Modeling Letter-to-Phoneme Conversion as a Phrase Based Statistical Machine Translation Problem with Minimum Error Rate Training

Taraka Rama, Anil Kumar Singh, Sudheer Kolachina
Language Technologies Research Centre,
IIIT-Hyderabad, India.
{taraka@students,anil@research,sudheerkpg08@research}.iiit.ac.in

Abstract

Letter-to-phoneme conversion plays an important role in several applications. It can be a difficult task because the mapping from letters to phonemes can be many-to-many. We present a language independent letter-to-phoneme conversion approach which is based on the popular phrase based Statistical Machine Translation techniques. The results of our experiments clearly demonstrate that such techniques can be used effectively for letter-to-phoneme conversion. Our results show an overall improvement of 5.8% over the baseline and are comparable to the state of the art. We also propose a measure to estimate the difficulty level of L2P task for a language.

1 Introduction

Letter-to-phoneme (L2P) conversion can be defined as the task of predicting the pronunciation of a word given its orthographic form (Bartlett et al., 2008).The pronunciation is usually represented as a sequence of phonemes. Letter-to-phoneme conversion systems play a very important role in spell checkers (Toutanova and Moore, 2002), speech synthesis systems (Schroeter et al., 2002) and transliteration (Sherif and Kondrak, 2007). Letter-to-phoneme conversion systems may also be effectively used for cognate identification and transliteration. The existing cognate identification systems use the orthographic form of a word as the input. But we know that the correspondence between written and spoken forms of words can be quite irregular as is the case in English. Even in other languages with supposedly regular spellings, this irregularity exists owing to linguistic phenomena like borrowing and language variation. Letter-to-phoneme conversion systems can facilitate the task of cognate identification by providing a language independent transcription for any word.

Until a few years ago, letter-to-phoneme conversion was performed considering only one-one correspondences (Black et al., 1998; Damper et al., 2004). Recent work uses many-to-many correspondences (Jiampojamarn et al., 2007) and reports significantly better accuracy for Dutch, German and French. The current state of the art systems give as much as 90% (Jiampojamarn et al., 2008) accuracy for languages like Dutch, German and French. However, accuracy of this level is yet to be achieved for English.

Rule-based approaches to the problem of letter-to-phoneme conversion although appealing, are impractical as the number of rules for a particular language can be very high (Kominek and Black, 2006). Alternative approaches to this problem are based on machine learning and make use of resources such as pronunciation dictionaries. In this paper, we present one such machine learning based approach wherein we envisage this problem as a Statistical Machine Translation (SMT) problem.

The outline of the paper is as follows. Section 2 presents a brief summary of the related work done in L2P conversion. Section 3 describes our model and the techniques for optimizing the performance. Section 4 describes the letter-to-phoneme alignment. The description of the results and experiments and a new technique for estimating the difficulty level of

L2P task has been given in Section 5. Error analysis is presented in Section 6. Finally we conclude with a summary and suggest the directions for future work.

2 Related Work

In the letter-to-phoneme conversion task, a single letter can map to multiple phonemes [x $\rightarrow$ ks] and multiple letters can generate a single phoneme. A letter can also map to a null phoneme [e $\rightarrow \varphi$] and vice-versa. These examples give a glimpse of why the task is a complex task and a single machine learning technique may not be enough to solve the problem. A overview of the literature supports this claim.

In older approaches, the alignment between the letters and phonemes was taken to be one-to-one (Black et al., 1998) and the phoneme was predicted for every single letter. But the more recent work (Bisani and Ney, 2002; Jiampojamarn et al., 2007) shows that multiple letter-to-phoneme alignments perform better than single letter to phoneme alignments. The problem can be either viewed as a multi-class classifier problem or a structure prediction problem. In structure prediction, the algorithm takes the previous decisions as the features which influence the current decision.

In the classifier approach, only the letter and its context are taken as features. Then, either multiclass decision trees (Daelemans and van den Bosch, 1997) or instance based learning as in (van den Bosch and Daelemans, 1998) is used to predict the class, which in this case is a phoneme. Some of these methods (Black et al., 1998) are not completely automatic and need an initial handcrafted seeding to begin the classification.

Structure prediction is like a tagging problem where HMMs (Taylor, 2005) are used to model the problem. Taylor claims that except for a preprocessing step, it is completely automatic. The whole process is performed in a single step. The results are poor, as reasoned in (Jiampojamarn et al., 2008) due to the emission probabilities not being informed by the previous letter's emission probabilities. Pronunciation by Analogy (PbA) is a data-driven method (Marchand and Damper, 2000) for letter-to-phoneme conversion which is used again by Damper et al (2004). They simply use an Expectation-Maximisation (EM) like algorithm for aligning the letter-phoneme pairs in a speech dictionary. They claim that by integrating the alignments induced by the algorithm into the PbA system, they were able to improve the accuracy of the pronunciation significantly. We also use the many-to-many alignment approach but in a different way and obtained from a different source.

The recent work of Jiampojamarn et al (2007) combines both of the above approaches in a very interesting manner. It uses an EM like algorithm for aligning the letters and phonemes. The algorithm allows many-to-many alignments between letters and phonemes. Then there is a letter chunking module which uses instance-based training to train on the alignments which have been obtained in the previous step. This module is used to guess the possible letter chunks in every word. Then a local phoneme predictor is used to guess the phonemes for every letter in a word. The size of the letter chunk could be either one or two. Only one candidate for every word is allowed. The best phoneme sequence is obtained by using Viterbi search.

An online model MIRA (Crammer and Singer, 2003) which updates parameters is used for the L2P task by Jiampojamarn et al (2008). The authors unify the steps of letter segmentation, phoneme prediction and sequence modeling into a single module. The phoneme prediction and sequence modeling are considered as tagging problems and a Perceptron HMM (Collins, 2002) is used to model it. The letter segmenter module is replaced by a monotone phrasal decoder (Zens and Ney, 2004) to search for the possible substrings in a word and output the n-best list for updating MIRA. Bisani and Ney (2002) take the joint multigrams of graphemes and phonemes as features for alignment and language modeling for phonetic transcription probabilities. A hybrid approach similar to this is by (van den Bosch and Canisius, 2006).

In the next section we model the problem as a Statistical Machine Translation (SMT) task.

3 Modeling the Problem

Assume that given a word, represented as a sequence of letters $\mathbf{l} = l_1^J = l_1 ... l_j ... l_J$, needs to be transcribed as a sequence of phonemes, represented as **f**

= f_1^I = $f_1 ... f_i ... f_I$. The problem of finding the best phoneme sequence among the candidate translations can be represented as:

$$\mathbf{f}_{best} = \arg\max_{\mathbf{f}} \{\Pr(\mathbf{f} \mid \mathbf{l})\} \quad (1)$$

We model the problem of letter to phoneme conversion based on the noisy channel model. Reformulating the above equation using Bayes Rule:

$$\mathbf{f}_{best} = \arg\max_{\mathbf{f}} p(\mathbf{l} \mid \mathbf{f})\, p(\mathbf{f}) \quad (2)$$

This formulation allows for a phoneme n-gram model $p(\mathbf{f})$ and a transcription model $p(\mathbf{l} \mid \mathbf{f})$. Given a sequence of letters **l**, the argmax function is a search function to output the best phonemic sequence. During the decoding phase, the letter sequence **l** is segmented into a sequence of *K* letter segments $\bar{l}_1^K$. Each segment $\bar{l}_k$ in $\bar{l}_1^K$ is transcribed into a phoneme segment $\bar{f}_k$. Thus the best phoneme sequence is generated from left to right in the form of partial translations. By using an *n*-gram model p_{LM} as the language model, we have the equations:

$$\mathbf{f}_{best} = \arg\max_{\mathbf{f}} p(\mathbf{l} \mid \mathbf{f})\, p_{LM} \quad (3)$$

with $p(\mathbf{l} \mid \mathbf{f})$ written as

$$p(\bar{l}_1^K \mid \bar{f}_1^K) = \prod_{k=1}^{K} \Phi(\bar{l}_k \mid \bar{f}_k) \quad (4)$$

From the above equation, the best phoneme sequence is obtained based on the product of the probabilities of transcription model and the probabilities of a language model and their respective weights. The method for obtaining the transcription probabilities is described briefly in the next section. Determining the best weights is necessary for obtaining the right phoneme sequence. The estimation of the models' weights can be done in the following manner.

The posterior probability $\Pr(\mathbf{f} \mid \mathbf{l})$ can also be directly modeled using a log-linear model. In this model, we have a set of M feature functions $h_m(\mathbf{f}, \mathbf{l}), m = 1...M$. For each feature function there exists a weight or model parameter $\lambda_m, m = 1...M$. Thus the posterior probability becomes:

$$\Pr(\mathbf{f} \mid \mathbf{l}) = p_{\lambda_1^M}(\mathbf{f} \mid \mathbf{l}) \quad (5)$$

$$= \frac{\exp\left[\Sigma_{m=1}^{M} \lambda_m h_m(\mathbf{f}, \mathbf{l})\right]}{\sum_{\acute{f}_1^I} \exp\left[\Sigma_{m=1}^{M} \lambda_m h_m(\acute{f}_1^I, \mathbf{l})\right]} \quad (6)$$

with the denominator, a normalization factor that can be ignored in the maximization process.

The above modeling entails finding the suitable model parameters or weights which reflect the properties of our task. We adopt the criterion followed in (Och, 2003) for optimising the parameters of the model. The details of the solution and proof for the convergence are given in Och (2003). The models' weights, used for the L2P task, are obtained from this training.

4 Letter-to-Phoneme Alignment

We used GIZA++ (Och and Ney, 2003), an open source toolkit, for aligning the letters with the phonemes in the training data sets. In the context of SMT, say English-Spanish, the parallel corpus is aligned bidirectionally to obtain the two alignments. The IBM models give only one-to-one alignments between words in a sentence pair. So, GIZA++ uses some heuristics to refine the alignments (Och and Ney, 2003).

In our input data, the source side consists of grapheme (or letter) sequences and the target side consists of phoneme sequences. Every letter or grapheme is treated as a single 'word' for the GIZA++ input. The transcription probabilities can then be easily learnt from the alignments induced by GIZA++, using a scoring function (Koehn et al., 2003). Figure 1 shows the alignments induced by GIZA++ for the example words which are mentioned by Jiampojamarn et al (2007). In this figure, we only show the alignments from graphemes to phonemes.

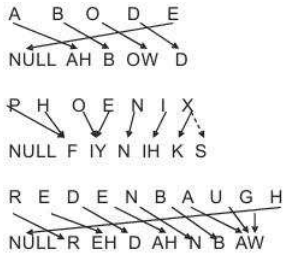

Figure 1: Example Alignments from GIZA++

5 Evaluation

We have evaluated our models on the English CMUDict, French Brulex, German Celex and Dutch Celex speech dictionaries. These dictionaries are available for download on the website of PROANALSYL[1] Letter-to-Phoneme Conversion Challenge. Table 1 shows the number of words for each language. The datasets available at the website were divided into 10 folds. In the process of preparing the datasets we took one set for test, another for developing our parameters and the remaining 8 sets for training. We report our results in word accuracy rate, based on 10-fold cross validation, with mean and standard deviation.

Language	Datasets	Number of Words
English	CMUDict	112241
French	Brulex	27473
German	Celex	49421
Dutch	Celex	116252

Table 1: Number of words in each Dataset

We removed the one-to-one alignments from the corpora and induced our own alignments using GIZA++. We used minimum error rate training (Och, 2003) and the A* beam search decoder implemented by Koehn (Koehn et al., 2003). All the above tools are available as parts of the MOSES (Koehn et al., 2007) toolkit.

5.1 Exploring the Parameters

The parameters which have a major influence on the performance of a phrase-based SMT model are the alignment heuristics, the maximum phrase length (MPR) and the order of the language model (Koehn et al., 2003). In the context of letter to phoneme conversion, *phrase* means a sequence of letters or phonemes mapped to each other with some probability (i.e., the *hypothesis*) and stored in a phrase table. The *maximum phrase length* corresponds to the maximum number of letters or phonemes that a hypothesis can contain. Higher phrase length corresponds a larger phrase table during decoding.

We have conducted experiments to see which combination gives the best output. We initially trained the model with various parameters on the training data and tested for various values of the above parameters. We varied the maximum phrase length from 2 to 7. The language model was trained using SRILM toolkit (Stolcke, 2002). We varied the order of language model from 2 to 8. We also traversed the alignment heuristics spectrum, from the parsimonious *intersect* at one end of the spectrum through *grow, grow-diag, grow-diag-final, grow-diag-final-and* and *srctotgt* to the most lenient *union* at the other end. Our intuitive guess was that the best alignment heuristic would be *union*.

We observed that the best results were obtained when the language model was trained on 6-gram and the alignment heuristic was *union*. No significant improvement was observed in the results when the value of MPR was greater than 5. We have taken care such that the alignments are always monotonic. Note that the average length of the phoneme sequence was also 6. We adopted the above parameter settings for performing training on the input data.

5.2 System Comparison

We adopt the results given in (2007) as our baseline. We also compare our results with some other recent techniques mentioned in the Related Work section. Table 2 shows the results. As this table shows, our approach yields the best results in the case of German and Dutch. The word accuracy obtained for the German Celex and Dutch Celex dataset using our approach is higher than that of all the previous approaches listed in the table. In the case of English and French, although the baseline is achieved through our approach, the word accuracy falls short of being the best. However, it must also be noted that the dataset that we used for English is slightly larger than those of the other systems shown in the table.

We also observe that for an average phoneme accuracy of 91.4%, the average word accuracy is 63.81%, which roughly corroborates the claim of Black et al (Black et al., 1998) that a 90% phoneme accuracy corresponds to 60% word accuracy.

5.3 Difficulty Level and Accuracy

We also propose a new language-independent measure that we call 'Weighted Symmetric Cross Entropy' (WSCE) to estimate the difficulty level of the L2P task for a particular language. The *weighted*

[1] http://www.pascal-network.org/Challenges/PRONALSYL/

Language	Dataset	Baseline	CART	1-1 Align	1-1 + CSIF	1-1 + HMM	M-M Align	M-M + HMM	MeR + A*
English	CMUDict	58.3±0.49	57.8	60.3±0.53	62.9±0.45	62.1±0.53	65.1±0.60	65.6±0.72	63.81±0.47
German	Celex	86.0±0.40	89.38	86.6±0.54	87.6±0.47	87.6±0.59	89.3±0.53	89.8±0.59	90.20±0.25
French	Brulex	86.3±0.67	-	87.0±0.38	86.5±0.68	88.2±0.39	90.6±0.57	90.9±0.45	86.71±0.52
Dutch	Celex	84.3± 0.34	-	86.6±0.36	87.5±0.32	87.6±0.34	91.1±0.27	91.4±0.24	91.63±0.24

Table 2: System Comparison in terms of word accuracies. **Baseline:**Results from PRONALSYS website. **CART:** CART Decision Tree System (Black et al., 1998). **1-1 Align, M-M align, HMM:** one-one alignments, many-many alignments, HMM with local prediction (Jiampojamarn et al., 2007). **CSIF:**Constraint Satisfaction Inference(CSIF) of(van den Bosch and Canisius, 2006). **MeR+A*:**Our approach with minimum error rate training and A* search decoder. "-" refers to no reported results.

SCE is defined as follows:

$$d_{sce_{wt}} = \sum r_t \left(p_l \, log \left(q_f\right) + q_f \, log \left(p_l\right)\right) \qquad (7)$$

where p and q are the probabilities of occurrence of letter (l) and phoneme (f) sequences, respectively. Also, r_t corresponds to the conditional probability $p(f \mid l)$. This transcription probability can be obtained from the phrase tables generated during training. The weighted entropy measure $d_{sce_{wt}}$,for each language, was normalised with the total number of such n-gram pairs being considered for comparison with other languages. We have fixed the maximum order of l and f n-grams to be 6. Table 3 shows the difficulty levels as calculated using WSCE along with the accuracy for the languages that we tested on. As is evident from this table, there is a rough correlation between the difficulty level and the accuracy obtained, which also seems intuitively valid, given the nature of these languages and their orthographies.

Language	Datasets	$d_{sce_{wt}}$	Accuracy
English	CMUDict	0.30	63.81±0.47
French	Brulex	0.41	86.71±0.52
Dutch	Celex	0.45	91.63±0.24
German	Celex	0.49	90.20±0.25

Table 3: $d_{sce_{wt}}$ values predict the accuracy rates.

6 Error Analysis

In this section we present a summary of the error analysis for the output generated. We tried to observe if there exists any pattern in the words that were transcribed incorrectly.

The majority of errors occurred in the case of vowel transcription, and diphthong transcription in particular. In the case of English, this can be attributed to the phenomenon of lexical borrowing from a variety of sources as a result of which the number of sparse alignments is very high. The system is also unable to learn allophonic variation of certain kinds of consonantal phonemes, most notably fricatives like /s/ and /z/. This problem is exacerbated by the irregularity that might exist in the language in allophonic variation itself.

7 Conclusion and Future Work

In this paper we have tried to address the problem of letter-to-phoneme conversion by modeling it as an SMT problem and we have used minimum error rate training to obtain the suitable model parameters, which is according to our knowledge, a novel approach for L2P task. The results obtained are comparable to the state of the art system and our error analysis shows that a lot of improvement is still possible.

Intuitively, the performance of the system can be improved in at least two aspects. The first is the Minimum Error Rate Training (MERT) and the second is the decoding phase. Using phonetic feature based edit distance or string similarity as the loss function in the MERT implementation can improve results significantly. In addition, incorporating more model parameters and extensive testing of these parameters might improve the results of the system. We also plan to introduce a decoding scheme similar to the substring based transducer (Sherif and Kondrak, 2007) to improve the usage of lower order language models.

Acknowledgements

This work was supported by ILMT grant 11(10)/2006-HCC(TDIL).

References

Susan Bartlett, Grzegorz Kondrak, and Colin Cherry. 2008. Automatic syllabification with structured SVMs for letter-to-phoneme conversion. In *Proceedings of ACL-08: HLT*, pages 568–576, Columbus, Ohio, June. ACL.

Max Bisani and Hermann Ney. 2002. Investigations on joint-multigram models for grapheme-to-phoneme conversion. In *International Conference on Spoken Language Processing*, pages 105–108, Denver, CO, USA, September.

A.W. Black, K. Lenzo, and V. Pagel. 1998. Issues in Building General Letter to Sound Rules. In *The Third ESCA/COCOSDA Workshop (ETRW) on Speech Synthesis*. ISCA.

M. Collins. 2002. Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on EMNLP-Volume 10*, pages 1–8. ACL, Morristown, NJ, USA.

K. Crammer and Y. Singer. 2003. Ultraconservative online algorithms for multiclass problems. *The Journal of Machine Learning Research*, 3:951–991.

Walter M. P. Daelemans and Antal P. J. van den Bosch. 1997. Language-Independent Data-0riented Grapheme-to-Phoneme Conversion. *Progress in Speech Synthesis*.

R.I. Damper, Y. Marchand, J.D. Marseters, and A. Bazin. 2004. Aligning Letters and Phonemes for Speech Synthesis. In *Fifth ISCA Workshop on Speech Synthesis*. ISCA.

Sittichai Jiampojamarn, Grzegorz Kondrak, and Tarek Sherif. 2007. Applying many-to-many alignments and hidden markov models to letter-to-phoneme conversion. In *HLT 2007: The Conference of the NAACL; Proceedings of the Main Conference*, pages 372–379, Rochester, New York, April. ACL.

Sittichai Jiampojamarn, Colin Cherry, and Grzegorz Kondrak. 2008. Joint processing and discriminative training for letter-to-phoneme conversion. In *Proceedings of ACL-08: HLT*, pages 905–913, Columbus, Ohio, June. ACL.

P. Koehn, F.J. Och, and D. Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the NAACL:HLT-Volume 1*, pages 48–54. ACL Morristown, NJ, USA.

P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, et al. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *ACL*, volume 45, page 2.

J. Kominek and A.W. Black. 2006. Learning pronunciation dictionaries: language complexity and word selection strategies. In *HLT-NAACL*, pages 232–239. ACL, Morristown, NJ, USA.

Y. Marchand and R.I. Damper. 2000. A Multistrategy Approach to Improving Pronunciation by Analogy. *Computational Linguistics*, 26(2):195–219.

F.J. Och and H. Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51.

F.J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on ACL-Volume 1*, pages 160–167. ACL, Morristown, NJ, USA.

J. Schroeter, A. Conkie, A. Syrdal, M. Beutnagel, M. Jilka, V. Strom, Y.J. Kim, H.G. Kang, and D. Kapilow. 2002. A Perspective on the Next Challenges for TTS Research. In *IEEE 2002 Workshop on Speech Synthesis*.

Tarek Sherif and Grzegorz Kondrak. 2007. Substring-based transliteration. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 944–951, Prague, Czech Republic, June. Association for Computational Linguistics.

A. Stolcke. 2002. Srilm – an extensible language modeling toolkit.

P. Taylor. 2005. Hidden Markov Models for Grapheme to Phoneme Conversion. In *Ninth European Conference on Speech Communication and Technology*. ISCA.

K. Toutanova and R.C. Moore. 2002. Pronunciation modeling for improved spelling correction. In *Proceedings of the 40th annual meeting of ACL*, pages 144–151.

A. van den Bosch and S. Canisius. 2006. Improved morpho-phonological sequence processing with constraint satisfaction inference. In *Proceedings of the Eighth Meeting of the ACL-SIGPHON at HLT-NAACL*, pages 41–49.

A. van den Bosch and W. Daelemans. 1998. Do not forget: Full memory in memory-based learning of word pronunciation. *proceedings of NeMLap3/CoNLL98*, pages 195–204.

R. Zens and H. Ney. 2004. Improvements in phrase-based statistical machine translation. In *HLT Conf. / NAACL*, pages 257–264, Boston, MA, May.