. We believe that our formalism will be especially applicable for augmenting the vocabulary of existing speech recognition and synthesis systems. This work is also the precursor to the ANGIE system [20] [36], which extends our lexical representation to the phonetic level, and applies successfully in speech recognition, word spotting and durational modeling [6].

1 Introduction

This paper describes a hierarchical lexical representation, designed to incorporate multiple linguistic knowledge sources for speech applications. We attempt to demonstrate the merits of this lexical representation with the task of bi-directional spelling-to-pronunciation / pronunciation-to-spelling generation. Spelling-to-pronunciation generation is important for speech output, especially in applications such as text-to-speech systems. Pronunciation-to-spelling generation is a crucial aspect in the problem of automatic detection or incorporation of new words, which is in turn critical for the development of large vocabulary speech recognition systems. Word pronunciations, which are a major concern in recognition and synthesis, can be influenced

[1] This research was conducted in the MIT Spoken Language Systems Group, under the supervision of Dr. Stephanie Seneff and Dr. Victor Zue.

by a number of linguistic knowledge sources. These include word morphology, syntax, semantics, etc. For example, the pronunciation of "unionization" depends on whether the topic of interest concerns "ions" or "unions," which may give the respective derivations "un+ionization" or "union+ization."[2] Semantics is needed for the disambiguation of homophones such as "see," "sea" and "C". Syntax leads to the different pronunciations between the noun and verb forms of "conduct" /k aa n d ax k t/ and /k ax n d ah k t/. Coarticulatory effects in different phonetic contexts and across word boundaries are expressed as phonological rules [31]. Examples include the *flapping* of the /t/ in "water" /w ao dx ax r/ and the *palatalization* of the /d/ before the word boundary in "did you" /d ih jh y u/. Naturally, these rules are found in both synthesizers and recognizers.

It is therefore obvious that these *inter-related* knowledge sources are indispensable for modeling orthographic-phonological correspondences. We feel that instead of allowing these knowledge sources to reside individually, it will be desirable to model their inter-relationships in an integrated framework. The objective of this work is to propose a methodology for such integration. The resultant lexical framework should facilitate the concurrent utilization of the knowledge sources, and should be applicable in speech synthesis, recognition and other applications.

To successfully achieve our goal, several important issues should be addressed. First, what should be the inventory of linguistic or lexical units for describing English orthographic-phonological regularities? Second, how should these units be incorporated into the representation of English orthography and phonology? Third, what algorithms can be used to synthesize and analyze the spelling and pronunciation of an English word in terms of these lexical units? These three issues will be addressed in detail in the following when we describe our approach and report on our system's performance for both spelling-to-pronunciation and pronunciation-to-spelling generation. The novel features of our approach include: (i) the bi-directional nature of the combined parsing and generative processes, (ii) the ability to provide multiple output hypotheses, (iii) the capability of handling uncertainty in the input, as well as (iv) our treatment of nonparsable words.

2 Previous Work

2.1 Spelling-to-Pronunciation Generation

Generating pronunciation from spelling is a task that have been attempted by many. Thorough reviews can be found in [7], [11] and [16]. The *rule-based approach*, as typified in MITalk [2], the NRL system [9], and DECtalk [5], use a set of hand-engineered,

[2] This example is borrowed from [9].

ordered rules (sometimes in multiple passes) for transliteration. The approach has by far given the best generation performance. The *induction approach* attempts to infer letter-to-sound rules from a body of training data. Each rule provides a letter and its transcription under a specified spelling context. Examples can be found in [13], [18], [15], [14], [22], [30], [34] and [48]. *Hidden Markov Models (HMMs)* have also been used [23], [33], [47]. In addition, there is the *connectionist approach*, and examples include NETtalk [35], [10] as well as the Syntactic Neural Networks (SNN) [21]. *Psychological approaches* attempt to model the way humans pronounce by analogy [8], [45]. Finally, *case-based or memory-based reasoning* are approaches which try to generate a pronunciation based on similar examples in the training corpus [4], [11], [41], [42].

Direct comparisons of performance of the different systems is difficult due to the lack of standardized phone sets, data sets, or scoring algorithms. However, the rule-based systems generally achieve the highest performance, with word accuracies up to the high 90% range. The automatic systems have reported phone accuracies hovering in the low 90% range.

2.2 Pronunciation-to-Spelling Generation

In comparison to the spelling-to-pronunciation generation, the reverse task of pronunciation-to-spelling is less commonly undertaken. We know of three approaches that have previously been adopted. The first is a *combined rule-based and induction approach* [51]. Inferred rules are backed up by a set of manually generated rules, since the phoneme-to-letter mapping has much less consistency than letter-to-phoneme mapping. *HMMs* [3] have also been used to model the acoustics of training sentences directly from their orthographic transcriptions. *SNNs* [21] as mentioned above is the only bi-directional system that we know of, using two neural networks to perform statistical string translation. General word accuracies for this task vary between 20% to 30%, corresponding to letter accuracies in the 70% range.

3 The Hierarchical Lexical Representation

As mentioned previously, a variety of linguistic knowledge sources play an important role in determining English spelling/pronunciation correspondences. For example, part-of-speech causes the noun and verb forms of "record" to be pronounced differently. A morphological boundary causes the letter sequence "sch" in "discharge" to be realized differently from that in "school" or "scheme". Stress can change the identity of vowels in a word, e.g. "define" vs. "definition". Also, syllabic constraints are expressible in terms of the sequential ordering of distinctive features – sonority sequencing in manner features, and phonotactic constraints in place and voicing features. Furthermore, there are graphemic constraints for transition from one letter to the next. A novel feature of our system is that multiple layers of representation are combined to capture long and short distance constraints. These include *word class, morphs, syllables, manner classes, phonemes and graphemes.*

We created a framework which describes the spellings and pronunciations of English words using only a small inventory of labels associated with morphological and phonological units. These units are organized as a hierarchical tree structure, where the various levels of linguistic knowledge are collectively used to describe orthographic-phonological regularities. Figure 1 illustrates the description of the word "define."

The higher levels in the hierarchy encode longer distance constraints across the word, while the lower levels carry more local constraints. The layer of terminal nodes, which currently represent letters in a word spelling, can conceivably represent phones in the word pronunciation *as well.* As such the task of spelling-to-pronunciation generation should involve deriving the higher levels in the hierarchy based on an input spelling, and subsequently generating a phone sequence (pronunciation) while making use of all available hierarchical constraints. This process can also be used for pronunciation-to-spelling generation, by means of swapping the input/output specifications. Our current system does not involve phones, because our experimental corpus provides phonemic pronunciations only. However, if the terminal nodes were actually dual in nature, the hierarchical representation should be able to capture phonological rules between layers 6 (phonemes) and 7 (phones).

We have made an attempt to distinguish the context which calls for a long vowel from that which calls for a short vowel. A long vowel tends to be succeeded by a grapheme containing the letter "e", such as in the /ay/ in "define" being followed by #[ne] (Figure 1). An effort is made to preserve the context for "defining" (Figure 2), by using the terminal #[n_e]. These *underbar* terminals are restricted to the CODA positions of stressed syllables, and are predicted during the parsing process. Therefore, long vowels tend to transition from left-to-right to the CODA of a grapheme terminal ending with the letter "e" (or an "underbar" terminal), because the bigram constraint disfavors the advancement to a stressed M-ONSET[3] such as in "definition" (Figure 3).

Our representation promotes extensive structural sharing among words. For example, "define" and "defining" share all the structures corresponding to the prefix and root, and "definition" shares all structures corresponding to the prefix. A compact lexical representation with sharing capabilities is potentially applicable and desirable

[3]The special "moved-onset" M-ONSET category is used due to the Maximal Onset Principle and the Stress Resyllabification Principle.

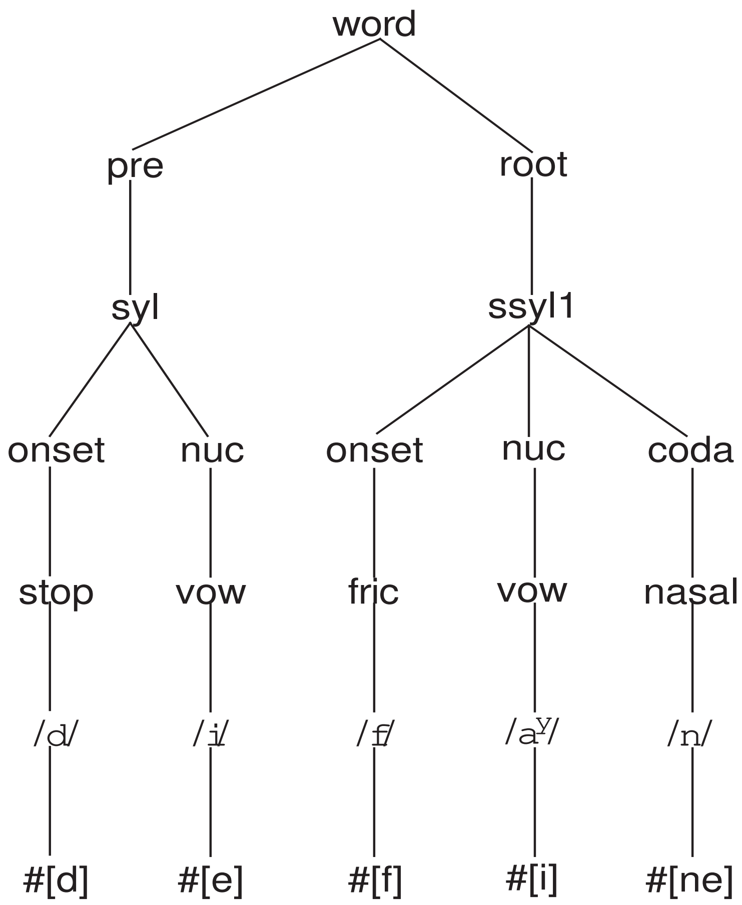

Figure 1: Lexical representation for the word "define."

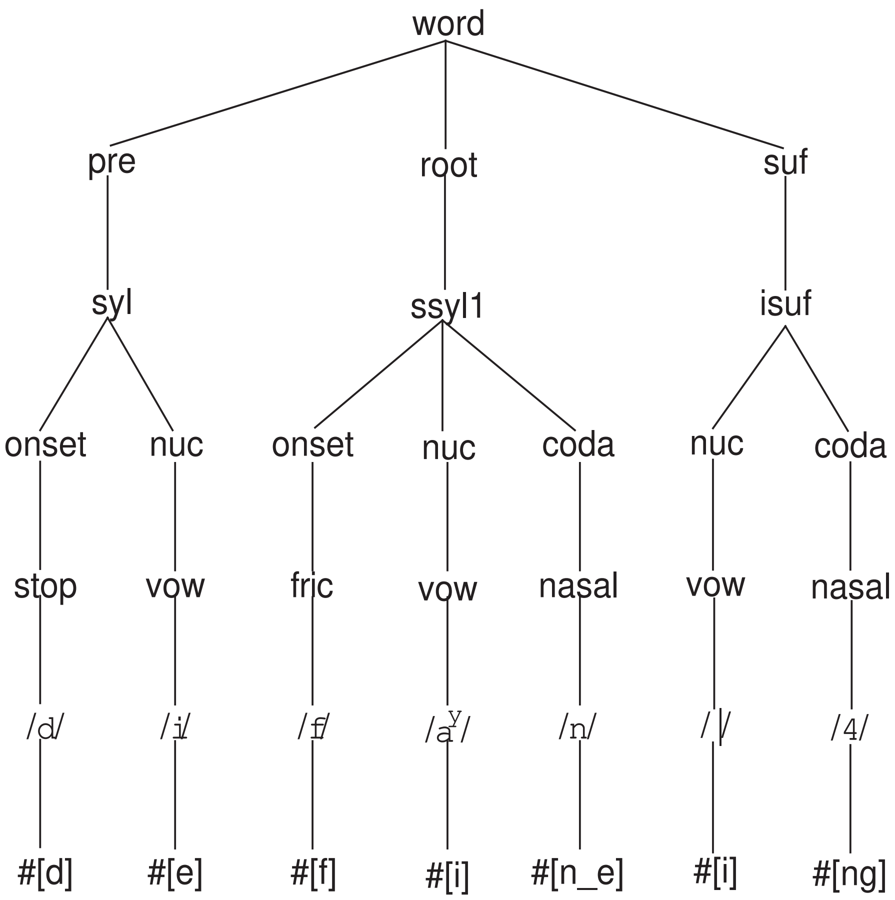

Figure 2: Lexical representation for the word "defining."

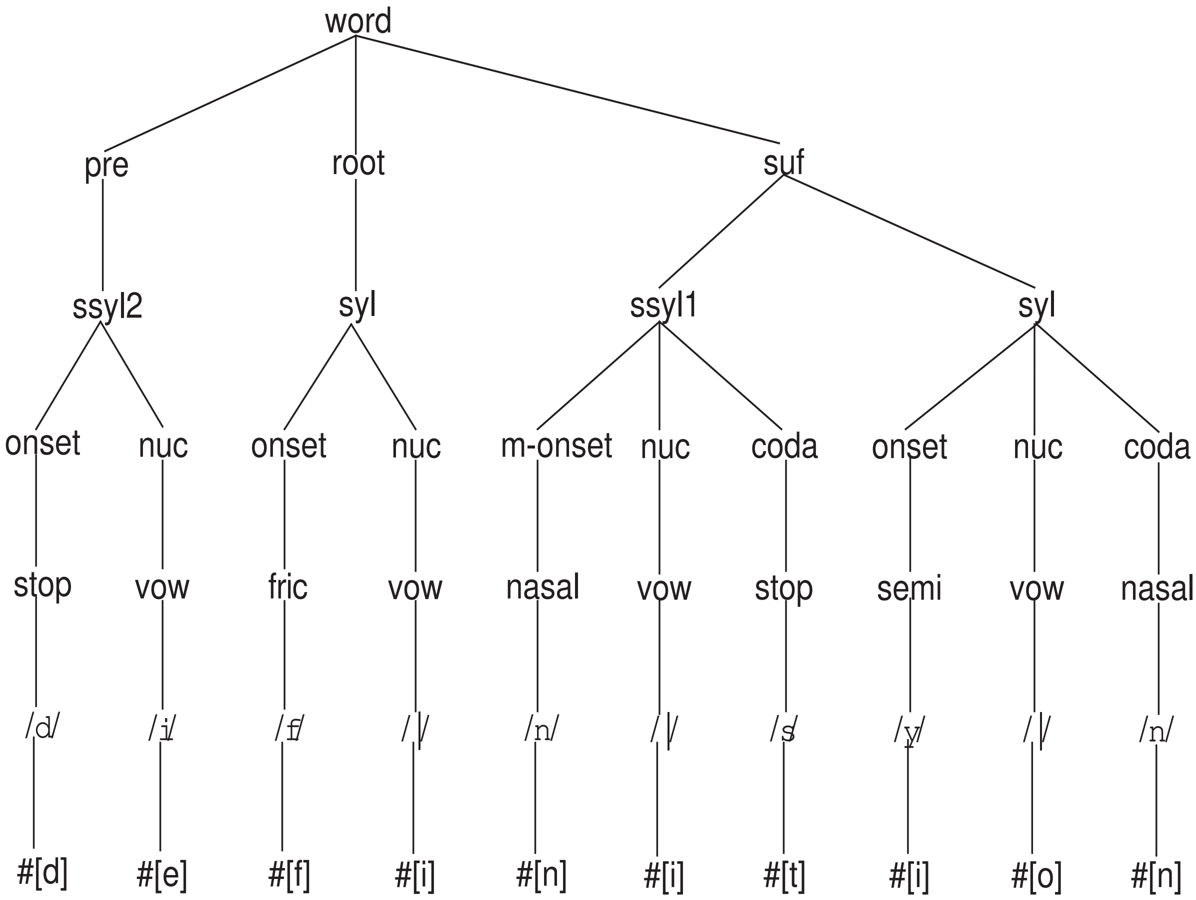

Figure 3: Lexical representation for the word "definition."

for large-vocabulary speech recognition tasks. It allows sharing of probabilities which ameliorate the sparse data problem, under the assumptions that sharing takes place among similar distributions only. In this work, the lexical representation is combined with a parsing framework in order to cast the spelling-to-pronunciation generation problem as directly symmetric to the pronunciation-to-spelling generation problem.

4 The Parsing Algorithm

The hierarchical lexical representation forms the infrastructure upon which generation is carried out. The approach for generation is one of synthesis-by-analysis in a parsing framework. Our technique is a hybrid of rule-based and data-driven strategies. The advantage of the hybrid approach is to tradeoff the expensive efforts in providing an elaborate set of letter-to-phoneme mappings from linguistic expertise, with a small set of simple rules augmented by constraints automatically discovered from a body of training data. Probabilities prioritize some constraints over others, and thus elegantly bypass problems with rule interactions and conflicts. In the following, a detailed account of data preparation, and the training and testing procedures is provided.

4.1 Data Preparation

The training corpus consists of about 8,000 words derived from the Brown Corpus [17]. Table 1 shows several examples of the lexical entries in the training corpus. The spellings are marked with symbols for specifying syllable stress and morphological decomposition.[4] The symbols include markers for prefix (=), ROOT2[5] (==), suffix (+), inflectional suffix (++), compound word ($), moved onset (@), primary stress (!) and secondary stress (?).[6]

A small set of context-free rules are written for the TINA [37] parser.[7] These rules serve to incorporate linguistic knowledge in the training parse trees, constituting the rule-based portion of our approach. About 100 context-free rules, such as those illustrated in Table 2 are written by hand, and training words are parsed using TINA [37] according to their marked linguistic specifications. The parse trees, with format as shown in Figure 1, are then used to train the probabilities in a set of "layered bigrams" [38].

4.2 Training Procedure

The training parse trees are used to train layered bigram probabilities, which constitutes the data-driven aspect of our hybrid approach. These probabilities become the sole parameters of our parser. Probabilities are attached to the sibling-sibling transitions in the context-free rules, which serve to incorporate linguistic knowledge into the parser. The training procedure is geared towards the implementation of a

[4] The morphological decomposition of the training words were provided by Dr. Sheri Hunnicutt.

[5] The second unstressed syllable of the root.

[6] Unstressed syllables are not marked.

[7] The TINA parser was previously used with a context-free grammar to achieve spoken language understanding [37]. In this work we have adapted the parser to handle sublexical parses. Parsing sublexical structures is simpler than parsing sentences, since the former does not have traces or gaps.

D!E=DI+@C?ATE++*D	/d eh d ih k ey t ix d/
H!ER==O	/h ih r ow/
ST!AND$P?OINT	/s t ae n d p oy n t/

Table 1: Examples of lexical entries in the training corpus.

word	→ [prefix] root root2 [suffix]
root	→ stressed-syllable [unstressed-syllable]
root2	→ (stressed-syllable secondary-stressed-syllable unstressed-syllable)
stressed-syllable	→ [onset] nucleus [coda]
onset	→ [aspirant] [nasal] [fricative] [stop] semivowel
fricative	→ (/f/ /v/ /th/ /dh/ /s/ /z/ /sh/ /zh/)
/f/	→ (“ph” “f” “gh” “fe” “ff”)

Table 2: A few examples of generalized context-free rules for different levels in the hierarchy.

bottom-up, left-to-right parser. In the analysis of a training parse tree, a basic 4-tuple is used. It consists of the elements:

1. Right-Sibling (RS): This can be any category in the parse tree.
2. Right-Parent (RP): The category above RS in the parse tree.
3. Left-Sibling (LS): The category to the left of RS in the parse tree.
4. Left-History (LH): The entire column history (consisting of seven categories, one from each layer) to the left of RS. This may assume the special case of START at the beginning of the parse.

We estimate the *bottom-up prediction probabilities* for each node in the parse tree (i.e. $P(RP|RS, LH)$) , and *column advancement probabilities* for extending a column to the next terminal (i.e. $P(RS|LH)$). This is done by tallying counts from all the training parse trees. As an example, refer to the center of the tree in Figure 3. A possible 4-tuple may be:

1. RS is M-ONSET
2. RP is SSYL1
3. LS is NUC
4. LH is [WORD ROOT SYL NUC]

4.3 Testing Procedure

During spelling-to-pronunciation generation, the parse tree is constructed from bottom-up and left-to-right. Bottom-up prediction accumulates *bottom-up probabilities*, and terminates when the root note [WORD] is reached, or when the RP merges with the LP, signifying the derivation from a single context-free rule. At this point, the parser will proceed to advance to the next terminal, by accumulating the *column-advancement probability*. A complete parse theory is obtained when all the input terminals are exhausted, and when the final column advances to an END node. Details were described in [28].

Our parsing algorithm in [28] needs to be modified and extended in order to accomplish pronunciation-to-spelling generation. Here the parser accepts a string of phonemes as input, and from it creates an entire parse tree with grapheme terminals, which provide the output spelling of our generation process. The first phonemic preterminal is used for bottom-up prediction of the first parse column, until we reach the root node of the parse tree for [WORD]. As such we have created the first six layers of the first column. This entire partial column provides context for our parsing algorithm to identify possible grapheme terminals in order to complete the column. Identification is limited to contexts previously found in the training data. The completed column is then advanced to the next possible grapheme terminal(s), similar to the spelling-to-pronunciation generation procedure. However the second phonemic preterminal is then used as a constraint to filter for plausible parse theories as we construct the second parse column bottom-up. The completed second column is then advanced and parsing proceeds iteratively. The output spelling is eventually generated from the terminal layer of the complete parse.

Our parser employs an inadmissible stack decoding search algorithm. A theory is pushed onto the stack after an entire column is built bottom-up. The stack is then sorted, and the highest scoring theory is popped and advanced from left-to-right. Our scoring scheme is as follows:

$$\hat{f}(c) = \alpha \hat{f}(c') + (1-\alpha) p(c', c) \tag{1}$$

where $\hat{f}(c)$ is the stack score from the START column to the current column c,
c' is the column preceding c in the parse tree,
$p(c', c)$ is the log-likelihood associated with extending the parse tree from c' to c, and
α is some fading factor $(0 < \alpha < 1)$

This search algorithm encodes some degree of short-term memory in the stack score, where the new column always contributes towards a certain portion of the stack score (according to the pre-set weight of $\alpha = 0.95$), while the remaining portion associated with the past gradually fades away, so that the distant past contributes less to the stack score than the recent history, and the score tends to remain quite stable over time. The outcome of this search is that the ordering tends to place together parse theories with similar recent columns and different distant columns. Furthermore, while this stack decoding search algorithm is in action, our stack contains multiple parse theories of different lengths, which are ranked according to the stack score. The highest scoring theory is popped off the stack and advanced first. As parsing proceeds, it may happen that the newly advanced theory produces a complete parse tree (or hypothesis) for the input word – in other words, the complete parse tree provides a full pronunciation (or spelling) hypothesis for the input spelling (or pronunciation).

We continue to pop theories of the stack and further advance each parse, until all the theories on the stack are completed. As such our system has the advantage of producing multiple hypotheses, and the parsing process can be designed such that it terminates after a desired number of complete hypotheses have been popped off the stack. Alternatively, a limit can be set on the maximum number of theories (partial and complete) to be popped off the stack. The complete theories thus produced are subsequently re-ranked according to their actual parse score. The idea is to first use the "fading" stack criterion as an inexpensive means to obtain a handful of plausible complete hypotheses, and follow up with a more careful means of ranking (with no fading) in order to find the "best" theory. Though our search is inadmissible, we are able to obtain *multiple* hypotheses inexpensively.

5 Experimental Results

This section reports on the performance of bi-directional generation based on the same training and test sets. Hence in addition to pronunciation-to-spelling generation performance, we have included performance on spelling-to-pronunication (as reported in [28]), followed by the combined analysis of both generation tasks for completeness.

Our experimental corpus consists of about 10,000 most frequent words appearing in the Brown Corpus [17], and each lexical entry contains a spelling and a single phoneme string as its pronunciation. The words are ordered alphabetically, and entries which are marked as either function words (e.g. "a," "am," "by," and "have") or abbreviations (e.g. "AUG," "CORP," "ETC.") are discarded. We set aside one tenth of the words as a future test set, and also one tenth of the remaining words as the development test set. The rest of the words (about 8,000 in total) are used for training. The results given in this section are based on the development test set only, as slight improvements and a robust parsing mechanism will ensue, after which we will use our combined techniques to report on our performance for the test set.

The parser was set to terminate after obtaining up to a maximum of 30 complete parse theories, or after the maximum number of theories (partial and complete) popped off the stack reaches 330, whichever happens first. These numbers are empirically chosen as a limit on the depth of the search. The thirty hypotheses are then re-ranked according to their actual parse score, and the performance accuracies reported below are based on the new set of rankings.

Experiments on both spelling-to-pronunciation and pronunciation-to-spelling generation were conducted using 26 letters, one graphemic place-holder and 52 phonemes (including stressed and unstressed versions of several vowels and pseudo diphthongs such as /or/).

5.1 Evaluation Criteria

The criteria which we use are similar to those used in other systems reported previously.

1. Word accuracy — This requires a strict string match in terms of spelling or

the pronunciation phoneme string. Our corpus only provides a single reference pronunciation per word. Any deviation from the reference string is regarded as an error.

2. Letter/Phoneme accuracy — In order to indicate the extent to which a generated spelling or pronunciation is correct, phoneme/letter accuracy should be a good evaluation criterion to use for spelling-to-pronunciation / pronunciation-to-spelling generation. The generated string is aligned with the "correct" string using a dynamic programming algorithm, which selects the alignment with the minimum number of insertion, deletion and substitution operations necessary to map the generated string to the reference string. The accuracy is computed by subtracting the sum of the insertion (I), deletion (D) and substitution (S) error rates from 100%, i.e.

$$accuracy = 100\% - (I + D + S)\% \tag{2}$$

This evaluation criterion is the one adopted by NIST for measuring the performance of speech recognition systems.

5.2 Results on Bi-directional Generation

In spelling-to-pronunciation generation, about 6% of the development test set was nonparsable. This set consists of compound words, proper names, and other words that failed due to sparse data problems. Results for the parsable portion of the test set are shown in Table 3. The 69.3% word accuracy corresponds to a phoneme accuracy of 91.7%, where an insertion rate of 1.2% has been taken into account, in addition to the substitution and deletion errors. Our phoneme accuracy lies within the low 90's percentage range of the automatic spelling-to-pronunciation generation systems described earlier in this paper. Our N-best word accuracy with $N = 5$ is in the mid 80 percent range, and is more comparable with the performance of rule-based approaches. This may suggest that we can seek performance improvement by means of better search procedures. Alternatively, we can try to improve performance by using more contextual information during parsing, or devise post-processes to select among the top few generated outputs.

Figure 4 is a plot of cumulative percent correct of whole word theories as a function of the N-best depth for the development test set. Although 30 complete theories were generated for each word, no correct theories occur beyond N =18 after re-sorting. Performance reaches an asymptotic value just beyond 89%.

In pronunciation-to-spelling generation, about 4% of the test set was nonparsable. Results for the parsable words are shown in Table 3; top-choice word accuracy for pronunciation-to-spelling is about 52%. This corresponds to a letter accuracy of

Spelling-to-Pronunciation Generation				
Accuracy		top choice correct	top 5 correct	top 10 correct
train	word	77.3%	93.7%	95.7%
	phoneme	94.2%	–	–
test	word	69.3%	86.2%	87.9%
	phoneme	91.7%	–	–

Pronunciation-to-Spelling Generation				
Accuracy		top choice correct	top 5 correct	top 10 correct
train	word	58.8%	85.0%	89.3%
	letter	90.6%	–	–
test	word	51.9%	77.0%	81.1%
	letter	88.6%	–	–

Table 3: Bi-directional generation experiments: Word, phoneme and letter accuracies for training and testing data. Nonparsable words are excluded.

88.6%, with an insertion error rate of 2.5% taken into account in addition to substitution and deletion errors. Our system seems to compare favorably with the other systems.

Figure 5 is a plot of the cumulative percent correct (in pronunciation-to-spelling generation) of whole word theories as a function of N-best depth of the test set. The asymptote of the graph shows that the first 30 complete theories generated by the parser contain a correct theory for about 83% of the test words. Within this pool, re-sorting using the actual parse score has placed the correct theory within the top 10 choices for about 81% of the cases, while the remaining 2% have their correct theories ranked between $N = 10$ and $N = 30$. Re-sorting seems to be less effective in the pronunciation-to-spelling case, presumably because many more "promising" theories can be generated than for spelling-to-pronunciation. For example, the generated spellings from the pronunciation of "connector" i.e., the phoneme string /k ix n eh k t er/ include: "conecter," "conector," "connecter," "connector," "conectar," "conectyr," "conectur," "connectyr," "connectur," "conectter," "connectter" and "cannecter." A possible reason for this is the ambiguity in phoneme-to-letter mapping, e.g., the phoneme /er/ above is mapped to "er," "or," "ar," "yr" and "ur". Another reason is that geminate letters are often mapped to the same (consonantal) phoneme, e.g., the phoneme /n/ above can be mapped to "n" or "nn." Many of these hypotheses can be rejected with the availability of a large lexicon of legitimate English spellings.

5.3 Error Analyses

Both of the cumulative plots shown above reach an asymptotic value well below 100%. The words that belong to the portion of the test set lying above the asymptote appear intractable – a correct pronunciation / spelling did not emerge as one of the 30 complete theories. We have grossly classified these errors into four categories, and examples are shown in Table 4: (1) Generated pronunciations that have subtle deviations from the reference strings. (2) Unusual pronunciations due to influences from foreign languages. (3) Generated pronunciations which agree with the regularity of English letter-phoneme mappings, but were nevertheless incorrect. (4) Errors attributable to sparse data problems. For the example "braque" in Table 4, one cause of error is the foreign origin of the word. However, if the training data included the words such as "plaque" or "critique" to provide for the statistics whereby the letter sequence "que" can be pronounced as /k/ in a coda context, it is conceivable that the system can produce the correct pronuncation /b r ae k/ for "braque". It is also interesting to note that much overlap exists between the set of problematic words in spelling-to-pronunciation and pronunciation-to-spelling generation. This suggests that improvements made in one generative direction should carry over to the opposite

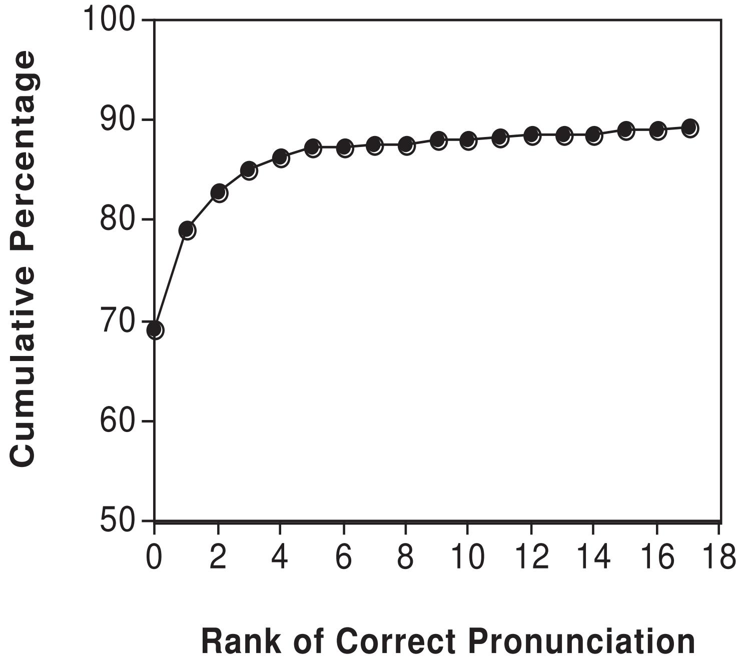

Figure 4: Spelling-to-pronunciation generation experiments: Percent correct whole-word theories as a function of N-best depth for the test set.

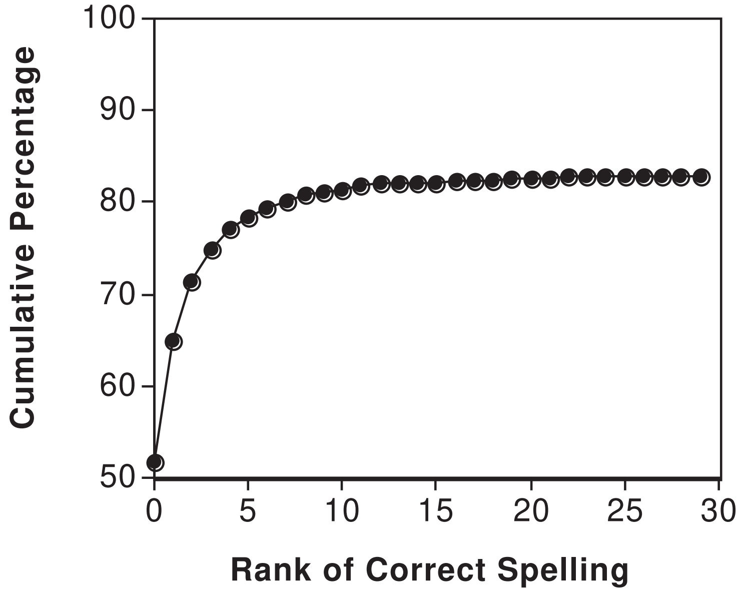

Figure 5: Pronunciation-to-spelling generation experiments: Percent correct whole-word theories as a function of N-best depth for the test set

direction as well.

Certain pronunciation errors, such as the generated pronunciation for "acquiring," /ix k w ay r ix ng/, may be considered by some as correct. Likewise are other examples such as /p ae s y ix n ey t/ generated from "passionate" instead of /p ae s y ix n ix t/,[8] /k ix r t u n/ for "cartoon" instead of /k aa r t u n/, and /p iy p el/ for "people" instead of /p iy p ix l/. These cases can perhaps be rectified if alternate "correct" pronunciations were available. Spelling errors that are near misses — words like "viscossity" (instead of the correct spelling "viscosity"), "abundent" (instead of "abundant"), "idiological" (instead of "ideological"), or names like "ilynoy" (instead of "illinois") or "claten" (instead of "clayton"), can perhaps be salvaged by spell-checking or verification.

[8]Recall that we are using the "underlying phonemic" form, and as a result the /sh/ phoneme is transcribed as /s y/ in "passionate".

Category	correct spelling	generated spelling	generated pronunciation	correct pronunciation
(1) Subtle	acquiring	equiring	/ix k w ay r ix ng/	/ix k w ay er ix ng/
	balance	balence	*correct*	/b ae l ix n s/
	launch	lawnch	*correct*	/l ao n ch/
	pronounced	pronounst	/p r ix n aw n s t/	/p r ow n aw n s t/
(2) Unusual	champagne	shampain	/ch ae m p ix g n iy/	/sh ae m p ey n/
	debris	dibree	/d iy b r ix s/	/d ix b r iy/
(3) Regular	basis	*correct*	/b ae s ix s/	/b ey s ix s/
	elite	aleat	/ix l ay t/	/ix l iy t/
	violence	viallence	*correct*	/v ay ix l ix n s/
	viscosity	viscossity	/v ih s k ow s ix t iy/	/v ih s k aa s ix t iy/
(4) Sparse	braque	brack	/b r ae k w iy/	/b r ae k/

Table 4: Some examples of generation errors.

6 Evaluating the Hierarchy

In order to explore the relative contribution of each linguistic level in the generation task, we have conducted a series of experiments whereby an increasing amount of linguistic knowledge (quantified in terms of the number of layers in the hierarchy) is omitted from the training parse trees. We focus on the letter-to-sound generation performance for this investigation, as we expect that the reverse generation process should display similar trends. The system is re-trained on the training set and re-tested on the development test set for each reduced configuration. Four measurements are recorded for each experiment:

1. Top-choice word accuracy on the development test set, where a word is considered correct when there is an exact match between the generated phoneme string and the *single* pronunciation provided by the lexical entry.

2. Perplexity, i.e., the average number of distinct possibilities for the next grapheme, as predicted from the current grapheme, which may be interpreted as the geometric mean of the possible choices for the next grapheme. It is computed as:

$$PP = exp\{\frac{1}{n} * ln(P_T)\} \tag{3}$$

where	PP	is the perplexity,
	T	is the parse theory,
	P_T	is the probability of the parse theory T, which is the product of all the bottom-up prediction and left-to-right advancement probabilities involved, and
	n	is the number of columns in the parse theory, including the END column.

3. Coverage of the test set, obtained by subtracting the percentage of nonparsable words from 100%. Nonparsable words are those for which no complete parse is generated.

4. The number of system parameters.

The top-choice word accuracy and perplexity reflect the amount of constraint provided by the hierarchical representation, while coverage exhibits the extent to which the parser can share training data across different layers in the hierarchy, so that it can generalize and process previously unseen structures. The number of system parameters is a measurement from which one can observe the parsimony of the hierarchical framework in capturing and describing English orthographic-phonological regularities. It also provides some indication of the computational load required by the configuration of the system.

Omitted Layer	Word Accuracy	Perplexity	Coverage	No. of Params.
NONE	65.4%	8.3	94.4%	32,700
2 (morphs)	60.4%	9.3	95.7%	24,700
3 (stress)	57.4%	8.5	95.0%	24,000
5 (broad cl.)	67.5%	8.0	93.9%	32,000
2 and 5	62.8%	9.0	95.4%	24,600
3 and 5	59.9%	8.1	94.5%	23,800
2, 3 and 5	56.4%	9.1	96.1%	17,300
2, 3, 4 and 5	51.1%	10.1	97.1%	14,800

Table 5: Experimental results showing the relative contribution of the different layers in the hierarchical representation

With the omission of each linguistic level, we expect to see two antagonistic effects on generation accuracy — the diminishing use of linguistic knowledge decreases the amount of constraint provided for generation, which should cause degradation in performance. On the other hand, relaxing constraints brings about more sharing of training data across levels. This should help alleviate the sparse data problem and enhance wider coverage, which may potentially contribute to performance improvement.

6.1 Results and Analysis

The experimental results on investigating the hierarchy are tabulated in Table 5. As before, system uses 26 letters, 1 graphemic place-holder and 52 phonemes (including several unstressed vowels and pseudo diphthongs such as /ao r/). Word accuracy refers to the percentage of the test set for which a correct pronunciation is generated from the word spelling. Nonparsable words are counted as errors. The resultant percentage value is thus lower than the word accuracy reported earlier in the previous section, which is computed based on the parsable fraction of the test set. The number of system parameters in each case is rounded to the nearest hundred.

The advantages of using higher level linguistic knowledge for letter-to-sound generation can be gleaned from Table 5. Each layer in the hierarchical representation embodies one type of linguistic knowledge, and for every layer omitted from the representation, linguistic constraints are usually lost, manifested as a lower generation accuracy, higher perplexity and greater coverage. Fewer layers also require fewer training parameters.

Such phenomena are generally true except for the case of omitting the layer of broad classes (layer 5), which seems to introduce *additional* constraints, thus giving a higher generation accuracy, lower perplexity and lower coverage. This can be understood by realizing that broad classes can be predicted from phonemes with certainty,[9] and therefore the broad class layer provides no additional linguistic constraint. The inclusion of the broad class layer probably led to too much smoothing across the individual phonemes within each broad class, resulting in inferior performances.

Since we have discovered that omitting the layer of broad classes leads to slight performance improvement, this reduced configuration is maintained in all of our subsequent experiments.

6.2 A Non-Linguistic Approach

We also compared our current hierarchical framework with an alternative approach which excludes higher level linguistic knowledge. This non-linguistic approach performs transliteration using local letter context, and therefore bears resemblances with the case-base approaches and psychological approaches presented in Section 1. The approach is also designed to mirror the layered bigrams with fragment bigram constraints, as will be explained in this section.

The word is represented mainly by a spelling and an aligned phonemic transcription, using the /null/ phoneme for silent letters. Consonant phonemes for geminate letters, however, are duplicated. In general, alignment complies with the training parse trees from the hierarchical approach. For instance, "bright" is transcribed as /b r ay null null t/, and "spaghetti" as /s p ix g null eh t t iy/. The word is then fragmented exhaustively to obtain letter sequences (word fragments) shorter than a set maximum length. During training, bigram probabilities and phonemic transcription probabilities are then computed for each letter sequence. Therefore this approach captures some graphemic constraints within each word fragment, but higher level linguistic knowledge is not explicitly incorporated. Letter-to-sound generation is accomplished by finding the "best" concatenation of letter sequences which constitutes the spelling of the test word. Mathematically, let l denote the spelling of the test word, an s_i denote a letter sequence (or word fragment) with t_i being its most probable phonemic transcription. Furthermore, let S be a possible concatenation which constitutes l, i.e. $l = S = s_1 s_2 ... s_n$ which corresponds to the phonemic transcription $T = t_1 t_2 ... t_n$. The spelling-to-pronunciation generation process can then be represented as:

[9] These unity probabilities are also counted as system parameters. Broad classes may still serve a role as a "fast match" layer in recognition experiments, where their predictions could no longer be certain, due to recognition errors.

Word	Segmentation	Generated Pronunciation
acquire	#ACQUIR+E#	/ix k k w ay er/ + /null/
bothered	#BOTH+ERED#	/b ow th null/ + /er null d null/
bulletin	#BULLE+TIN+#	/b uh l l ix/ + /t ix ng null/
enjoyment	#ENJOY+MENT#	/ix n jh oy null/ + /m ix n t/

Table 6: Examples of generated outputs using the non-linguistic approach

$$\begin{aligned} T &= \max_S P(T \mid S,\ l)\ P(S \mid l) \\ &= \max_S \prod_{i=1}^{n} P(t_i \mid s_i,\ l)\ P(s_i \mid s_{i-1},\ l)^{\alpha} \end{aligned} \tag{4}$$

In the above equation, α is a weighting factor for the language score and its value was optimized using the training data. A bigram language model is used for the letter sequences. In going from the first step to the second in Equation 4, we have assumed that the prediction of the next letter sequence is dependent on the current letter sequence only. This is purposely designed to conform with the layered bigrams, where the *current column* is the only context applied in predicting the *next column*. Another assumption is that the phonemic transcription of a letter sequence is independent of the context outside the letter sequence itself, so that each letter sequence is directly mapped to its *single* most probable phonemic transcription. The testing procedure uses a Viterbi search to find the most probable segmentation for the spelling of a word, drawing lexical analogies while capturing context through the use of longer letter sequences, and eventually derives the generated phonemic transcription from the top-scoring segmentation. Some examples of generated outputs from the non-linguistic approach are tabulated in Table 6.

To ensure a fair comparison with the hierarchical approach, we use the same training set and development test set to run spelling-to-pronunciation generation experiments with the non-linguistic approach. Duplicated phonemes and the /null/ phoneme are removed from the generated pronunciations before matching against reference pronunciations. Several different value settings were used for the maximum word fragment length. Generation accuracy is expected to improve as the maximum word fragment length increases, because longer letter sequences can capture more context. This should also be accompanied by an increase in the number of system parameters due to the combinatorics of the letter sequences.

Max. Word Fragment Length	Word Accuracy	Perplexity	No. of Params.
4	60.5%	14.8	303,300
5	67.1%	13.9	508,000
6	69.1%	13.2	693,300

Table 7: Experimental results for spelling-to-pronunciation generation using the non-linguistic approach

6.3 Results with the Non-Lingustic Approach

The results for letter-to-sound generation using the non-linguistic approach are shown in Table 7. The system includes 26 letters and 58 phonemes. The 6 extra phonemes are: the /null/ phoneme, the pseudo phoneme /ixk/ which can map to the second letter in "mcclellan", as well as 4 pseudo affricates, /ksh/, /ngz/, /tz/ and /ks/, mapping respectively to the third letter in "luxury", "anxiety", "nazi" and "taxi". Thus, letter-to-phoneme alignment is one-to-one, and no grapheme place-holder is needed. There are no cases of nonparsability in testing because the non-linguistic approach can "back off" to mapping individual letters to their most probable phonemes.

Experiments were conducted with maximum word fragment lengths of 4, 5 and 6 respectively.[10] As a comparison, the two longest graphemes in the hierarchical approach are 4 letters long – "ough" and "eigh", which are pronounced respectively as /ow/ and /ey/ in "dough" and "eight".

We also conducted a baseline experiment for the non-linguistic approach by setting maximum fragment length to 1 (i.e., the pronunciation of a word is obtained by mapping each letter in the spelling to its most probable phoneme). This gave a word accuracy of 0.8%, which indirectly shows that the non-linguistic approach relies heavily on the use of long word fragments to capture contextual information for letter-to-phoneme mappings.[11]

Table 8 shows some erroneous outputs of the single-layered approach. These errors seem to be mostly a result of: (i) forcing a one-to-one letter-to-phoneme mapping, and (ii) the exhaustive fragmentation of the word spelling regardless of the higher level linguistic organization of the word, e.g., its syllabification. For instance, the generated pronunciation for "bubble" lacks a syllable, and ends with the phoneme sequence /b l/ which is not licit in English. The letter sequence "th" at the beginning

[10]We did not investigate cases where maximum word fragment lengths are set beyond 6, due to computational limitations, and the vast number of training parameters required.

[11]The mean fragment length used for the test set (with maximum fragment length set at 6) was 3.7, while the mean grapheme length in the hierarchical approach was 1.2.

Word	Segmentation	Generated Pronunciation
bubble	#B+UBB+L+E#	/b/ + /ah b b/ + /l/+/null/
suds	#S+UDS#	/s/ + /null d z/
thigh	#T+HIGH+#	/t/ + /h ay null null/

Table 8: Error examples made by the non-linguistic approach

of a word like "thigh" should be a syllable onset, which is often pronounced as /dh/ or /th/, but never the phoneme pair /t h/. Another example is the word "suds", in which the letter sequence "uds" is transcribed as /d z/. Here, an analogy is drawn from "clouds" /k l aw null d z/ but the letters "ou" which should together map to the syllable nucleus have been split. These kinds of errors are not found for the hierarchical approach, because they are precluded by the higher level linguistic constraint in the hierarchical framework.

The highest word accuracy obtained from the hierarchical approach (67.5%)[12] outperforms the single-layer approach with maximum word fragment length set at 4 and 5, but lies below that with maximum length set at 6. However, the hierarchical approach is capable of reversible generation using about 32,000 parameters, while the single-layer approach requires 693,300 parameters for uni-directional spelling-to-pronunciation generation. In order to achieve reversibility, the number of parameters needs to be doubled.

Given an input spelling, the hierarchical approach may generate *multiple* parses with the correct pronunciation. Summing the probabilities of all the relevant parse theories is computationally prohibitive, hence the perplexity values shown in Table 5 are computed from the top-scoring parse theory *only*, and form upper bounds of the *true* perplexity values. The hierarchical approach obtains its lowest perplexity value of 8.0 while omitting the broad class layer, and the single-layer approach obtains 13.2 with the maximum fragment length set at 6. Though the hierarchical approach seems to provide more constraints, direct comparison of these perplexity values may not be entirely fair because the two approaches divide their probability spaces in very different ways. Perhaps a better comparison is to assume up front that partial information (e.g. segmentation) about the top-scoring parse is provided, and focus on the corresponding probability subspace. In this case, the perplexity values computed for the hierarchical approach and the single-layer approach are 5.3 and 7.7 respectively. The hierarchical approach is still substantially more constraining.

[12] 6% of the errors in the hierarchical approach are due to parse failure. The next section presents a number of robust parsing strategies to overcome this problem.

7 Robust Parsing

In a parsing paradigm there is a constant tradeoff between providing parsing constraints and obtaining sufficient coverage. Constraints are important to prevent overgeneration, but in order to account for previously unseen structures, it is necessary to relax certain constraints and generalize. Both of our generation experiments yielded a set of nonparsable words. These words are nonparsable because our original parser cannot generate a complete parse tree based on the input spelling / phonemic pronunciation, and hence no output is produced. We have found that omitting the layer of broad classes from our hierarchy would help improve performance, but resulted in a nonparsable percentage of 6% for spelling-to-pronunciation and 4% for pronunciation-to-spelling generation, both from the development test set. Some nonparsable examples spelling-to-pronunciation are: "cocktail," "settling" and "stubborn". Some nonparsable examples for pronunciation-to-spelling generation are: "aesthetic," "nevertheless" and "segregation." We have augmented our parser with a "robust parsing" capability in order to deal with these problematic words. We will first describe the causes of parse failure, followed by the architecture of our robust parser and finally the performance improvement brought about by robust parsing.

7.1 The Causes of Parse Failure

The sparse training data problem is more pronounced as we descend the hierarchy, because terminal categories and nonterminal categories at the lower levels are more specific. For example, the syllable-part category [ONSET] consists of subcategories including all the consonant phonemes and grapheme/letter terminals that occur in the syllable onset position. Hence, there is a conglomerate of statistics in the training data that relates to the [ONSET] category and therefore the associated probabilities should be quite robustly trained. Comparatively speaking, a grapheme terminal such as #[que] in "critique" has a higher degree of specificity and occurs less frequently. Other specific categories may not appear in the training data at all.

Hence it may be apparent that the main cause of parse failure is due to zero advancement probabilities, i.e.

$$P(RS = \textit{next terminal} \mid LH = \textit{current history}) = 0 \tag{5}$$

This often occurs at the word boundary of compound words, e.g. "cocktail" and "everyday," because our training set consists predominantly of simple words. A second failure case is that of new grapheme terminals, e.g. for the word "sioux," pronounced as /s uw/, the last four letters "ioux" in the spelling should correspond to the second phoneme in the pronunciation. The third failure case is simply due to sparse training data. For example, in "tightly", parse failure is caused by the zero advancement probability from the column history {WORD ROOT SSYL1 ONSET /t/ #[t]} to the next grapheme terminal #[igh]. The symmetric case of pronunciation-to-spelling generation yielded spelling outputs of "titely", "tytely" and "teitly."

7.2 The Robust Parsing Strategy

The top-level architecture of our robust parser is shown in Figure 6. It contains the basic modules of bottom-up prediction, left-to-right advancement and a stack, which is similar to the original parser. We have augmented the regular path in left-to-right advancement with three other options, labelled as "end-start," "skip," and "partial-history," in order to handle the three causes of parse failure as described previously.

Typically, when a column advances to the next terminal, each of the four options can be selected with finite probability. The values of these probabilities are empirically set to sum to 1, and to favor the regular advancement path. The respective values are: P(regular) = 0.94, P(end-start) = 0.025, P(skip) = 0.025, P(partial-history) = 0.01.

Referring to the robust parsing strategies in Figure 6:

- END-START deals with compound words. It allows the parser to end a parse tree in the middle of a word spelling or phonemic transcription and start a new parse for the remaining part of the input.

- SKIP deals with geminate letters, e.g. #[gg] in "luggage" and #[ss] in "embassy." Consonant phonemes can be mapped to their corresponding letter terminals or geminate letter terminals. The robust parser allows skipping of one of the two geminate letters, and the word is parsed as though the gemination is replaced by a single letter.[13]

- PARTIAL slackens constraints when the parser encounters zero advancement probabilities outside the two previously mentioned conditions. This is accomplished by reducing the Left-History to include probabilities only on the syllable part level. Such an advancement will also incur a penalty of P(*partial*). For example in "tightly",

$$P(\#[\text{igh}] \mid \{\text{WORD ROOT SSYL1 ONSET } /\text{t}/ \ \#[\text{t}]\} = 0$$

but

[13] One may wonder if allowing the SKIP option would lead to errors such as treating "wagged" as "waged". In our modeling, the long vowel in "waged" is followed by the terminal #[ge]. The case is similar to "defining" in Figure 2. The short vowel in "wagged" is modeled differently, as it is followed by the terminal #[g]. This difference in modeling can avoid confusion of one word for the other in our analysis.

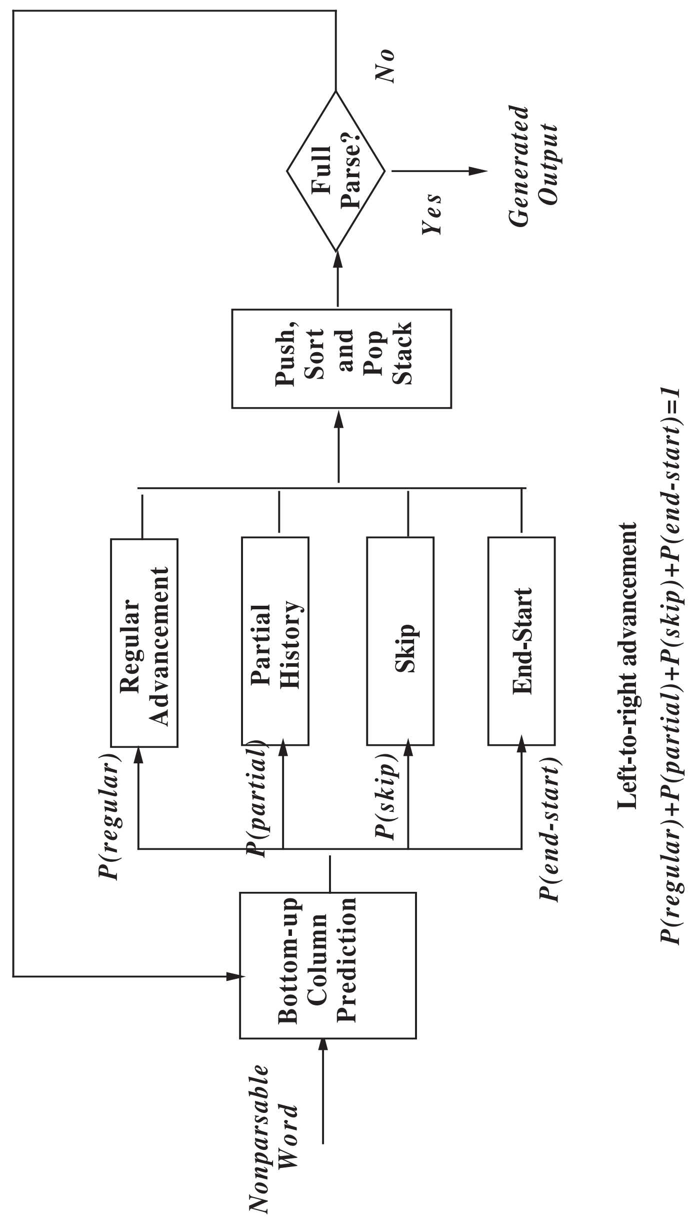

Figure 6: Top-level architecture for the robust parser.

$$P(\#[\text{igh}] \mid \{\text{WORD ROOT SSYL1 ONSET}\}) \; > \; 0$$

The advancement from {WORD ROOT SSYL1 ONSET /t/ #[t]} to #[igh] incurs both the "backoff" advancement probability as well as the "backoff" penalty, i.e.,

$$P_{partial\ history\ advancement} \;=\; P(\#[\text{igh}] \mid \{\text{WORD ROOT SSYL1 ONSET}\}) * P(partial)$$

7.3 Robust Parser Performance

The additional advancement options in the robust parser enable it to propose and explore many more theories at a given instant than the original parser. Therefore, due to efficiency concerns, we revert to the robust parser only when the original parser fails.

This new coupled configuration has attained complete coverage of the development test set, and brought about a slight performance improvement in the development test set as tabulated in Table 9. The nonparsable words in the previous experiments (which did not attain complete coverage) had been given a word accuracy and phoneme/letter accuracy of zero.[14]

[14]For the results in this section we have omitted the layer of broad classes from our hierarchical framework. Experiments based on the development test set has shown that this leads to higher performance. However, the corresponding percentage of nonparsable words in the development test set for pronunciation-to-spelling generation rose from 4% to 5%, while that for spelling-to-pronunciation generation remains the same.

Task	Coverage	Word Accuracy	Phoneme/Letter Accuracy
Spelling-to-pronunciation (original)	94%	67.5%	87.6%
Spelling-to-pronunciation (robust)	100%	69.2%	91.3%
Pronunciation-to-spelling (original)	95%	52.9%	83.8%
Pronunciation-to-spelling (robust)	100%	53.7%	88.5%

Table 9: Performance improvement on the development test set with the addition of robust parsing. Zero accuracies were given to nonparsable words.

Task	Coverage	Word Accuracy	Phoneme/Letter Accuracy
Spelling-to-pronunciation (original)	93%	65.0%	86.9%
Spelling-to-pronunciation (robust)	100%	66.3%	90.5%
Pronunciation-to-spelling (original)	94%	51.0%	82.6%
Pronunciation-to-spelling (robust)	100%	52.0%	87.9%

Table 10: Performance improvement on the test set with the addition of robust parsing. Zero accuracies were given to nonparsable words.

The performance on the test set is tabulated in Table 10. Using only the original parser, about 7% of the real test set was nonparsable in spelling-to-pronunciation generation, and the corresponding value for pronunciation-to-spelling generation was 6%. Complete coverage was attained with the inclusion of the robust parser.

In general, the word accuracies on the development test set are about 2-3% higher than those of the test set, and the phoneme/letter accuracies are about 1% higher. Similar trends are observed in *both* test sets when robust parsing is incorporated. Robust parsing has brought about slight improvements in word accuracies (about 1%), phoneme accuracies (about 3%) and letter accuracies (about 5%). The increment is small because the nonparsable words seem to be a difficult subset. Analysis of the nonparsable words in the development test set shows that the robust parser achieved a 28% word accuracy and 69.5% phoneme accuracy in spelling-to-pronunciation generation. Similar analysis of the nonparsable words in the development test set from pronunciation-to-spelling showed that robust parsing gave a 15.6% word accuracy and 75.5% letter accuracy. The robust parse trees for the word "henrietta" in both generation tasks are included as shown in Figures 7 and 8.

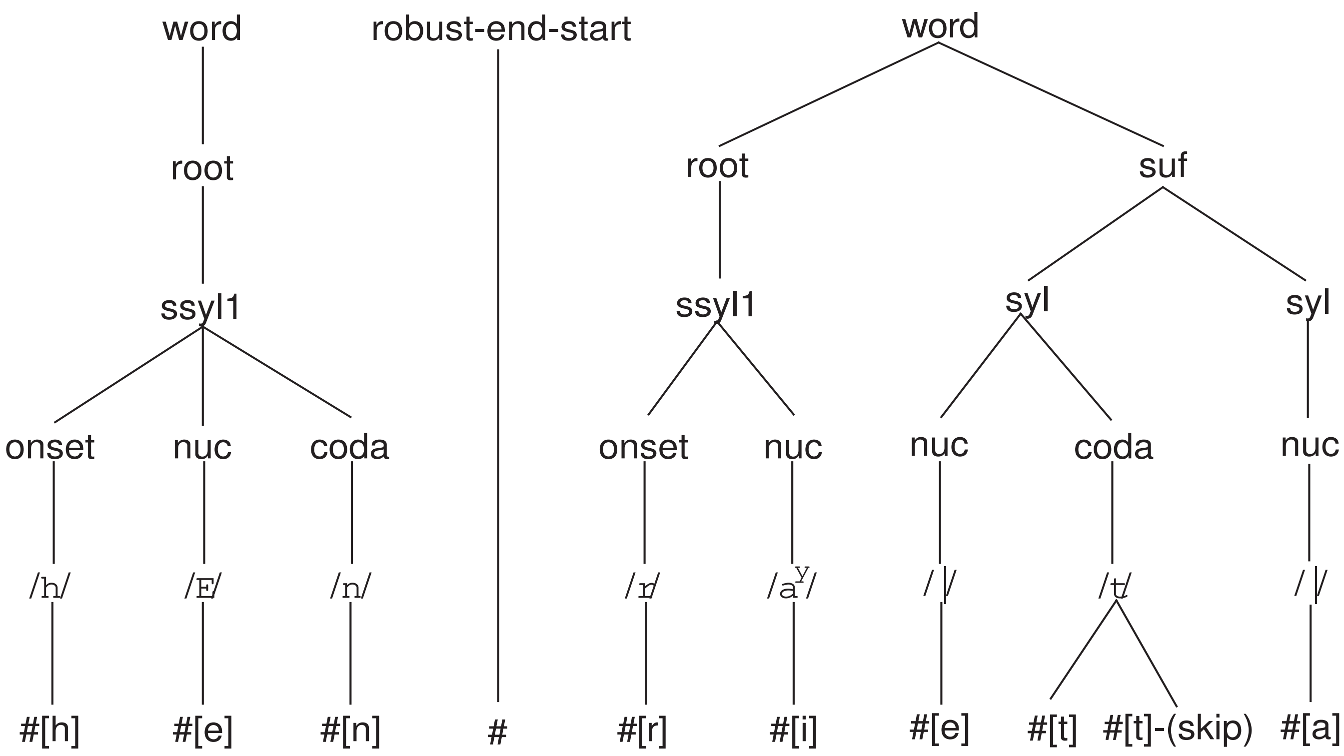

Figure 7: Parse tree for the word "henrietta" from robust spelling-to-pronunciation generation.

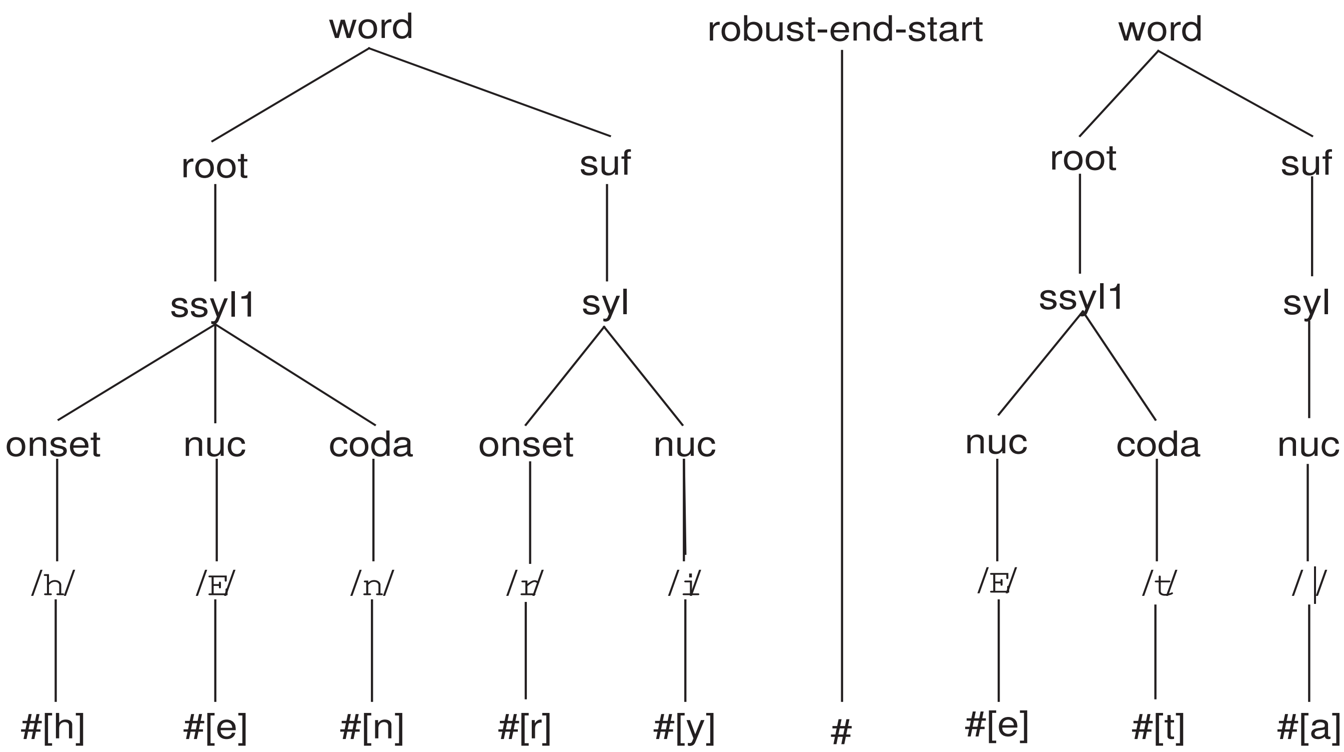

Figure 8: Parse tree for the word "henrietta" from robust pronunciation-to-spelling generation.

8 Extending the Hierarchy

We have thus far focused on demonstrating how our hierarchical lexical representation can be used for bi-directional spelling/pronunciation generation. However, we would also like to take the first step to demonstrate how our hierarchy is extendable, by adding a layer of phones beneath the layer of phonemes in the hierarchy. As a consequence, the terminal layer becomes dual in nature. The phonetic and graphemic layers both reside in the terminals. Using the augmented hierarchical representation with the layered bigram probabilities, we are able to capture some dialectal and phonological variations both *within* a word and *across word boundaries*.

We intend to use our extended framework to capture phonological constraints. This is a potential alternative to *ordered rewrite rules* used for explaining context dependencies for phoneme realizations. [31] [53]. A previous attempt has been made which attaches probabilities to the rules productions [52]. The probability of a given rule reflects the number of times the rule is actually applied, normalized over the number of times the rule can be applied. Each rule is compiled to form a series of rule clauses specifying the rule's transformations on a phone lattice under different contexts. Upon rule application, the rule probability will be incorporated into the arc probabilities of the phone lattice. However, these rule probabilities assume that rule applications are independent of one another, which is not valid for *ordered* rules. Instead, these production probabilities should be dependent on the preceding rule productions, which may perhaps make the probabilities more complicated and less trainable.

For this pilot study, we concentrated our efforts on a small set of data carefully selected for potential validation of our approach. Rather than complicating the experiment with a recognition task, we have selected as our experimental corpus the "sa" sentences of TIMIT, for which carefully transcribed phonetic labels are available. These sentences are designed especially for the study of dialectal variations. They also provide phoneme-to-phone alignments for generating our training parse trees, and the phonetic transcriptions are labelled manually through listening tests along with visual aids of the spectrogram and waveform [19]. The two "sa" sentences are:

1. "She had your dark suit in greasy wash water all year."

2. "Don't ask me to carry an oily rag like that."

The speakers come from 8 different dialectal regions,[15] namely, New England, Northern, North Midland, South Midland, Southern, New York City, Western and

[15]This is the geographical dialect area where the subject spent the most years between ages 0 and 10.

```
            U                    I
            O        kcl         i s          F
            5        k           | z          t
S i h @ d y U r d a r k s u t I n g r i s i w O S w O t 5 O l y I r
        dcl y              F             Or
        d   y              ?             U
        J   y              tcl           5
                           t             a
                                         ^
```

Figure 9: Some phonological variations occurring in the sa-1 training sentences — "She had your dark suit in greasy wash water all year." *dcl* and *kcl* denote d-closure and k-closure respectively.

"Army Brat." In our study, we use 362 training speakers, and a disjoint set of the 24 NIST designated "core" test speakers. The test set consists of 3 speakers from each of the 8 dialectal regions.

8.1 Phonological Variations

The two "sa" sentences are especially designed for the study of phonological variations. In this subsection, we provide a brief description of such variations as observed in our training data.

Some of the phonological variations in the "sa-1" sentences are illustrated in Figure 9. For example, on either side of the word boundary between "had" and "your," the phoneme /d/ transitioning to the phoneme /y/ [16] Therefore, we observe cases where the *alveolar* stop /d/ is *palatalized* to become either the affricate [ch] or [jh] followed by the semi-vowel /y/, which may be deleted. Another example concerns the vowel in "wash." It can be realized as [ao], [uh], [ah], [er], or [aa]-[r], with retroflexion inserted in the last two cases.

Similarly, we find much variation in the "sa-2" sentences, as illustrated in Figure 10. On either side of the word boundary between "Don't" and "ask," the phoneme /t/ transitioning to /ae/ can be realized as /ae/ can be realized as [t]-[ae], [tcl]-[ae], *glottal stop*-[ae], [null]-[aa], [null]-[eh], [null]-[aw], [null]-[ae], [null]-[ey] and [d]-[ae]. Here we observe several different allophones for the phoneme /t/ in "don't." It may be *released* in the phone [t] or [d], *unreleased* in [tcl], *glottalized* to become a *glottal*

[16] [null] denotes the NULL phone.

```
                                                   k T
       t           t                               k D
       null        F                               kcl D
       ?           null                            kcl d
d o n  t @ s k m i t u k @ r i @ n o^y l i r @ g l a^y k D @ t
             k                     o^y             a^y
             kcl                   o               a
             null
```

Figure 10: Some phonological variations occurring in the sa-2 training sentences — "Don't ask me to carry an oily rag like that." *tcl* and *kcl* denote t-closure and k-closure respectively.

stop, or deleted as in [null]. The subsequent vowel /ae/ may also take many different forms. Finally, the vowel in "like," /ay/, may assume various identities, including [ae], [ay], [iy], [aa] and [ah].

We would like to capture such dialectal and phonological variations in the "sa" sentences, by extending *both* the hierarchical lexical representation and the layered bigrams framework. Our experiments attempt to parse the phonetic transcription (with no indication of the word boundary locations) of an "sa" sentence. The purpose is to obtain the sequence of words in the sentence without prior knowledge of the lexicon.

8.2 Extended Parse Trees

The two "sa" sentences together consist of 21 distinct words which constitute the vocabulary of our current task. For each word in our limited vocabulary, we generate a parse tree from the word level down to the phoneme level, using our letter-to-sound generation system. The parse tree is then extended to the phone level based on the phoneme-to-phone alignments provided by TIMIT. The parse tree for an *entire* training sentence is then obtained by concatenating the parse trees for each *word* in the sentence. An example of the training parse tree of an "sa-2" sentence is shown in Figures 11 and 12.

The extended parse tree has the generic category SENTENCE at the top level, followed by a layer of WORD categories separated by the word boundary terminal #. Some of the phonological variations described in the previous subsection can

[h]

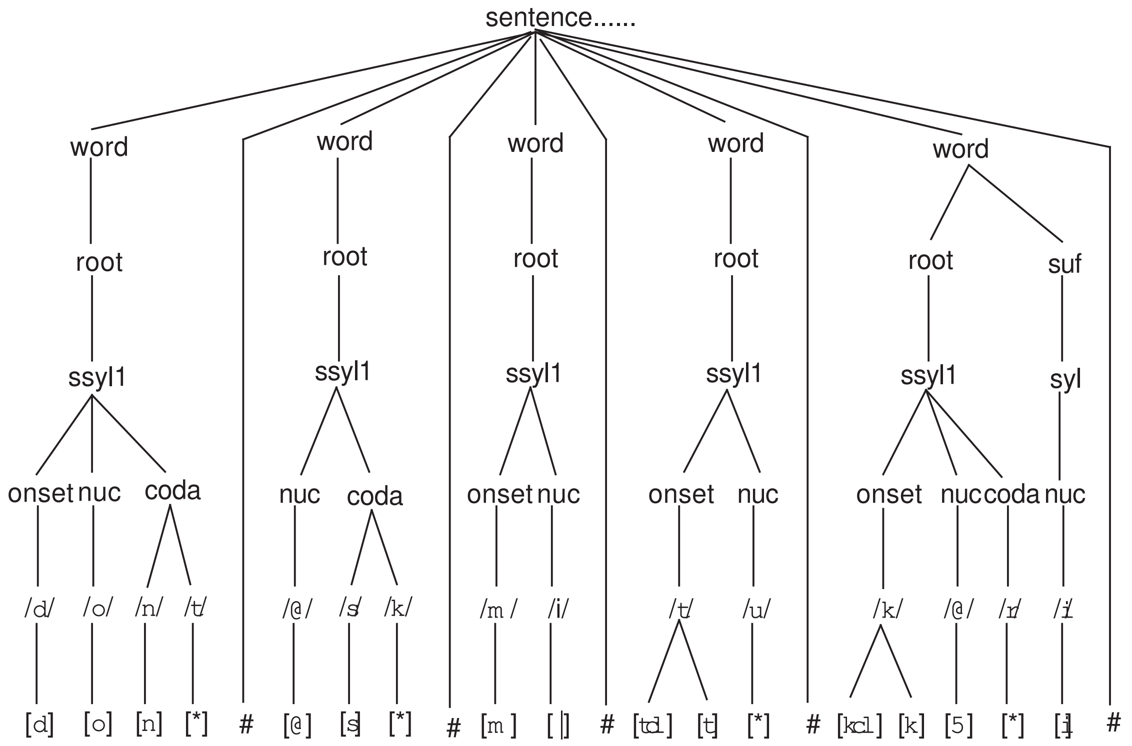

Figure 11: The sentence parse tree of the first half of a training "sa-2" sentence — "Don't ask me to carry..." — with a terminal phonetic layer.

be observed from this sentence parse tree. For example, the phoneme /t/ in the word "don't," and the phoneme /k/ in the word "ask" have been deleted, and the vowel /uw/ in the word "to" has been devoiced. In the word "carry," the phoneme /k/ illustrates the case where one phoneme is mapped to multiple phones [kcl] and [k]), and the phone [er] illustrates the case where one phone corresponds to multiple phonemes. Phonological rules specify variations in the arena of the two bottom layers of phonemes and phones, and are extracted automatically and implicitly during training by the layered bigram probabilities.

8.3 Extended Parser

The layered bigrams are extended to parse the phonetic transcription of an *entire* sentence into a sentence parse tree depicted in Figures 11 and 12. The main augmentations encompass training, robust parsing during testing, as well as lexical access.

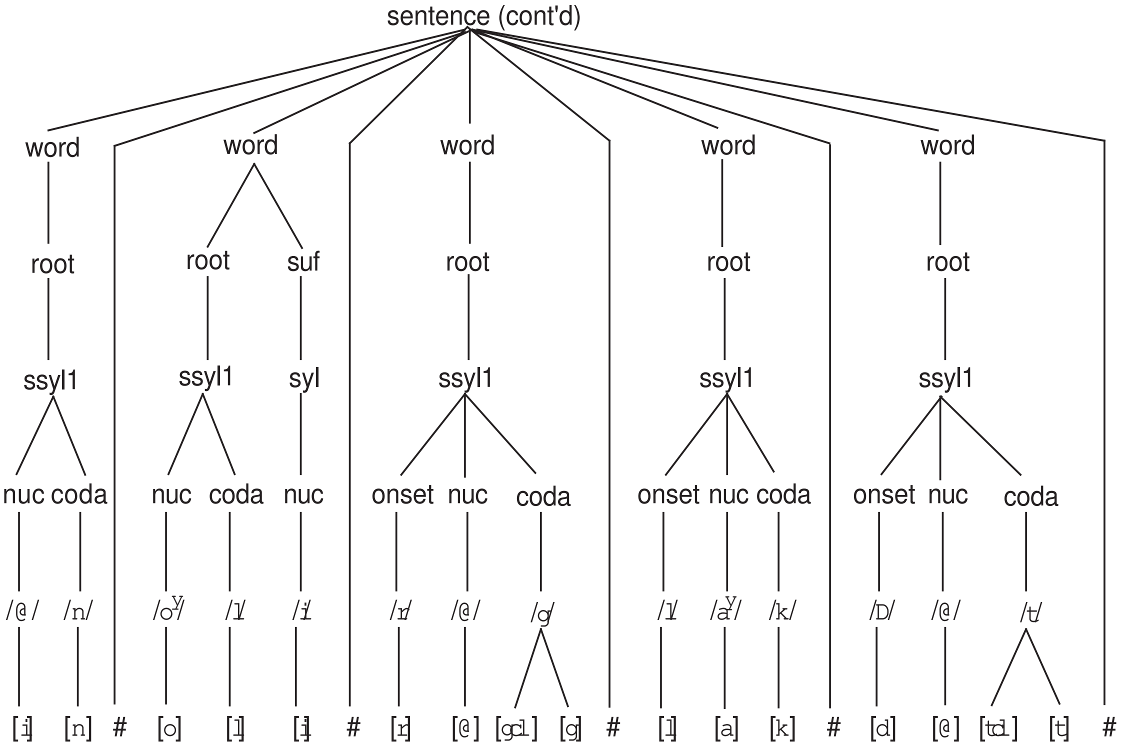

Figure 12: The sentence parse tree of the second half of a training "sa-2" sentence — "...an oily rag like that." — with a terminal phonetic layer.

1. In training, we have created the special terminal # to denote a *word boundary*. This terminals is predicted in just the same way as other terminals, using the left-history. However, if the current child is #, then the current history does not provide much information for the prediction of the following terminal. In this case, the prediction of the phoneme-phone pair to the *right* of a word boundary is conditioned upon the phoneme-phone pair to the *left*.

2. Robust parsing is used during testing, similar to that described in the previous section. Testing amounts to creating a complete parse tree for the phonetic transcription of a test "sa" sentence. Each phonetic transcription of TIMIT is a single phone sequence. This is unlike the output of a speech recognizer which provides a phone lattice with alternative phone choices for each time segment. Phone sequences which have not been previously observed in the training data lead to parse failures, unless alternate phone choices are provided. In the event of a partial theory not being able to advance to the next input

phone (i.e. the left-to-right advancement probability equals zero), the robust parsing capability allows skipping the phone, and ending the partial parse with a "robust-end" node which incurs a penalty score. The "robust-end" node is subsequently pushed onto the stack. When this node is later popped off the stack for extension, the robust parser proceeds by starting a new parse for the remaining part of the input.

Figure 13 shows an example of a sentence parse tree which contains skipped phones. The parse shows the first half of an "sa-1" sentence, whose phonetic transcription is missing a vowel for "She," and missing retroflexion in the word "your." At both locations, we see a skipped phone, followed by a "robust-end" node, and the parse is continued with a "robust-start" node. Figure 14 shows the rest of the sentence.

3. Lexical access is a straightforward table-lookup procedure. For each of the 21 words in the vocabulary, the table stores the spelling and a *single* phoneme sequence as the pronunciation. Given a complete sentence parse tree, lexical access involves extracting the phoneme sequences between adjacent WORD-END or ROBUST-END nodes, and mapping each phoneme sequence into a word spelling. For example, referring to Figure 13, the first three phoneme sequences are /sh/, /ae/, /d/ and /y/ respectively, none of which can map to any word. The fourth phoneme sequence, /d aa r k/, maps to "dark," and so forth. Therefore, lexical access based on this parse tree outputs the word sequence "NIL NIL NIL dark suit in greasy wash water all year."

8.4 Captured Phonological Variations

The phonological variations captured using the layered bigrams can be observed by examining the trained probabilities. Three types of variations have been brought into consideration:

1. Allophonic variations: This refers to the different realizations of a phoneme in the context of the neighboring phones/phonemes. For example, the phoneme /t/ can be found in the words "suit," "water," "don't" and "that," in the context of {WORD ROOT SSYL1 CODA} — the coda of a stressed syllable in a root morph. According to our training data, this phoneme can be realized as several different phones — *glottal stop*, [dx], [dcl], [d], [tcl], [t] and [null]. Our parser also favors the various selections to different extents due to trained probabilities.

2. Across-word phonological variations: This accounts for the changes in a word pronunciation depending on the neighboring words. One example is the pho-

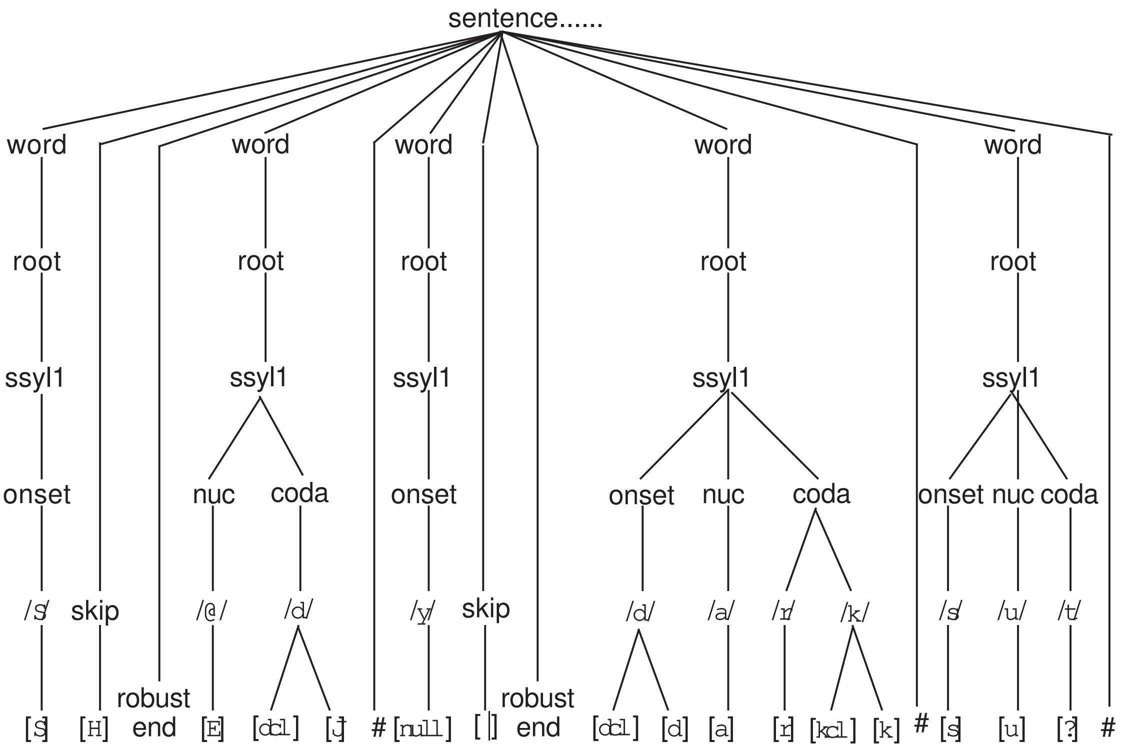

Figure 13: Example of the first half of an sa-1 sentence parse tree which contains skipped phones — "She had your dark suit..."

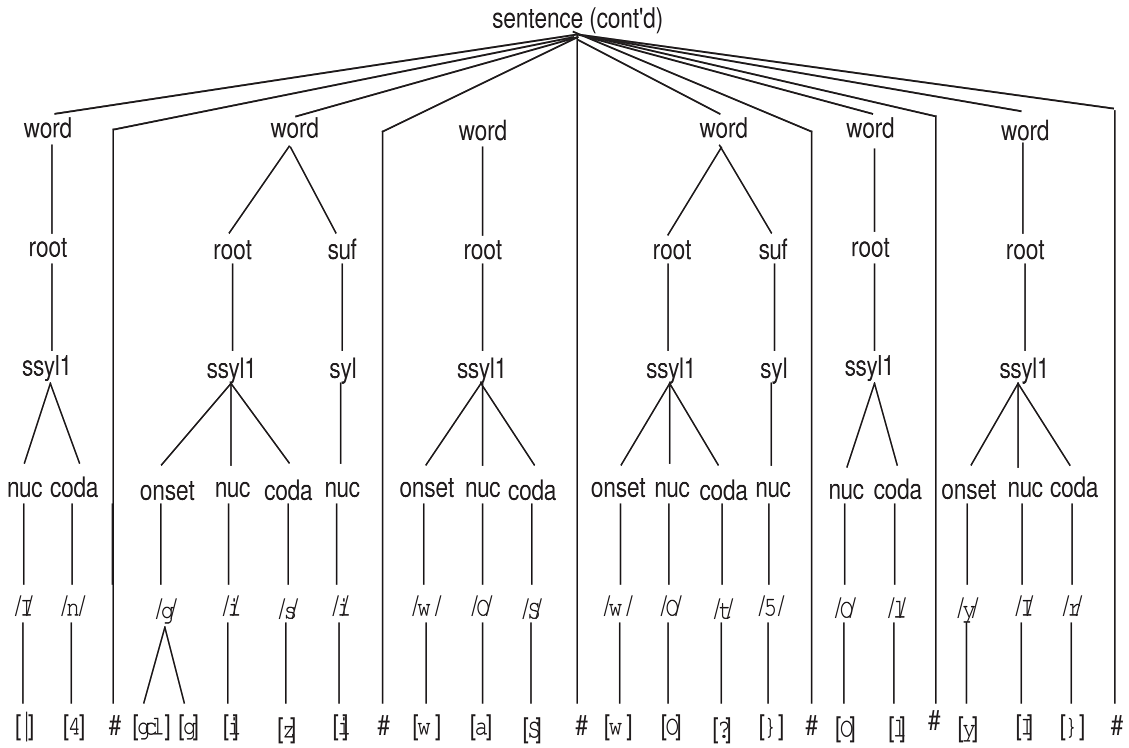

Figure 14: Example of the second half of sa-1 sentence parse tree — "...in greasy wash water all year."

netic realizations of /d/- /y/ occuring on either side of the word boundary "had your" in the "sa-1" sentence. We found seven distinct phonetic realizations from our training data: [null]-[y], [dx]-[y], [dcl]-[y], [d]-[y], [ch]-[null], [jh]-[y] and [jh]-[null]. Among these seven options, palatization of /d/-/y/ to become [jh]-[null] is predominant, and occurred 96% of the time.

3. Within-word phonological variations: This contributes towards alternative realizations/pronunciations for words, and is captured within the 3-tuple of left-history, right-phone and right-phoneme. For example, in the word "like", where left-history is {WORD ROOT SSYL1 ONSET /l/ [l]} and righ-phoneme is /ay/, the right-phone yields the allophones in this context: [ah], [aa], [iy], [ae] and [ay], where [ay] is the most common realization.

8.5 Capturing Phonological Rules

To illustrate how the layered bigrams capture a typical phonological rule, such as the *flapping* of /t/, we should examine the contrastive word pairs "don't ask" (in Figure 11) and "suit in" (in Figures 13 and 14). Flapping of the phoneme /t/ generally occurs in an intervocalic position. Thus we should observe flapping across the word pair "suit in," but not across "don't ask."

Referring to Figure 11, we see in the word "don't," the phoneme /n/ is realized as the phone [n] in the column context of {WORD ROOT SSYL1 CODA /n/ [n]}. The realization of the subsequent phoneme /t/ is thus dependent on the column-advancement probabilities from the given column context. These probabilities are tabulated in Table 11.

Left Column History (LH)	Right Sibling (RS)	Probability $P(RS\|LH)$
{WORD ROOT SSYL1 CODA /n/ [n]}	[null]	0.5202
{WORD ROOT SSYL1 CODA /n/ [n]}	[t]	0.0202
{WORD ROOT SSYL1 CODA /n/ [n]}	[tcl]	0.1768
{WORD ROOT SSYL1 CODA /n/ [n]}	*glottal stop*	0.2828

Table 11: Column advancement probabilities to show possible phonetic realizations of the phoneme /t/ in the word "don't".

Since the phoneme /t/ in the word "don't" has the nasal [n] in its left history, the context is not intervocalic, and we do not expect to see *flapping* of /t/. Indeed, the trained probabilities displayed in Table 11 shows that given the left history, /t/ can be realized as [null], [tcl], [t] or *glottal stop*, but has a zero probability of *flapping*. The probabilities thus agree with our expectation.

However, if we inspect the across-word prediction probabilities with the flapped /t/ as the left context, i.e. ($P_{across-word\ prediction}(right\ phoneme - phone \mid$ /t/$-$[dx])), we observe probability values as tabulated in Table 12.

Left phoneme-phone pair	Right phoneme-phone pair	Across-word prediction probability $P(right\ phonme - phone \mid /t/ - [dx])$
/t/-[dx]	/ih/-[ih]	0.4242
/t/-[dx]	/ih/-[iy]	0.0152
/t/-[dx]	/ih/-[ix]	0.5606

Table 12: Across-word prediction probabilities to show right phoneme-phone transitions from the left phoneme-phone pair (/t/-[dx]).

Table 12 shows that when a phoneme /t/ is *flapped* to the left of a word boundary, it will only transition to a right phoneme /ih/. This indicates that the trained probabilities are derived entirely from the word pair "suit in" in the sa-1 training sentences. This is also consistent with our expectation, that the /t/ may be flapped due to an intervocalic context.

The gist of this subsection is to illustrate how the layered bigram probabilities can encode phonological rules, using the *flapping* rule as an example. Flapping is prohibited for the /t/ in "don't" because the nasal in the left-history shows that the context is *not intervocalic*. The word pair "suit in" provides an *intervocalic*, *across-word* context, and therefore flapping is possible. We believe that the probabilities used for parsing should help prioritize the applications of different phonological rules. This should be verified by conducting experiments using larger data sets.

8.6 Word and Sentence Accuracies

We measured both the word accuracies and sentence accuracies for our test set. A sentence is correct if lexical access yields a word sequence identical to the reference sentence. Figure 15 plots the performance values as the amount of training data is increased. As we increase the number of training speakers from 20 to 362 (the full set), the word accuracies for the "sa-1" and "sa-2" sentences rose by about 9% and 6% respectively, while the sentence accuracies increased by roughly 22% and 30% respectively. This indicates that the layered bigrams can capture more phonological variations from wider exposure to training data, leading to improvements in performance.

Most of the errors in the sentence parse trees are due to sloppy articulation in continuous speech. We include two examples of errors from testing the "sa-1" sentences,

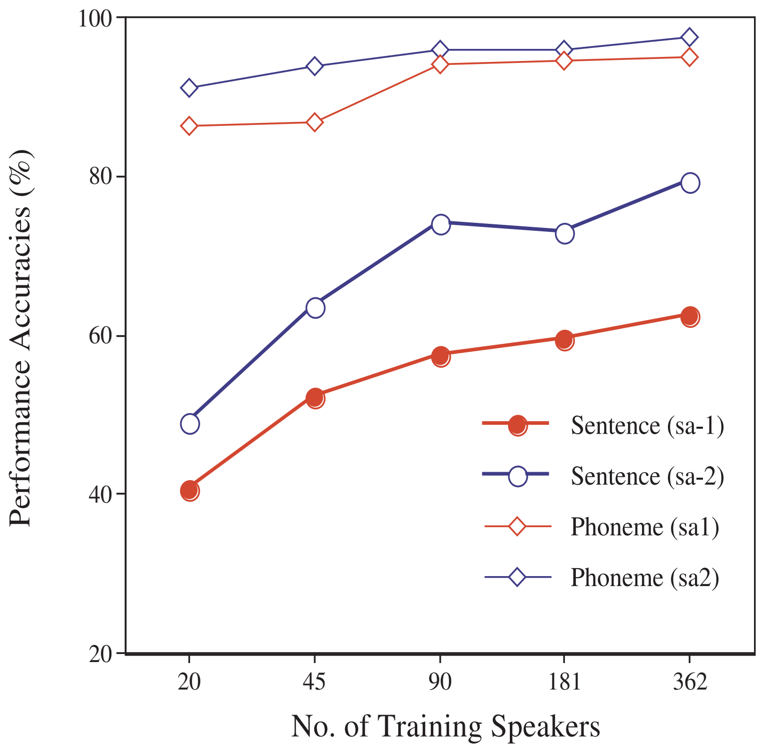

Figure 15: Word and sentence accuracies of the layered bigrams in parsing sentences, plotted as a function of increasing training data.

having trained on the full training set.

- The input phone sequence was:
  [sh] [iy] [hv] [ae] [dcl] [d] [y] [ih] [dcl] [d] [aa] [r] [kcl] [k] [s] [ux] [dx] [ih] [n] [gcl] [g] [g] [r] [iy] [s] [iy] [w] [ao] [sh] [w] [ao] [dx] [er] [ao] [l] [y] [ih] [er]
  Lexical access did not yield the word "your" due to the missing retroflexion.

- The input phone sequence was:
  [sh] [iy] [ae] [dcl] [jh] [axr] [dcl] [d] [aa] [r] [kcl] [k] [s] [ux] [tcl] [eng] [gcl] [g] [r] [iy] [z] [iy] [w] [aa] [sh] [w] [ao] [dx] [er] [ao] [y] [ih] [axr]
  Error occurred with the word "all" due to the missing phone [l].

Hence we have demonstrated the feasiblity of extending our hierarchical lexical representation to the phonetic level, for capturing phonological rules. Our parsing process is permissive of dialectal variations and coarticulatory effects like palatalization and flapping. We believe our current paradigm can be used for other speech applications as well.

9 Summary and Future Work

Our current work demonstrates the use of a hierarchical framework, which is relatively rich in linguistic knowledge, capable of bi-directional spelling-to-phonemics and phonemics-to-spelling generation. The framework encompasses linguistic information such as morphology, stress, syllabification, phonemics and graphemics, each occupying a distinct layer in the hierarchy. We have chosen to evaluate our system design based on the task of bi-directional spelling-to-pronunciation and pronunciation-to-spelling generation. Our training and test sets are derived from the high-frequency portion of the Brown Corpus (10,000 words). The percentage of nonparsable words for spelling-to-pronunciation and pronounciation-to-spelling generation are 7% and 6% respectively. In spelling-to-pronunciation generation, we have achieved a word accuracy of 65.0% and a phoneme accuarcy of 86.9%, which is competitive with the other automatic generation systems. In pronunciation-to-spelling generation, we have achieved a word accuracy of 51.0% and a phoneme accuarcy of 82.6%, which compares favorably with other similar systems.

In order to strive for complete coverage in generation, we have also introduced a robust parsing strategy, which employs multiple methodologies in relaxing constraints during parsing. Our baseline system, augmented with the robust parser, achieved 100% coverage for both tasks, and also a 66.3% word and 90.5% phoneme accuracy for spelling-to-pronunciation generation, and 52.0% word and 87.9% letter accuracy for pronunciation-to-spelling generation.

The task of bi-directional spelling / pronunciation generation is a testbed for studying the use of a hierarchical lexical representation, which is a central idea in this work. We have thus far seen that when hierarchical sublexical modeling is used in conjunction with a probabilistic parsing strategy, it becomes possible to automatically derive the morphological decompositions of an input word. Referring back to Figure 2, if we simultaneously walk horizontally across the second layer and the terminal layer of the parse tree, we see that the word "defining" consists of a prefix ("de"), a root ("fine") with a deleted letter 'e', and an (inflectional) suffix ("ing"). The notion of a *morph*, according to our framework, is a syllable-sized unit which may have linguistic meaning. Knowledge about English morphology and etymology, and the availability of morphologically-marked lexicons will certainly benefit our system. However, there is much ambiguity in determining the morphological decompositions of English words. We have not found any large-scale lexicons that are directly applicable for our work. As a consequence, we have devoted effort in marking our own lexicon.

We believe that the morph is an important subword unit that is particularly useful for large-vocabulary speech recognition.[17] As an example, consider a training lexicon with the words "connect," "connecting" and "connections" – they all share a common root morph *connect*. Instead of modeling each of these inflectional form individually, we can avoid splitting the training data by sharing their common subword structure, and in so doing create better trained acoustic models. Furthermore, if we concatenate the root morph *connect* with inflectional suffixes (e.g. 's' and 'ed') derived from other in-vocabulary words, we can automatically obtain the inflectional forms "connects" and "connected", that may be missing from the training data. The morph is a lingusitically-motivated subword unit, which can be used for structural sharing to make large-vocabulary speech recognition more manageable, and for pooling training data across different words for better acoustic training. The morph also enables the derivation of new words from existing words in the vocabulary, and therefore offers a potential solution for handling out-of-vocabulary (OOV) words.

It is important to note that our sublexical hierarchy is extendable to encompass natural language constraints, prosody, discourse and perhaps even dialogue modeling constraints on top, as well as phonetics and acoustics at the bottom. In this paper, a pilot study has been included to demonstrate how phonological rules can be captured. As regards prosody, we expect the lingustic knowledge in our hierarchical representation to be extremely useful for duration modeling for speech recognition and synthesis. For example, vowels within the context of stressed roots should have different durations when compared to their counterparts; and consonents in an onset

[17] To our knowledge, past approaches which utilize morphs in speech technology development are few. The ones that we are aware of include MITalk [2] for text-to-speech synthesis in English, as well as the Speech Maker Formalism [50] for text-to-speech synthesis in Dutch.

position should have durations different from their non-onset counterparts.

This work is the precursor of the ANGIE system, which further develops the notion of sublexical modeling for various speech technologies, including speech recognition and duration modeling. Initial work using the ANGIE system has shown that the sophisticated sublexical modeling leads to improvements in phonetic recognition. The experiments compares the use of a baseline phone bigram with ANGIE's modeling where the sublexical layers above the phone are also considered. Results show a reduction of phone recognition error rate from 40% to 36% [36]. Experiments which utilize the sublexical hierarchy for duration modeling has also led to improvements in word-spotting [6] [20].

Acknowledgments
This research is part of the author's doctoral thesis work, supervised by Dr. Victor W. Zue and Dr. Stephanie Seneff.

References

[1] Allen, J., "Synthesis of speech from unrestricted text," *Proc. IEEE*, vol. 64, pp. 422-433, 1976.

[2] Allen, J., S. Hunnicutt and D. Klatt, *From Text to Speech: The MITalk System,* Cambridge University Press, Cambridge, 1987.

[3] Alleva, F. and K. F. Lee, "Automatic New Word Acquisition: Spelling from Acoustics," *Proc. of the DARPA Speech and Natural Language Workshop,* October 1989.

[4] Coker, C., K. Church and M. Liberman, "Morphology and Rhyming: Two Powerful Alternatives to Letter-to-Sound Rules for Speech Synthesis," *Proc. of the Conference on Speech Synthesis*, European Speech Communication Association, 1990.

[5] Conroy, D., T. Vitale and D. Klatt, *DECtalk DTC03 Text-to-Speech System Owner's Manual,* Educational Services of Digital Equitpment Corporation, P.O. Box CS2008, Nashua, NH 03061, 1986. Document number EK-DTC03-OM-001.

[6] Chung, G. and S. Seneff, "Hierarchical Duration Modeling for Speech Recognition using the ANGIE framework," *Proc. ICSLP-97*, pp. 1475-1478, 1997.

[7] Damper, R., "Self-Learning and Connectionist Approaches to Text-Phoneme Conversion," *Proc. of 2nd Neural Computation and Psychology Workshop,* edited by J. Levy, Forthcoming.

[8] Dedina, M. and H. Nusbaum, "PRONOUNCE: A Program for Pronunciation by Analogy," *Computer Speech and Language*, Vol. 5, No. 1, pp. 55-64, January 1991.

[9] Elovitz, H., R. Johnson, A. McHugh and J. Shore, *Automatic Translation of English Text to Phonetics by means of Letter-to-Sound Rules,* Naval Research Laboratory Technical Report 7949, January 1976.

[10] Fukada, T., Y. Sagisaka, 'Automatic Generation of a Pronunciation Dictionary Basd on a Pronunciation Network," *Proc. Eurospeech 1997*, pp. 2471-2474, 1997.

[11] Golding, A., *Pronouncing Names by a Combination of Case-based and Rule-based Reasoning,* Ph.D. Thesis, Stanford University, 1991.

[12] Golding, A. and P. Rosenbloom, "A Comparison of ANAPRON with Seven Other Name-pronunciation Systems," *Journal of the Americal Voice I/O Society,* August 1993, pp.1-21.

[13] Hochberg, J., S. M. Mniszewski, T. Calleja and G. J. Papcun, "A Default Hierarchy for Pronouncing English," *IEEE Transactions on Pattern Matching and Machine Intelligence*, Vol. 13, No. 9, pp. 957-964, September 1991.

[14] Jiang, L., H. Hon and X. Huang, "Improvements on a Trainable Letter-to-Sound Generator," *Proc. Eurospeech 1997*, pp. 605-608, 1997.

[15] Klatt, D. and D. Shipman, "Letter-to-Phoneme Rules: A Semi-automatic Discovery Procedure," *J. Acoust. Soc. Am.,* 82, pp.737-793, 1982.

[16] Klatt, D., "Review of Text-to-speech Conversion for English," JASA 82 (3), Acoustic Society of America, pp. 737-793, 1987.

[17] Kucera, H. and W. N. Francis, *Computational Analysis of Present-Day America English*, Brown University Press, 1967.

[18] Kuhn, R., J. Junqua and P. Martzen, "Rescoring Multiple Pronunciations Generated from Spelled Words," *Proc. ICSLP-98*, 1998.

[19] Lamel, L. F., R. H. Kassel and S. Seneff, "Speech Database Development: Design and Analysis of the Acoustic-Phonetic Corpus," *Proc. DARPA Speech Recognition Workshop*, Report No. SAIC-86/1546, pp. 100-109, February 1986.

[20] Lau, R. and S. Seneff, "Providing Sublexical Constraints for Word Spotting within the ANGIE Framework," *Proc. Eurospeech 1997*, pp. 263-266, 1997.

[21] Lucas, S. and R. Damper, "Syntactic Neural Networks for Bi-directional Text-phonetics Translation," in *Talking Machines, theories, models and designs,* G. Bailly and C. Benoit (Eds.), pp. 127-142, North-Holland Publishers, 1992.

[22] Lucassen, J. and R. Mercer, "An Information Theoretic Approach to the Automatic Determination of Phonemic Baseforms," *Proc. ICASSP-84*, pp. 42.5.1-42.5.3, San Diego, 1984.

[23] Luk, R. and R. Damper, "Inference of Letter-Phoneme Correspondences with Pre-defined Consonant and Vowel Patterns," Proc. ICASSP-93, pp. 203-206, Minneapolis, 1993.

[24] McCulloch, N., M. Bedworth and J. Bridle, "NETspeak — a reimplementation of NETtalk," *Computer Speech and Language*, Vol. 2, pp. 289-301, 1987.

[25] Meng, H., S. Seneff and V. Zue, "Phonological Parsing for Bi-directional Letter-to-Sound/Sound-to-Letter Generation," *Proc. ARPA HLT-94*, New Jersey, 1994.

[26] Meng, H., S. Seneff and V. Zue, "The Use of Higher Level Linguistic Knowledge for Spelling-to-Pronunciation Generation," *Proc. ISSIPNN-94*, pp. 670-673, Hong Kong, 1994.

[27] Meng, H., S. Seneff and V. Zue, "Phonological Parsing for Reversible Letter-to-Sound/Sound-to-Letter Generation," *Proc. ICASSP-94*, pp. II-1 to II-4, Adelaide, Australia, 1994.

[28] Meng, H., S. Hunnicutt, S. Seneff, V. Zue, "Reversible Letter-to-Sound/Sound-to-Letter Generation based on Parsing Word Morphology," *Speech Communication*, Vol. 18, pp. 46-64, 1996.

[29] Nilsson, J., *Problem-solving methods in artificial intelligence,* Computer Science Series, McGraw-Hill.

[30] Oakey, S. and R. Cawthorne, "Inductive Learning of Pronunciation Rules by Hypothesis Testing and Correction," *Proc. IJCAI-81*, pp. 109-114, Vancouver, Canada, 1981.

[31] Oshika, B., V. Zue, R. Weeks, H. Nue and J. Auerbach, "The Role of Phonological Rules in Speech Understanding Research," *IEEE Transactions on ASSP*, Vol. ASSP-23, pp. 104-112, 1975.

[32] Pallet, D., J. Fiscus, W. Fisher, J. Garofolo, B. Lund and M. Przbocki, "1993 Benchmark Tests for the ARPA Spoken Language Program," *Proc. of the ARPA Workshop on Human Language Technologyc*, pp. 49-54, March 1994.

[33] Parfitt, S. and R. Sharman, "A Bi-directional Model of English Pronuciation," *Proc. Eurospeech*, pp. 801-804, 1991.

[34] Segre, A., Sherwood, B. and W. Dickerson, "An Expert System for the Production of Phoneme Strings from Unmarked English Text using Machine-induced Rules," *Proc. First European ACL*, Pisa, Italy, pp. 35-42, 1983.

[35] Sejnowski, T. J. and C. R. Rosenberg, "NETtalk: parallel networks that learn to pronounce English text," *Complex Systems,* 1, pp. 145-168, 1987.

[36] Seneff, S., R. Lau and H. Meng, "ANGIE: A new framework for speech analysis based on morpho-phonological modeling," *Proc. ICSLP-96*, pp. 110-113, 1996.

[37] Seneff, S., "TINA: A Natural Language System for Spoken Language Applications," *Computational Linguistics,* Vol. 18, No. 1, pp. 61-86, March 1992.

[38] Seneff, S., H. Meng and V.Zue, "Language Modeling using Layered Bigrams," *Proc. ICSLP-92*, pp. 317-320, Banff, Calgary, Canada.

[39] Shipman, D. and V. W. Zue, "Properties of large lexicons: Implications for advanced isolated word recognition systems," *Proc. ICASSP-82*, pp. 546-549, 1982.

[40] Spiegel, M. and M. Macchi, "Synthesis of Names by a Demisyllable-based Speech Synthesizer (Orator)," *Journal of the American Voice Input/Output Society,* 7, 1990. Special RHC/RBOC issue.

[41] Stanfill, C. and D. Waltz, "Toward Memory-Based Reasoning," *Communications of the ACM,* 12(12), pp. 1213-1228, December 1986.

[42] Stanfill, C., "Memory-Based Reasoning Applied to English Pronunciation," *Proc. AAAI-87*, pp. 577-581, 1987.

[43] Stanfill, C., "Learning to Read: A Memory-based Model," *Proc. of the Case-based Reasoning Workshop*, Clearwater Beach, FL, pp. 406-413, 1988.

[44] B. Suhm, M. Woszczyna, and A. Waibel, "Detection and transcription of new words," in *Proc. European Conf. Speech Communication and Technology*, pp. 2179–2182, Berlin, September 1993.

[45] Sullivan, K. and R. Damper, "Novel-word Pronunciation Within a Text-to-speech System," in *Talking Machines, theories, models and designs*, G. Bailly and C. Benoit (Eds.), pp. 183-195, North Holland Publishers, 1992.

[46] van Coile, B. M. J. "The Depes development system for text-to-speech synthesis," *Proc. ICASSP-89*, pp. 250-253, Glassgow, Scotland, May 1989.

[47] van Coile, B. "Inductive Learning of Grapheme-to-Phoneme rules," *Proc. ICSLP-90* pp. 765-768, Kobe, Japan, 1990.

[48] van Coile, B., S. Lyes and L. Mortier, "On the Development of a Name Pronunciation System," *Proc. ICSLP-92*, pp. 487-490, Banff, Canada, 1992.

[49] van den Bosch, A. and W. Daelemans, "Data-Oriented Methods for Grapheme-to-Phoneme Conversion," *Proc. Sixth European ACL*, pp. 45-53, 1993.

[50] van Leeuwen, H. C., "Speech Maker Formalism: a rule formalism operating on a multi-level, synchronized data struction, *Computer Speech and Lnaguage,* volume 7, number 4, October 1993.

[51] Yannakoudakis, E. and P. Hutton, "Generation of spelling rules from phonemes and their implications for large dictionary speech recognition," *Speech Communication*, Vol. 10, pp. 381-394, 1991.

[52] Weintraub, M., and J. Bernstein, "RULE: A System for Constructing Recognition Lexicons," Proc. DARPA Speech Recognition Workshop, Report No. SAIC-87/1644, pp. 44-48, February 1987.

[53] Zue, V., J. Glass, D. Goodine, M. Phillips and S. Seneff, "The SUMMIT Speech Recognition System: Phonological Modelling and Lexical Access," *Proc. ICASSP*, pp. 49-52, Albuquerque, NM, 1990.