SPECOM'2004: 9th Conference
Speech and Computer
St. Petersburg, Russia
September 20-22, 2004

Decision Tree Learning for Automatic Grapheme to Phoneme Conversion for Tamil

N.Udhyakumar[†], *C.S.Kumar*[*], *R.Srinivasan*[†] *and R.Swaminathan*[†]

Amrita Vishwa Vidyapeetham, Ettimadai, Coimbatore, Tamilnadu, INDIA – 641105.
[†]*{udhay_ece,vasan_it,swami_ece}@rediffmail.com,* [*]*cs_kumar@ettimadai.amrita.edu*

Abstract

This paper describes a novel approach for grapheme to phoneme conversion using decision tree learning technique. The proposed approach, unlike the rule based approach, can generate rules spanning wider context and thus give better accuracy for the conversion.

1. Introduction

Grapheme to phoneme conversion (g2p) is important in both text-to-speech and speech recognition systems. Although lexicons prepared by linguists are the most reliable resource, they are never exhaustive as they cannot cover the complete list of words in any language. A detailed lexicon also consumes a lot of storage space. This restricts the application of the speech technologies to dynamic vocabulary environments. Hence an alternative procedure to predict pronunciation from orthographic form is necessary.

Several approaches have been used for automatic grapheme to phoneme conversion. Inspired by the phonetic languages, grapheme-to-phoneme conversion is usually carried out by a set of rules [1,2]. They require lot of expertise to develop and sometimes hard to check for rule interference and redundancy. Statistical modeling has been successfully applied to the problem of g2p rules in many languages such as English, French and German in [3]. Use of neural networks in grapheme to phoneme conversion is explored in [4]. These approaches can eliminate the time consuming step of rule writing, so the training process can be done fully automatically. A decision tree based automatic g2p mapping is studied in this paper. The main objective of our work is to build an automatic grapheme to phoneme conversion using decision tree clustering for Tamil text-to-speech and speech recognition applications.

Tamil, like many other Indian languages, is a phonetic language with nearly a one to one mapping between the grapheme and the phoneme. However direct g2p mapping is not possible because when these characters appear in certain contexts produce different phones. For eg. ¸ in Tamil is pronounced as ***k*** in *கடல்* or ***g*** in *தங்கம்* or ***hh*** in *பகல்*. Morphological and phonetic analysis of Tamil have proved that these variations follow certain complex language regularities, which motivates the use of data-driven techniques like decision trees for the task [6]. The trees need to be trained on a large manually transcribed lexicon. For Tamil there is no such pronunciation dictionary available in the public domain. Hence we use a rule-based system to generate a lexicon and is then refined by hand checking.

The rest of the paper is organized as follows. Section 2 describes the details of Tamil orthography and phone set. Rule-based and decision tree approaches are explained in Section 3 and 4 respectively. Section 5 presents the results of automatic g2p conversion using decision trees and finally, section 6 concludes.

2. Tamil orthography and phone set

Tamil orthography consists of 12 vowels termed as Uyir (Soul) and 18 consonants as Mey (Body). The consonants combine with vowels individually to form a new set of 216 additional graphemes named Uyirmey. The grapheme ஃ has a unique name Ayutham (Weapon). Tamil sounds are classified into 43 phones. Table.1 gives the details of Tamil phone set.

Grapheme	Phone	Example
அ	ax	அம்மா
ஆ	aa	ஆடு
இ	ih	இலை
ஈ	iy	ஈகை
உ	uh	உணவு
	U	தப்பு
ஊ	uw	ஊர்
எ	ae	எட்டு
ஏ	ey	ஏணி
ஒ	ao	ஒன்று
ஓ	ow	ஓம்
ஔ	aw	ஔவை
ஷ்	sh	விஷம்
ஜ்	jh	ஜாதி

க்	k	கல்வி
	g	பங்கு
	hh	பகல்
ங்	ng	தங்கம்
ச்	ch	காட்சி
	s	சங்கு
ஞ்	ny	ஞானம்
ட்	t	அட்டை
	d	வண்டி
ண்	nn	கண்
த்	tx	தட்டு
	dh	பந்தம்
	T	மதி
ந்	nd	பந்து
	n	நான்
ப்	p	பாடம்
	b	அம்பு
	P	லாபம்
ம்	m	கரும்பு
ய்	y	யமுனை
ர்	r	மரம்
ல்	l	மாலை
வ்	v	வயது
ழ்	lzh	பழம்
ள்	ll	உள்ளம்
ற்	tr	ஒற்றன்
	dr	கன்று
	rr	அறம்
ன்	n	அன்னை

Table.1: Tamil graphemes and the phone set

3. Rule-based g2p conversion

It is evident from the examples in Table.1 that there is no trivial one-to-one mapping between the Tamil graphemes and its phonemes. However, in most cases, these variations can be decoded based on the contextual information. Hence powerful context sensitive rules are designed to accurately predict the pronunciation. The rules have specific conventions about lookup, context and the target phone. For e.g. the rule

nasal k (*) => g

means that **k** when preceded by a nasal (e.g. **ng** (ங்)) and followed by anything becomes the sound **g** as in பங்கு. As with any *expert system* it is difficult to anticipate all possible relevant cases. For example the grapheme **k** in the word கிராமம் (kiramam) has a **g** sound though this usually happens only if **k** is preceded by a nasal. Similarly the grapheme **p** has a **b** sound in the word பந்தம் (pandtham), which is also not captured by rule-based system. The exceptions need to be stored in a separate exception dictionary. There is generally a trade-off between size of the exception dictionary and the complexity of the rules. This proves that grapheme to phoneme mapping is highly complicated which makes this field attractive for machine learning algorithms like decision trees.

4. Decision tree based g2p conversion

Decision trees can be used to capture the phonetic variations of a phoneme based on the nature of the neighboring phones. This can be interpreted accordingly as neighboring characters in the orthography, and the system in turn can use the graphemes to capture these phonetic variations to generate the transcription.

Decision trees consist of nodes that contain the rules and leaves, which are labeled with target classes. At each node a decision criteria assigns the object to the left or right sub-tree. When the object reaches a leaf, the class label of this leaf is used as the answer for the specific transcription.

4.1. Alignment procedure

Before training, we must align each grapheme to its corresponding phones. Due to strong relation between graphemes and phones in Tamil, complex alignment procedures [7] are not required. Grapheme strings are always equal in length or longer than phone sequences in Tamil. To have one-to-one correspondence with phone sequences, phonemic epsilons (Nulls) are inserted at appropriate places. Certain words have alternate pronunciations where a single grapheme maps to more than one phone. For e.g the grapheme ச் in சென்னை can be pronounced as either ***/ch/*** or ***/s/***. We deal with this case by adding a new entry ***/ch*s/*** in the phone set. The set of allowed phones is also specified for each grapheme to align them with the corresponding phone.

4.2. Training of decision trees

The basic decision tree consists of a set of *yes-no* questions and a procedure to select the best question in a way to maximize the information gain at every node split to grow the tree from the root [8]. This information can be quantified as entropy, eqn. (1). The split continues until the increase in information – decrease in entropy, as in eqn. (2), falls below a threshold. The entropy E is defined as

$$E = -\sum_{i=1}^{n} f_i \log_2 f_i \qquad (1)$$

where *fi* is the relative frequency of occurrence for the i^{th} phoneme, and *N* is the number of phonemes.

For an attribute, the entropy after the split, $\mathbf{E}^{\mathbf{S}}$ is computed as the average of the entropies of the subsets. The information gain **G** for an attribute is given by

$$G = E - E^{s} \qquad (2)$$

4.3.Questions for decision trees

The basic yes-no question for G2P conversion looks like "Is the first left grapheme 'm'?".Questions for phones are not included as they are much similar to questions for graphemes in Tamil. The range of question positions must be enough to cover the long-distance phonological variations. It was found that a 5-grapheme window (2 for left context and 2 for right context) is generally sufficient. A primitive set would be the set of all the singleton questions (question about a single alphabet). One problem with allowing only simple questions is that the data may be over-fragmented, resulting in similar leaves in different locations of the tree. Splitting data across different nodes leads to redundant clusters, reduced trainability and less accurate entropy reduction [9].

We observed that if we allow the node to have a complex question with phonetically relevant questions like "Is the left grapheme a nasal?", the depth of the tree will be greatly reduced and the performance improved. Complex questions can also alleviate the data fragmentation problem caused by the greedy nature of the decision tree algorithm. vowels, consonants, nasals, stops, glides and retroflex are some of the phone clusters used for generating the composite questions.

4.4. Generation of phone sequences

After the decision trees are trained, they can be used for generating phone sequences for the words in the vocabulary. For each word, a phoneme sequence is generated by going through the word from left to right one grapheme at a time. Based on the contextual information the tree is traversed starting at the root node. The program traverses through the tree until a leaf is reached and the identity of the leaf is the transcription for the current grapheme. Then the process moves on to the next grapheme in a similar way. In case of pseudophonemes alternative pronunciations are produced to get the final transcriptions. For eg. சென்னை, */ch*s ae n h ax y/* is expanded as */ch ae n h ax y/* and */s ae n h ax y/*.

5. Experiments

We have used EMILLE corpus [10] for our experiments. This corpus is represented in Unicode and for ease of being handled by the software, we converted them to an ASCII like representation. We selected 85,000 words from the corpus and generated the lexicon to train the decision tree. The decision trees are implemented using Weka [11], a tool for experimenting with machine learning algorithms developed at the University of Waikato, New Zealand.

5.1. Rule based g2p system

The rule-based system is prepared to capture the grapheme-phoneme dependencies in terms of simple linguistic rules using just one right and one left context. Capturing rules beyond one grapheme span is very difficult in a rule-based system. The system is tested for its accuracy and is listed in Table.2

5.2.Decision tree based g2p

Decision trees to produce transcriptions are trained using a manually generated lexicon. The lexicon was initially bootstrapped using the rule-based system and refined manually by an expert linguist. First, we used singleton questions about the grapheme identity to build the decision trees. The system is tested and the accuracy is compared with the rule-based approach.

Further, to enhance the accuracy, we formed phonetically motivated cluster of phones like nasals, front vowels, back vowels, consonants, stops, etc. and used these as questions (complex questions) to train the trees. (Table.2) lists the performance results for this experiment.

g2p System	Word Level	Phone Level
Rule-Based	94%	95.5%
Decision Tree-Singleton Questions	96.5%	98%
Decision Tree-Complex Questions	98%	98.5%

Table.2. Simple Vs Composite questions for g2p

6. Conclusion and future work

We have presented a system using decision tree clustering for grapheme to phoneme conversion. The present approach out-performs our previous rule based system.

This approach is suitable for applications like mobile communication environment, where the vocabulary is dynamic and has limited memory to store huge pronunciation lexicons.

To improve the accuracy further, it may be worth including other knowledge sources like phone n-grams as a post-processor. Also, inclusion of lexical tags in the attribute list may also enhance the accuracy [12]. Work in this direction is being pursued.

7. Acknowledgements

The authors would like to thank Mr.Shunmugom, Department of linguistics, Bharathiar University for useful linguistic discussions during the course of this work. We gratefully acknowledge all the students of our Institute who helped us in preparing the lexicon. Also we would like to express our appreciation to Dr. V.P. Mohan Das, Chairperson, ECE Department for his active support and encouragement. We also thank Mata Amritanandha Mayi for her love and blessings.

8. References

[1] Elovitz, H.S., Johnson, R., McHugh, A. and Shore, J.A. "Letter-to-sound rules for automatic translation of English text to phonetics", *IEEE Transactions on Acoustics, Speech and Signal Processing*, 1976.

[2] C.S.Kumar et al. "An Automatic Grapheme to Phoneme Converter for Malayalam", Indian Linguistics", *Proc. All India conference of linguists.* 2002.

[3] Black, A., Lenzo, K. and Pagel, V. "Issues in Building General Letter to Sound Rules", *3rd ESCA Speech Synthesis Workshop*, pp. 77-80, Jenolan Caves, Australia, 1998.

[4] Sejnowski T.J and Rosenberg C.R. "Parallel network that learn to pronounce English text" *Complex Systems*, 1:145 -168, 1987.

[5] R.Duraipandi, "The Morphophonemic Rule for Tamil Computing", *Tamil Internet02,* California, USA, 2002.

[6] L.Jiang, H.W.Hon and X.D.Huang, "Improvements on a Trainable Letter to Sound Converter," *Proc. Eurospeech'97*, Rhodes, Greece 1997.

[7] S.Deligne, F.Yvon, and F.Bimbot, "Variable-length sequence matching for phonetic transcription using joint multigrams". *Proc. EUROSPEECH*, 1995.

[8] L.Breiman, J.H.Friedman, R.A.Olshen, and C.J. Stone. "Classification and regression trees", Wadsworth, Inc. , California, USA 1984.

[9] Xuedong Huang, Alex Acero and Hsaio-Weun Hon, "Spoken Language Processing", Prentice Hall, New Jersey, 2001.

[10] www.emille.lancs.ac.uk

[11] Ian.H.Witten, Eibe Frank, "WEKA-Machine Learning Algorithms in Java", University of Waikato, Newzealand.

[12] L.Galescu, and E.K.Ringger, "Augmenting Words With Linguistic Information For N-Gram Language Models" *Proc. EUROSPEECH*, 1999.