Development of a phoneme-to-phoneme (p2p) converter to improve the grapheme-to-phoneme (g2p) conversion of names

Qian Yang[(a)] [1], Jean-Pierre Martens[(a)], Nanneke Konings[(b)], Henk van den Heuvel[(b)]

[(a)] Ghent University, Sint-Pietersnieuwstraat 41, B-9000 Ghent, Belgium
martens@elis.ugent.be

[(b)] Centre for Language and Speech Technology, Radboud University, Nijmegen, the Netherlands

Abstract

It is acknowledged that a good phonemic transcription of proper names is imperative for the success of many modern speech-based services such as directory assistance, car navigation, etc. It is also known that state-of-the-art general-purpose grapheme-to-phoneme (g2p) converters perform rather poorly on many name categories. This paper proposes to use a g2p-p2p tandem comprising a state-of-the-art general-purpose g2p converter that produces an initial transcription and a name category specific phoneme-to-phoneme (p2p) converter that aims at correcting the mistakes made by the g2p converter. The main body of the paper describes a novel methodology for the automatic construction of the p2p converter. The methodology is implemented in a software toolbox that will be made publicly available in a form that will permit the user to design a p2p converter for an arbitrary name category. To give a proof of concept, the toolbox was used for the development of three p2p converters for first names, surnames and geographical names respectively. The obtained systems are small (few rules) and effective: significant improvements (up to 50% relative) of the grapheme-to-phoneme conversion are obtained. These encouraging results call for a further development and improvement of the approach.

1. Introduction

Speech synthesis and recognition technologies can enable the development of very natural man-machine interfaces for applications such as car navigation, reverse directory assistance, stock management, etc. However, for being effective both technologies need good pronunciations (= phonemic transcriptions) of the many proper names (person names, place names, brand names, etc.) that occur in these kind of applications.

It is acknowledged though (Yvon et al, 1998; Quazza, van den Heuvel, 2000; Boula de Mareüil, 2005;…) that general-purpose grapheme-to-phoneme (g2p) converters usually produce a significant number of mistakes when converting proper names. This is not so surprising since proper names can have archaic spellings, they are often of a foreign origin and therefore not pronounced according to the normal g2p rules of the target language, etc. Consequently, one needs a dedicated g2p converter that models the peculiarities of the pronunciations of the envisaged name category. One way of accomplishing this is by developing such a converter using self-learning techniques. All it takes is good machine learning software and a sufficiently large pronunciation dictionary of names and their correct transcription(s). However, it has been shown (Bouma, 2000; Anderson, 1996) that the quality of such a g2p converter very much depends on the size and the quality of the available pronunciation dictionary. In (Bouma, 2000) for instance the word error rate augments by almost 40% if the size of the training dictionary is reduced from 40 to 20K words.

In this paper we propose an alternative, two-step approach. In the first step we let a full fletched general-purpose g2p converter produce an initial transcription. In the second step, we let a so-called phoneme-to-phoneme (p2p) converter correct this initial transcription. This is an attractive option because it permits the p2p converter to profit from the knowledge of the general-purpose g2p converter (e.g. its morphological knowledge which gave rise to the syllabification and accentuation). Furthermore, since the p2p converter only has to focus on pronunciation rules which are typical for the envisaged name category, we anticipate that it will be compact (few rules), and that it will not take a very large name pronunciation dictionary to attain a good performance.

The idea of using a two-step approach for g2p conversion is not entirely new: Bouma (2000) developed a g2p converter as a tandem of a simple system based on handcrafted rules and a more elaborate finite state transducer that was designed in a data-driven way to improve the initial transcriptions. Main differences between our work and that of Bouma are (1) that we depart from a more powerful state-of-the-art g2p converter, (2) that we consider syllabification and stress assignment as integral parts of the g2p conversion, (3) that we adopt other rule formalisms and rule learning approaches, and (4) that we presume to have no access to the exact grapheme-phoneme transformations performed by the initial stage.

The remaining part of the paper is organized as follows. In Section 2 we provide some definitions and conventions that we need further on. In Sections 3 and 4 we outline the proposed g2p-p2p tandem approach, and the different steps involved in the automatic p2p learning process. In Section 5 we describe how the p2p converter is integrated in the transcription process. In Section 6 we

[1] Has recently moved to Nuance

discuss how the software tools were used with success for the creation of p2p converters for three name categories: first names, surnames and geographical names. The paper ends with a discussion of the obtained results and the directions for future research.

2. Some definitions and conventions

Although we presume that the reader knows very well what graphemes and phonemes are, we think it is useful at this point to make explicit what we mean by a phonemic and orthographic transcription in the present work.

A phonemic transcription is a text string that can be decomposed into phonemic units, and a phonemic unit can be a phoneme, a boundary mark or a stress mark. Boundary marks represent intra-word and inter-word syllable boundaries (a name may consist of several constituent words). Stress marks represent primary or secondary lexical syllable stresses. Phonemic units can be multi-character strings, provided they do not hamper a unique decomposition of the phonemic transcriptions.

An orthographic transcription is a text string that can be decomposed into graphemic units, and a graphemic unit can be a grapheme, a punctuation mark or a space. It has been demonstrated by Bouma (2000) and others that the g2p conversion is easier to formulate if one considers multi-character strings that usually translate to one phoneme as graphemes. Therefore, we also support the definition of graphemes like "aa", "ie", "eau" etc. Nevertheless, when aligning an orthographic transcription with a phonemic one (see Section 4) we will allow that "ie" for instance is decomposed into "i" and "e" if this is supported by the phonemic transcription.

In the following sections we will use the term *pattern* to indicate a sequence of consecutive units (phonemic or graphemic).

3. A two-step g2p converter strategy

The general architecture of the proposed two-step g2p conversion system is depicted on Figure 1.

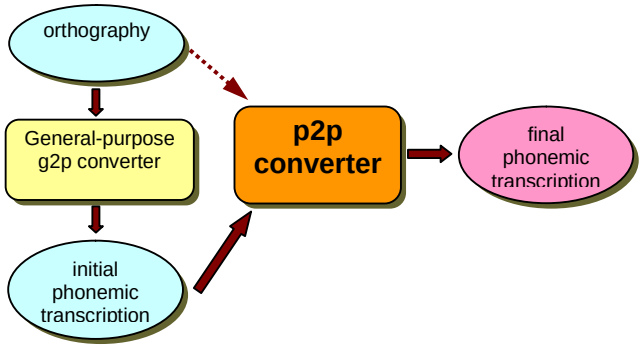

Figure 1 : Architecture of a two-step g2p converter

The general-purpose g2p converter creates an initial phonemic transcription which is then corrected by the p2p converter. In order to perform its work, the p2p converter can inspect both the initial phonemic transcription and the orthography of the name it has to process. The heart of the p2p converter is a set of stochastic correction rules, with each rule expressing the following:

> If a particular phonemic pattern (called the *rule input*) occurs in the initial phonemic transcription and if the context in which it occurs meets the *rule condition*, then it may have to be transformed, with a certain *firing probability*, to an alternative phonemic pattern (called the *rule output*) in the final transcription.

The rule condition can describe constraints on the identities of the phonemes surrounding the rule input, the stress level of the syllable associated with that input, the position of this syllable in the word, etc. It can also express constraints on graphemic patterns corresponding to the rule input and its neighborhood.

We will distinguish three types of correction rules: (1) stress substitution rules (SS-rules) which replace a stress mark by another (*no-stress* is also considered as a stress mark here), (2) phoneme substitution and deletion rules (PSD-rules) which transform a phonemic pattern into another one (including the empty pattern representing a pattern deletion) and (3) phoneme insertion rules (PI-rules) inserting a phonemic pattern at some position. The linguistic features for describing the context can be different for the respective rule types.

The rewrite rules can be implemented in the form of decision trees (DTs), like in (Andersen, 1996), or in the form of rule networks (RNs), like in (Yang & Martens, 2002). In the case of DTs each tree comprises the rules that apply to a particular rule input whereas in the case of RNs each network comprises the rules that perform the same input/output transformation. The DTs and RNs are learned automatically from training examples by means of machine learning algorithms that were previously applied with success to add pronunciation variants to the lexicon of an automatic speech recognizer (Yang, 2005).

4. Learning correction rules

The learning of correction rules is a four-step procedure. The aim of the first step is to align the initial phonemic transcription with the other two transcriptions, namely the correct phonemic and the orthographic transcriptions (the latter alignment is only needed if orthographic features will be used). The second step then retrieves from these alignments the rule input/output transformations that can explain the systematic errors made by the standard g2p. Given these transformations, the alignments are re-used (step 3) to generate the training examples from which to learn the correction rules. The final step is the actual rule induction from theses examples. The whole rule learning process is visualized on Figure 2.

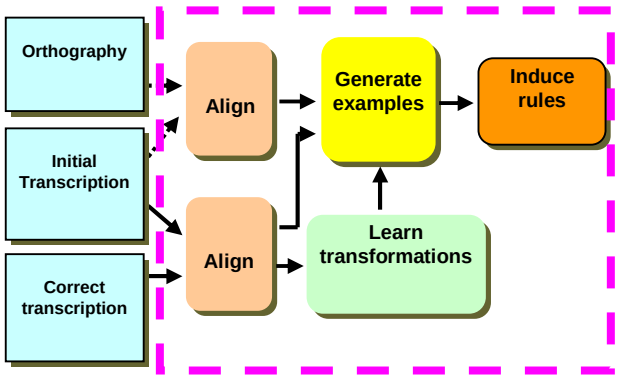

Figure 2 : Automatic learning of the p2p converter

4.1. Step 1 : transcription alignments

The p-to-p alignment between the initial and the correct phonemic transcription is intended to reveal the pattern transformations that can convert the initial transcriptions into correct ones. The alignment is represented in the form of two equally long *extended* transcriptions differing from the original transcriptions by the presence of so-called *empty* units, denoted as tildes. The following example shows such an alignment:

```
I:  ' r o . d $ ~ . b A ~ . '2 x l a n
C:  ' r o . d $ n . b A x . ~  ~ l a n
```

Figure 3 : alignment between the initial (I) and the correct (C) phonemic transcription

For the moment it suffices to note that this alignment reveals a missing /n/, a misplaced final syllable boundary (/./) and a superfluous secondary stress mark (/'2/) in the initial transcription. In the next section we will describe which transformations to derive from this example.

If the p2p converter selects a certain pattern in the initial phonemic transcription and if it wants to determine the graphemic context in which this pattern occurs, then it needs the grapheme-phoneme correspondences implied by the initial g2p conversion. Since we presume to have no access to internal information of the initial g2p converter, we need to perform an independent p-to-g alignment between the initial phonemic transcription and the orthography. An example alignment is the following:

```
I:  ' r o . d $ ~ . b A . '2 x ~ l a  n
O:  ~ r o ~ d e n ~ b a ~ ~  c h l aa n
```

Figure 4 : alignment between the initial (I) phonemic and the orthographic (O) transcription

It shows for instance that the phoneme /x/ comes from the graphemic pattern "ch" and that the final /a/ comes from the multiple-character grapheme "aa".

The p-to-p alignment is a DTW process which is driven by a simple correspondence model, consisting of an associated unit set per phonemic unit. It expresses that if a phonemic unit occurs in the initial transcription, the unit at the corresponding position of the correct transcription is normally expected to be the same unit or a unit belonging to its associated set. The p-to-g alignment is driven by a similar correspondence model, but now the associated set of a phonemic unit consists of graphemic units. This model expresses that a phonemic unit occurring in the initial transcription is expected to originate from a member of its associated set of graphemic units.

The p-to-p alignment is so designed that a boundary/ stress mark cannot be lined up with anything else than a boundary/stress mark. In the p-to-g alignment on the other hand, a boundary mark can be lined up with a predefined set of graphemic units (e.g. a space, a hyphenation mark, etc.) whereas a stress mark has no equivalent at all in the orthography. This means that a stress mark is always lined up with an empty unit.

Both alignment processes (p-to-p and p-to-g) are finally controlled by four probabilities: (1) the chance of lining up a unit of the 1st (initial) transcription with an empty unit of the 2nd (correct) transcription, (2) the chance of lining it up with a unit of its associated set, (3) the chance of lining it up with an eligible unit outside its associated set, and (4) the chance of lining up a unit of the 2nd transcription with an empty unit of the 1st transcription. These probabilities can be given common sense initial values first and can be further optimized using maximum-likelihood re-estimation.

The presented p-to-g alignment strategy can be considered as an extension of the "allowable mapping" approach proposed in Black et al (1998) and later utilized in Bouma (2000). However, a novelty is that if a multi-character grapheme corresponds to two phonemes having associated sets containing the constituent parts of this grapheme, the grapheme is decomposed. An example of this is *Aisha*: grapheme "ai" is pronounced as /a i/ here.

If both the p-to-p and the p-to-g alignments are available, extra empty units can be introduced to visualize the correspondences between all three transcriptions (see Figure 5 combining the alignments of Figures 3 and 4).

```
C: ' r o . d $ n . b A x . ~  ~ ~ l a  n
I: ' r o . d $ ~ . b A ~ . '2 x ~ l a  n
O: ~ r o ~ d e n ~ b a ~ ~ ~  c h l aa n
```

Figure 5: two alignments combined into one

Since all the extended transcriptions are equally long, the alignment can be viewed as a table with one column per unit and three units (from three transcriptions) per column.

4.2. Step 2 : transformation learning

The initial-to-correct pattern transformations that can convert the initial transcription into the correct one are retrieved from the p-to-p alignment. First we determine the stress mark transformations that are needed, and then we determine the phonemic pattern transformations.

Since we do not think it is easy to find rules for correcting the number of syllables, we have only tried to identify stress mark substitutions for initial syllables that are lined up with a correct syllable. From the example of Figure 3, we would retrieve one such a substitution /'2/-to-/'0/ with /'0/ representing the no-stress case.

Once the stress transformations have been identified, the stress marks are removed from the extended transcriptions. For the example of Figure 3 we then get

```
I:  r o . d $ ~ . b A ~ . x l a n
C:  r o . d $ n . b A x . ~ l a n
```

On this transcription pair we then perform a left-to-right (starting at the initial column) longest mismatch search. At a given position it determines the successive columns containing different units. The transformations are then obtained by removing the empty units from the input and output pattern found in these columns (these empty units are only there to visualize the alignment). For the given example we would retrieve two phonemic pattern transformations, namely //-to-/n/ and /.x/-to-/x./.

We collect all the transformations that can explain a sufficiently large fraction of the observed errors in a so-called transformation list.

4.3. Step 3 : training example generation

Once the transformation list is available, the generation of training examples can start. Recalling that we want to learn three types of correction rules: (1) stress substitution rules (SS-rules), (2) phoneme substitution and deletion rules (PSD-rules) and (3) phoneme insertion rules (PI-rules), we will also have to generate three types of examples. Each example consists of a rule input, a rule output and a set of features describing the linguistic context in which the rule input occurs.

For the generation of SS-examples we determine the corresponding syllables in the initial and the correct transcription. For each syllable of the initial transcription that is lined up with a correct syllable we then establish the input & output stress mark and the features describing the syllabic context. Examples of such features are: the stress levels of the previous and next syllable, the identity of the vowel (grapheme), etc. Obviously, if graphemic features are involved, the p-to-g alignment must be taken into account as well. Once the SS-examples have been generated, the stress marks are removed from the transcriptions.

For generating the PSD and PI-examples, we adopt a process involving (1) a segmentation of the initial transcription into modifiable patterns (rule inputs) and non-modifiable units, (2) a determination of the correct patterns (rule outputs) for the modifiable patterns, and (3) an extraction of the linguistic features describing the context in which the rule inputs occur. The modifiable patterns are non-empty inputs of transformations produced in step 2.

The segmentation of the initial transcription is a two-stage process. In the first stage, the transcription is aligned with a stochastic loop model comprising forward branches that consume a modifiable pattern (one branch per pattern that can occur), one forward branch that can consume any phonemic unit, and one feedback branch that does not consume any unit. The transition probabilities on the different branches are defined in such a way that the aligner will prefer modifiable patterns over other patterns and longer modifiable patterns over sequences of shorter ones. If there are several solutions with the same number of segments, the aligner will select the one with the largest product of rule input frequency counts (these counts are presumed to be available in the transformation list).

Once the initial segmentation is performed one knows how many non-empty modifiable patterns to select, and thus, how many PSD-examples to generate. However, before one can generate these examples one must also take the transformation list and the correspondences between the initial and the correct transcription into account. This can best be illustrated by an example. Suppose that the initial segmentation gave rise to the following result (now presented in tabular form) :

I:	..	I	~	.	k	o	.	v	E	s	.	~	t	r
C:	..	i	k	.	h	o	.	v	$	~	.	s	t	r

and that (/Es./,/$.s/) is in the transformation list. Then this segmentation has to be changed to

I:	..	I	~	.	k	o	.	v	E	s	.	~	t	r
C:	..	i	k	.	h	o	.	v	$	~	.	s	t	r

In general, a rule input pattern is extended at one or both ends with empty units originating from the p-to-p alignment (recognizable by a non-empty unit in the correct transcription) if by this extension one can increase the number of initial-to-correct transformations belonging to the transformation list. Once the final segmentation is established, one can determine for each identified rule input the corresponding correct output and the linguistic features describing the context in which the input occurs. Examples of such features are: the identities of the phonemes in the immediate vicinity of the rule input, the stress level of the syllable to which the rule input belongs, the initial grapheme of the graphemic pattern that was responsible for the production of the rule input, etc.

Once the PSD-examples are generated one can finally generate the PI-examples, one for every segment with a non-empty initial transcription. If no empty units are found in the initial transcription in front of this segment, the rule output is empty, in the other case it is the phonemic pattern under the found empty pattern. In the example above, there would be one real insertion, namely the //-to-/k/ in front of the first syllable boundary.

4.4. Step 4 : rule induction

The generated training examples are supplied to an inductive learning algorithm. Thus far, we have implemented two such algorithms: (1) a top-down divisive clustering algorithm producing a decision tree (DT) for each modifiable pattern appearing in the transformation list, and (2) a bottom-up agglomerative clustering algorithm producing a rule network (RN) for each transformation of the same list.

The TD learning algorithm will grow a DT by splitting a node into two sub-nodes on the basis of the most informative yes/no question that can be asked at that node. Since it has been shown that more robust trees can be learned if asking questions about whether a feature belongs to particular value class are allowed (Andersen, 1996), we have accommodated the facility to specify such value classes for the different feature types that appear in the linguistic description of the training examples.

The RN learning algorithm will first construct a layer of all the different feature combinations that were encountered in the training data at positions where a given transformation had to be performed. Then a network generator will create more and more general nodes by removing one feature at the time. Which features can be removed from a given node is determined on the basis of the expected importance of the features constituting this node. To that end the feature set is divided into subsets with increasing importance.

Both learning algorithms require a rule set evaluation criterion that can be used to decide whether a new node has to be added to a DT or whether a node can be eliminated from an RN. The two implemented criteria are: (1) entropy, measured on the training set (Yang & Martens, 2000) and (2) quality, measured on a validation set (Yang, 2005).

5. The actual p2p conversion

If the orthography is involved in the description of the correction rules, the p2p converter starts with performing an alignment between the initial phonemic transcription and the orthography.

The next step is the examination of each syllable of the initial transcription and the application of the SS-rules on syllables for which the conditions are met.

The third step consists of a segmentation of the initial phonemic transcription into modifiable patterns and non-modifiable units. The algorithm is the initial segmentation algorithm described in Section 4.3[2].

Once the segmentation is available, the pronunciation variant generator will try PI rules at the start of each non-empty segment and PSD rules at the start of each modifiable segment. If at a certain point one or more rules can be applied, different variants (including the one in which the input pattern is preserved) can be generated at the corresponding point in already created partial variants (Yang & Martens, 2002). The output of the pronunciation variant generator is a tree shaped network representing different transcriptions with different probabilities. The p2p converter will select the transcription with the highest probability as the final phonemic transcription. Obviously, one can expect that in a number of cases this transcription is identical to the initial transcription.

6. System implementation and evaluation

The methodology described in the previous sections was implemented in a software toolbox written in ANSI-C. It contains a transcription tool that can be used with and without a p2p converter being activated. A p2p converter is activated as soon as its rule set (produced by the learning tools) is uploaded.

The software allows the user to change the phonemic and graphemic unit set, the correspondence models controlling the aligners, the question list and the feature importance information driving the rule induction, the linguistic feature set, etc. This way it offers the flexibility that may be needed for the design of good dedicated p2p converters for medical terms, brand names, company names and other interesting name categories for which one is able to collect a proper training lexicon.

6.1. Experimental setup

The toolbox was used for the development of three g2p converters for three Dutch name categories: first names, surnames and geographical names (street & city names). The general-purpose g2p converter, designed for the transcription of words occurring in a general text, was the one embedded in the Belgian Dutch Nuance RealSpeak synthesizer (see http://www.nuance.com/realspeak).

The p2p converters are trained and tested on lexical databases containing one up to four manually verified transcriptions for each name. For each name category, the lexical database was divided into a training set and a test set. The test set is balanced with respect to criteria such as linguistic origin, length in characters, etc. of the names. Table 1 shows the number of names in the different sets.

The performance of a g2p converter is represented by its word error rate (WER), defined as the percentage of names of the test set which got an incorrect transcription, and by its word improvement rate (WIR), defined as the percentage of words with a better minus the percentage of words with a worse transcription than the one originating from a reference g2p converter (our reference is the general-purpose g2p converter).

category	training set	test set
first names	80059	8382
surnames	19556	3000
geographical names	20000	18056

Table 1 : Number of names in the training and test lexicons available for the three name categories

6.2. The first experiments

In this section we describe the first experiments we carried out so far. We only tested the impact of PSD rules, and we evaluated the transcriptions discarding the stress marks they contain.

We defined a graphemic unit set which is composed of 64 single-character units (letters, digits, punctuations) and 20 multi-character units such as /aa/, /ou/, /eau/, etc.

In a first attempt, we chose a set of 14 features to describe the rule conditions: four features specifying the phonemic context (2 phonemic units before and after the rule input), six features revealing the syllabic context (the stress levels and the identity of the vowel in the preceding, the present and the next syllable), two booleans indicating whether the rule input belongs to the initial or final syllable of the word, and two features describing the graphemic context (the initial unit of the graphemic pattern that produced the rule input, and a boolean telling whether this graphemic pattern was a multi-character pattern or not). The present feature set should by no means be considered as final. Future research is bound to find a better feature set.

Using the selected features we have learned two p2p converters per name category (we tested top-down rule learning in combination with entropy and quality as the evaluation criteria). The obtained g2p-p2p tandem performances are listed in Table 2 (in this Table "equal" means different from the reference, but equally good).

type	system	WER (%)	better (%)	equal (%)	worse (%)
first	g2p alone	34.2			
	g2p-p2pQ	30.0	8.8	0.6	2.9
	g2p-p2pE	29.7	8.7	0.7	2.2
sur	g2p alone	38.4			
	g2p-p2pQ	30.1	15.6	1.2	3.7
	g2p-p2pE	27.9	17.6	0.9	2.8
geo	g2p alone	32.5			
	g2p-p2pQ	22.6	16.5	1.3	1.9
	g2p-p2pE	22.3	17.4	1.4	2.1

Table 2 : performances of g2p and g2p-p2p systems on three name tasks (Q = quality, E = entropy)

All p2p converters were learned using the same settings of the control parameters. The number of learned correction rules ranged from 192 to 444. Although the differences are small, the best p2p converters for all name types were obtained using Entropy as the decision criterion.

7. Discussion and future work

The figures in Table 2 show that significant improvements over the baseline system are possible with small p2p

[2] Note that there is no need to extend the modifiable patterns (as during example generation) since there is no p-to-p alignment now, and thus no empty units that can result from it.

converters. For the *geo* name case, the WER is reduced by 30% and the p2p converter is able to transform more than 53% of the erroneous initial transcriptions to a better transcription. The WIR is 15.3% which is about 47% of the maximum attainable improvement of 32.5%.

Although these results are very valuable already, we conjecture that they can be surpassed by introducing a better feature set, and by implementing PI-rules and SS-rules on top of the PSD-rules tested thus far.

One research line we are pursuing in this respect is the development of a 'deductive' methodology. The aim is to use our linguistic knowledge as a top-down instrument to qualify differences between the initial transcriptions and the correct transcriptions. For geographical names we have observed, for instance, that many syllabification errors occurring in the initial transcription originate from the fact that the morphological integrity of topological entities such as 'kamp' (camp), 'erf' (estate), 'dijk' (dike), 'veld' (field) is not respected. Similarly, we have also observed a lot of syllabification errors in diminutive (first) names. The deductive approach will be explored along two lines. First of all, we will assess the improvement that can be obtained by implementing deductive rules to either the initial g2p transcriptions or the final transcriptions emerging from the g2p-p2p tandem. Secondly, we will incorporate graphemic patterns such as 'kamp', 'erf', 'tje', etc. in the linguistic feature set utilized by the inductive rule learner. This way we hope to find appropriate rules for coping with the mentioned morphemic problems. All this would then show the potential synergy of both the inductive and deductive approaches.

We also need to verify our claim that by using a g2p-p2p tandem we can do better than with a dedicated one-step g2p converter, trained on the same training lexicon the p2p converter is trained on. For geographical names we already trained one such a g2p using TIMBL (Daelemans et al, 2004). It yielded a WER of 22.1% and a WIR of 10.5% over the Nuance g2p. With the g2p-p2p-tandem the figures are 22.3% and 15.3% respectively.

Note that in applications involving different name types as well as ordinary words, the proposed g2p-p2p approach can lead to a much more memory efficient solution than an approach requiring multiple full-fletched dedicated g2p converters. For instance, the dedicated g2p emerging from TIMBL comprises all the training examples whereas our p2p converter needs to store only 192 correction rules.

8. Conclusion

This paper was devoted to the grapheme-to-phoneme conversion of proper names. We have established that the commercially available general-purpose g2p converters partly fail on such a task, and we have therefore proposed to adopt a two-step approach in which the initial transcriptions generated by such a converter are corrected by a phoneme-to-phoneme converter which is learned fully automatically on the basis of a pronunciation directory containing names from the envisaged name category (with their correct transcription of course). We have also proposed a novel learning methodology and implemented it in the form of a software toolbox. The first tests on first names, surnames and geographical names have demonstrated that significant improvements can be achieved. For geographical names for instance, 53% of the erroneous initial transcriptions were converted to a better transcription, and less than 1.4% of the correct initial transcriptions appeared to contain errors after p2p conversion. These results were obtained with a small rule set (192 rules) and a very simple feature set for describing the rule condition. We argue that larger improvements are possible with more intelligently chosen features, e.g. morphemically inspired graphemic patterns, which can be detected by means of a 'deductive' approach.

9. Acknowledgement

The presented work was carried out in the Autonomata project, granted under the Dutch-Flemish STEVIN program. The project partners are the universities of Gent, Nijmegen and Utrecht and the companies Nuance and TeleAtlas. The university of Utrecht and the two companies are acknowledged for designing and providing the name dictionaries that were used for the training and evaluation of the p2p converters.

10. References

Andersen, O.; Kuhn, R.; Lazarides, A.; Dalsgaard, P.; Haas, J. and Nöth, E. (1996). "Comparison of two tree-structured approaches for grapheme-to-phoneme conversion", Procs. ICSLP, Kobe, 1700-1703.

Black, A., Lenzo, K., Pagel, V. (1998). "Issues in building general letter to sound rules. Procs. ESCA/COCOSDA workshop on Speech Synthesis (Jenolan Caves), 77-81.

Boula de Mareüil, P.; d'Alessandro, C.; Bailly, G.; Béchet, F.; Garcia, M.; Morel, M.; Prudon, R. and Véronis, J (2005). "Evaluating the pronunciation of proper names by four French grapheme-to-phoneme converters", Procs. Interspeech, Lisbon, 1521-1524.

Bouma, G. (2000). "A finite state and data-oriented method for grapheme to phoneme conversion", Procs. ACL, 303-310.

W. Daelemans, J. Zavrel, K. van der Sloot, A.van den Bosch (2004). TiMBL: Tilburg Memory Based Learner, 5.1, Reference Guide. ILK Technical Report 04-02 (http://ilk.uvt.nl/downloads/pub/papers/ilk0402.pdf)

S. Quazza , H. van den Heuvel (2000). "The use of lexica in text-to-speech systems." In: F. Van Eynde, D. Gibbon: Lexicon development for speech and language processing. Kluwer Academic Publishers, Dordrecht, Boston, London, 207-233.

Yvon, F.; Boula de Mareüil, P.; d'Alessandro, C.; Aubergé, V.; Bagein, M.; Bailly, G.; Béchet, F.; Foukia, S.; Goldman, J.; Keller, E; O'Shaughnessy, D.; Pagel, V.; Sannier, F.; Véronis, J and Zellner, B. (1998). "Objective evaluation of grapheme to phoneme conversion for text-to-speech synthesis in French", Computer Speech and language 12, 393-410.

Yang, Q. and Martens, J.P. (2000). "Data-driven lexical modeling of pronunciation variations for ASR," Procs ICSLP (Bejing), vol. 1, 417-420

Yang, Q; Martens, J.P.; Ghesquiere, P.J. and Van Compernolle, D. (2002). "Pronunciation variation modeling for ASR: Large improvements are possible but small ones are likely to achieve," Procs PMLA (Estes Park, Colorado), 123-128.

Yang, Q. (2005). "Data-driven approaches to pronunciation variation modelling for automatic speech recognition", PhD thesis, University Gent.