Learning from Errors in Grapheme-to-Phoneme Conversion

Tatyana Polyakova, Antonio Bonafonte

Technical University of Catalonia
Jordi Girona 1-3, Barcelona, Spain
tatyana@gps.tsc.upc.es, antonio.bonafonte@upc.edu

Abstract

In speech technology it is very important to have a system capable of accurately performing grapheme-to-phoneme (G2P) conversion, which is not an easy task especially if talking about languages like English where there is no obvious letter-phone correspondence. Manual rules so widely used before are now leaving the way open for the machine learning techniques and language independent tools.
In this paper we present an extension of the use of transformation-based error-driven algorithm to G2P task. A set of explicit rules was inferred to correct the pronunciation for U.S. English, Spanish and Catalan using well-known machine-learning techniques in combination with transformation based algorithm. All methods applied in combination with transformation rules significantly outperform the results obtained by these methods alone.
Index Terms: grapheme-to-phoneme, speech synthesis, learning from errors

1. Introduction

The G2P conversion forms a very important part of the text-processing module of text-to-speech synthesis systems, where any errors are highly undesirable because a bad start decreases the overall performance of any system. In order to get good quality speech at the output, one cannot begin to synthesize with a high percentage of erroneous transcriptions. In TTS the presence of the errors in G2P conversion module is much more critical than in ASR.
For languages where one letter represents more than one phoneme, which are said to have deep orthography, the problem of G2P conversion still remains unsolved.
Since the language growth is a non-stop process and new words are entering from other languages every day, it is impossible to have a manually transcribed lexicon including all the words.
Facing the age when the methods of automatic data acquisition are the most economic in terms of time and effort, the manual dictionary updating is being substituted by the automatic ML methods.
A review of automatic G2P techniques can be found in [4].
The ML methods applied to the problem up to now leave the way for some further improvements. In English errors mostly appear when assigning pronunciation to vowels, whereas consonants are usually predicted better.
To improve the performance of the G2P transcription system, in this paper we propose to use an approach based on learning from errors committed by another conversion system used previously.

The transformation-based error-driven learning algorithm invented by Brill [3] was successfully applied to such NLP tasks as part-of-speech tagging, word sense disambiguation, phrase chunking etc. The high level of accuracy achieved for these tasks proves the effectiveness of this data-driven method.
Most of the experiments were performed on two lexicons of U.S. English.
For U.S. English four baseline conversion methods were used to obtain the necessary initial prediction and to train the error-driven system. These methods are: CART [1], FST [5], HMM [7] and such a naive prediction as the most-likely phone.
For other languages the initial prediction was done by CART.

2. Conversion Approaches

Many of the used methods required letters and phonemes to be aligned in a one-to-way, the alignment was done like in [1].

2.1. Decision trees (DT)

A decision tree [1] has as the input grapheme sliding window with three letters to the left and three to the right accordingly. This method is appropriate for discreet characteristics and produces rather compact models, whose size is defined by the total number of questions and leaf nodes in the output tree.
Usage only of grapheme context both on left and the right side by DT has a disadvantage: it assumes that the decisions are independent one from another so is that it cannot use the prediction of the previous phone as the reference to predict the next one. Another limitation introduced by the binary decision trees that every time a question is asked the training corpus is divided into two parts and further questions are asked only over the remaining parts of the corpus.

2.2. Finite State Transducers (FST)

This approach chooses the pronunciation φ that maximizes the probability of a phoneme sequence given the letter sequence g.

$$\hat{\varphi} = \arg \max_{\varphi} \{p(\varphi / g)\} \quad (1)$$

In this paper a finite state transducer similar to that of Galescu and Allen, [5] has been used.
To estimate the probability (1) is the same as to estimate the probability of the grapheme-phoneme pair, given a letter sequence. This estimation can be done using standard n-gram methods. Grapheme-phoneme pairs are extracted from the aligned dictionary.
N-grams can be represented by a finite-state automaton, where a new state is defined for each history h and a arc is created for each new grapheme-phoneme pair. These arcs are

labeled with a grapheme-phoneme pair and weighted with its probability given the history h. To derive the finite state transducer the labels attached to the automaton edges are split in a way that letters become input and phonemes become output.

The best pronunciation is equivalent to the best path through the FST maximizing the probabilities, given g; this is done by means of dynamic programming.

To allow maximum flexibility, the x-gram was used [2]. The x-gram is an n-gram with flexible length. In this model the length of the conditioning history depends on each particular history. Choosing x-gram's parameters carefully the number of states can be significantly reduced without any decrease in model's performance.

2.3. Hidden Markov Models (HMM)

It was recently proposed by Paul Taylor [10] to use HMM to confront the difficult problem of phoneme prediction.

In this case each phoneme is represented by one HMM and letters are the emitted observations.

The probability of transitions between models is equal to the probability of the phoneme given the previous phoneme. The objective of this method is to find the most probable sequence of hidden models (phonemes) given the observations (letters), using the probability distributions found during the model training

$$\hat{\varphi} = \arg\max_{\varphi} p(g,\varphi)\,p(\varphi) \qquad (2)$$

One model is trained for each phoneme; the maximum number of letters that a phoneme is able to generate was taken to be four, since it is uncommon that more than four letters represent a single sound, at least in English. No looping states were allowed unlike in the model configuration that serves for speech recognition.

For this method the G2P alignment of the dictionary is not necessary as it is done during the training stage by Baum-Welch training in which the HMM uses the probabilities of the G2P correspondence found in the previous step of the algorithm.

Our automatic speech recognition toolkit was used to train the HMM models and to decode graphemes into phonemes; the models were enhanced by phoneme x-gram of maximum length of 5 to benefit from the information about the phoneme context which was proved to be necessary.

3. Advantages of Transformation-Based Error-Driven Learning Approach (TBL)

The transformation-based error-driven algorithm (TBL) originally invented by Eric Brill [3] consists in learning the transformation rules from the training data that is labeled with some initial classes.

The main difference between manually derived set of rules and the set of rules extracted by TBL is that the second set doesn't need to be elaborated by experts. The method is fully automatic apart from the rule template creation step.

The order of rule application also does not require any knowledge about the language. It is established automatically during the training of the system. The rules that have the highest best score are put at the top of the list and then the other ones with a lower score are added.

The rules are language independent and could be applied to any supervised prediction task in combination with any machine learning technique, while the manually elaborated rules are non-transferable to other languages, which is an added disadvantage to their very high development cost.

As CART or FST, this method requires the data to be aligned in a one-to-one manner.

During the training process the algorithm's main goal is to capture certain regularity between the errors in the first prediction and to choose best transformation rule according to the environment where the error was committed.

The learning process is similar to when a human is trying to learn a language, a human learns from errors by memorizing certain conditions under which the error was committed, in the future trying to avoid the occurrence of the same error, given alike circumstances. If compared to foreign language learning many examples of similar situations can be found which could be incorrect word order in the question, erroneously memorized noun gender (for languages where it is present), stress misplacement, etc.

This learning mechanism is activated every time the error is committed. The first time one does something it is a normal thing to make a mistake, but after knowing the right way to do it, it is unlikely to repeat the same or similar mistake given the same conditions. The TBL algorithm works in the same way: it generates rules that try in the best way to generalize the transcription errors obtained by the initial prediction method. Once the patterns transformation_condition→correct_answer are captured, the TBL applies these patterns to correct the errors. The error itself usually forms part of the transformation condition as well as the conditions of its occurrence.

3.1 Applying the transformation-based learning to grapheme-to-phoneme conversion task.

Applying the transformation-based learning to G2P task seems to be of great advantage because it is completely language independent, efficient and can be understood easily.

Using the TBL algorithm to correct the prediction previously obtained by another classifier allows us to capture the imperfections of previous approximations to the linguistic irregularities into a set of context-dependent transformation rules, where the context serves as the conditioning features.

If the input set of features coincides with the one defined in the rule, then the rule is applied to this set of features and then its score is calculated depending on the number of the errors the rule has been able to correct in this step. The best rule is therefore chosen.

The rules are generated at each step, first each one of them is applied to the corpus initialized with some prediction, then the best rule found in the first step is applied to the training corpus and the second prediction is obtained and so on. In this optimization task the objective function is error rate. The process continues until no rule that improves the accuracy could be found or a rule with a score lower than the preset threshold has been generated.

Another advantage of the TBL algorithm is that it uses intermediate results to generate new prediction transformation, making those more reliable.

For example if the transformation rule is the following : *if* fon_-2= null, fon_-1=n, let_-2=k, let_-1=n, let_0=I, fon_0=I, change

fon_0 = aI, if the fon_-2 was erroneous, this rule would not apply, buy having corrected previously that phone two algorithms steps ago, by applying the rule : *if* let_-1=nothing, let_0=k, let_1=n, change fon_0= null, we obtain a better correction based on corrected pronunciation.

Rules are the main factor influencing on the overall performance. Rule templates define what set of rules is to be generated and applied to the corpus in search of the one that best corrects the erroneous prediction,

An example of rule templates is given in Figure 1.

```
let_-1 let_0 let_1 => fon
let_-1 let_0 let_-1 fon_0 => fon
let_-1 let_0 let_1 fon_-1 fon_0 fon_1 => fon
```

Figure 1 *Example of a rule template*.

In rule templates most of all the possible combinations of letter and phone contexts were considered, limiting the largest letter context to be plus/minus 5 letters and largest phone context to be plus/minus 3 phones, correspondingly.

To obtain the results the fnTBL toolkit, kindly provided for public use by its authors [6] was used. The fnTBL differs from the original Brill's TBL in the way that the objective function is calculated and that it reaches a speed up, without reducing the system's performance.

The Figure 2 shows a scheme of algorithm combination.

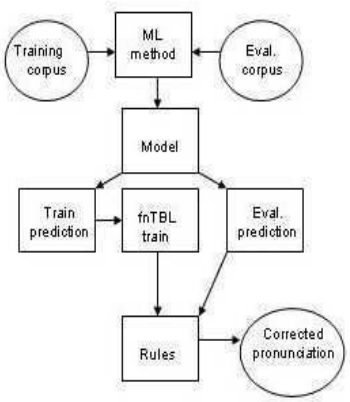

Figure 2 *Scheme of combination of ML learning techniques with fnTBL algorithm*

4. Experimental results

Most of the experiments were conducted using two lexicons of U.S. English: the LC-STAR dictionary covering about 50K words, produced by NSC, and the Unisyn lexicon publicly available from www.cstr.ed.ac.uk.; it has about 110K words. To test the tools on other languages the pronunciation was inferred for Catalan and Spanish LC-STAR lexicons (about 55K each). All 4 lexicons used included only common names.

The training and test data set are 90 and 10 percent accordingly.

To evaluate the word accuracy and the phoneme accuracy were calculated. The word accuracy is the most important as it shows the real potential of a transcription system to grasp the language behavior.

4.1. G2P results for U.S. English

First, the baseline results for both lexicons were obtained, then the fnTBL algorithm was applied to the output of all the machine-learning methods to correct the pronunciation predicted by them, following the scheme shown in the Figure 2. Also, such a naïve prediction as assigning the most-likely phone seen in the training to each grapheme accordingly, was considered.

In Tables 1 through 2 the G2P results for LC- STAR dictionary of U.S. English are given.

Table 1 *Baseline G2P results and those improved by combining of 4 transcription methods with fnTBL for the LC-STAR lexicon (phoneme accuracy).*

	baseline	cont=3	cont=4	cont=5
ML	59.70%	95.17%	95.59%	95.76%
DT	93.93%	94.64%	94.85%	95.00%
FST	93.63%	95.73%	95.87%	95.92%
HMM	84.16%	92.66%	93.15%	93.29%

Table 2 *Baseline and improved G2P results for the LC-STAR lexicon (word accuracy).*

	baseline	cont=3	cont=4	cont=5
ML	1.07%	75.67%	77.46%	78.26%
DT	68.32%	73.06%	74.13%	74.68%
FST	75.66%	78.79%	79.33%	79.63%
HMM	47.54%	67.01%	68.70%	69.08%

Tables 3 through 4 represent G2P results obtained for the Unisyn lexicon of U.S. English.

Table 3 *Baseline and improved G2P results for the Unisyn lexicon (phoneme accuracy).*

	baseline	cont=3	cont=4	cont=5
ML	56.36%	96.68%	97.01%	97.12%
DT	95.26%	96.44%	96.60%	96.67%
FST	97.30%	97.46%	97.47%	97.49%
HMM	86.93%	94.38%	94.75%	94.92%

Table 4 *Baseline and improved G2P results for the Unisyn lexicon (word accuracy).*

	baseline	cont=3	cont=4	cont=5
ML	1.42%	82.39%	84.00%	84.61%
DT	72.67%	80.74%	81.63%	82.08%
FST	86.65%	87.25%	87.28%	87.36%
HMM	54.87%	74.19%	74.95%	75.79%

FSTs give much higher word accuracy than HMM or DT for both lexicons. The method giving the poorest results is HMM. To improve these results a kind of preprocessing might be needed like proposed in [7].
Applying a number of rewrite rules to the dictionary, using context-sensitive models together with introducing of stress patterns, in the case of a stress lexicon, would allow obtain better results with HMM.
Tables 1 through 4 show the percentage of the correct phonemes and words as a result of combination of four classifiers with learning from errors algorithm next to the baseline results.

The goal was to learn rules that would be able to correct the prediction obtained previously. The rules were learned for 3 different sizes of letter context: the maximum letter context included in the rules varied from 3 to5 letters to the left and/or right.

Applying the error-correcting rules to the output of various algorithms shows a significant improvement of those results.

The biggest improvement was achieved for the methods whose performance at the start was the poorest; it is due to the fact that the abundance of errors gave a way to their better generalization and capturing into the transformation rules. The decision tree word accuracy results were improved by a measure of 8-10 %, the HMM results were improved by about 20 %. The hugest improvement was made for the most likely phone prediction, where the preliminary prediction scored about 50 % phonemes and 1% words correct. Among the correctly predicted phonemes there were mostly consonants. The improvement ranged from 75 to 80 %. The FST prediction improvement range was equal to 1-5 %. As the FST algorithm performs very well on the training corpus, it leaves a small number of errors to learn from that's why the improvement is smaller.
The improvement was the less context-sensitive the better was the baseline prediction. The largest context gave the best results although it was more expensive in terms of the computation time. The time needed for computation also depended on the number of baseline errors. If there were few errors to correct in the training corpus the algorithm converged faster.

4.2. G2P results for other languages

For Spanish and Catalan the pronunciation was inferred by applying DT and then the transformations rules were learned with the aim to correct the errors of the first classifier.
The results are given in Table 5. The maximum context used was to +-5 letters and +- 3 phones. For Spanish the results obtained with DT are very good due to its shallow orthography, for Catalan we obtained a 5 % improvement after applying the correction rules.

Table 5 *Baseline and word accuracy improved by fnTBL for Spanish and Catalan LC-STAR lexicons of common words.*

	baseline	+fnTBL
Spanish	98.49%	98.91%
Catalan	83.72%	88.79%

5. Conclusions

The goal to obtain better G2P conversion results was set and achieved by means of applying a set of transformations learned from errors. The transformation rules were learned automatically from a training corpus previously labeled using four classifiers. The rule templates are language independent and can be used to generate transformation rules for any language.

The combination of all methods with transformation-based error-driven algorithms significantly improved the results obtained by these methods alone. The best G2P results were obtained by combining FST with TBL algorithm.

Correcting the errors in the case where the most-likely phone was assigned to the letter also gave competitive results proving the effectiveness of the transformations rules. In fact the results were higher than those obtained by the widely used decision trees and by newly proposed HMM for both dictionaries of American English.
For Spanish the obtained results were high as expected since for languages with shallow orthography the pronunciation of common names can be as easily inferred by a small set of simple rules as by ML techniques. In the case of proper names and neologisms the simple rules have difficulty to predict the pronunciation. A preliminary study shows that the word accuracy obtained with our rule-based system for proper names scores only 60.90%. In the future we plan to extend the use of ML techniques in combination with TBL algorithm to the prediction of pronunciation of proper names.

Also in the future the algorithms will be applied to predict the stress and the influence of such information as part-of-speech tags on the conversion results will be studied.

6. Acknowledgements

This work was sponsored by the European Community in the frameworks of TC-STAR project (IST-2002-FP6-506738, http://www.tc-star.org).

7. References

Black A., K. Lenzo K. and V. Pagel, Issues in building general letter to sound rules. In Proc. of the 3rd ESCA workshop on speech synthesis., Jenolah Caves, Australia, pp. 77-80, 1998

Bonafonte A. and J. B. Mariño, Language modeling using X-grams. In Proc. of ICSLP-96, Vol.1, Philadelphia, pp. 394-397, 1996

Brill E., Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging .Computational linguistics 21(4), pp. 543-565, 1995

Damper R. I., Marchand Y., Adamson M. J. and Gustafson K., A comparison of letter-to-sound conversion techniques for English text-to-speech synthesis. In Proc. of the Institute of Acoustics 20(6), 1998

Galescu L., J. Allen, Bi-directional Conversion Between Graphemes and Phonemes Using a Joint N-gram Model, In Proc. of the 4th ISCA Tutorial and Research Workshop on Speech Synthesis, Perthshire, Scotland, 2001

Florian R., Ngai G., Transformation-based learning toolkit, John Hopkins University, 2001

Taylor P., Hidden Markov Models for grapheme-to-phoneme conversion, In Proc. of Interspeech 2005, Lisbon, Portugal, pp. 1973-1976, 2005