FACTA UNIVERSITATIS (NIŠ)
SER.: ELEC. ENERG. vol. 21, no. 1, April 2008, 91-105

A Hybrid Approach to Bilingual Text-To-Phoneme Mapping

Enikő Beatrice Bilcu and Jaakko Astola

Abstract: In this paper, we address the problem of bilingual text-to-phoneme (TTP) mapping in which the phonetic transcription of isolated written words must be found. In general, in the bilingual/multilingual TTP mapping for isolated words, two processing steps are applied to each input word. The language of each word is first identified and then the letters of the word are translated into their phonetic transcriptions according to the recognized language. We use the multilayer perceptron (MLP) neural network for the letter to phoneme conversion task and a hybrid approach composed of a MLP and a decision rule system for the language recognition task. We introduce a new bilingual TTP mapping system and we provide an analysis of the influence of several different factors on its phoneme accuracy.

Keywords: Neural network, multilayer perceptron, error back-propagation with momentum, text-to-phoneme mapping, phoneme accuracy.

1 Introduction

In text-to-speech systems, TTP mapping is one of the early steps which generates a sequence of phonemes corresponding to the input text. The synthetic speech is then produced from this phonetic transcription. The simplified block diagram of a text-to-speech synthesizer is depicted in Fig. 1. The block denoted as ”Bilingual/multilingual TTP mapping” is responsible for transcription of the input word into the corresponding phoneme string. For each phoneme the block ”Concatenate corresponding sound units” assigns a sound unit from a database called ”Stored sound units”. The speech is then generated in the ”Speech synthesizer” block.

The problem of TTP mapping can be classified in several different dimensions. From application point of view, TTP mapping can be divided into two main classes.

Manuscript received on January 3, 2008.

The authors are with Institute of Signal Processing, Tampere University of Technology, Korkeakoulunkatu 1, 33720, Tampere, Finland (e-mail: bilcub@cs.tut.fi).

In continuous TTP mapping the input text is translated into the corresponding phonemes and the dependence between the adjacent words is taken into account. In this case a word may be pronounced differently depending on the context in which it appears. An example is the word *the* which has a different phonetic transcription for different contexts. Another class of TTP mapping is the phonetic translation of isolated words. In this case, the words are generated independently and pronunciation of a certain word does not depend on the previous and subsequent words. Another dimension is to classify the TTP mapping systems based on the number of languages to which the input words can belong. If the written words can only belong to one input language, the TTP mapping is called monolingual. In such a case, the TTP mapping system must perform only the transcription of the written text into the corresponding phonemes. When the input words can belong to two or more languages the bilingual/multilingual TTP mapping is addressed and language identification is included into the system. In the bilingual approach the first step is to identify the language to which the input text belongs. After that the TTP mapping subsystem corresponding to the identified language generates the output phoneme string.

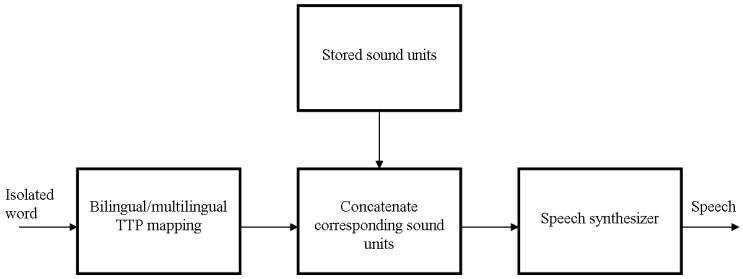

Fig. 1. A simplified block diagram of a Text-To-Speech synthesis system.

A very large number of approaches have been proposed to deal with the problem of TTP mapping. Many text-to-speech (TTS) systems use rule-based approaches or solutions based on dictionary look-up tables [1]. Dictionary look-up tables necessitate the storage of a large amount of data and writing phonetic rules, required in the rule-based systems, is a time consuming and difficult task. Alternatively, several data-driven solutions have been proposed. One such a solution is to use decision trees (DT) to generate the phonetic transcriptions as suggested in [2] and many other publications. Other alternatives are the Hidden Markov Model (HMM) [3] and the N-gram [4, 5] approaches. Solutions based on analogical reasoning [6] and memory-based learning [7] have also been proposed. Neural networks have been used, in the last few decades, in many speech processing applications such as speech and language recognition, speech coding and speech synthesis

to mention a few. One of the early text-to-speech (TTS) systems was the NETtalk, introduced in [8], which used a three layered neural network for text-to-phoneme mapping. The NETtalk system demonstrated that even a small NN can capture a significant part of the regularities and irregularities in the English pronunciation which ensure good TTP mapping accuracy. Starting from the NETtalk approach several other neural network based systems have been proposed [9–15]. Other TTP mapping systems implementing combinations of the above mentioned techniques have been also proposed in the open literature, for the monolingual as well as for the multilingual case [15]. In the case of multilingual TTP mapping, one important part of the system is the language identification module [10].

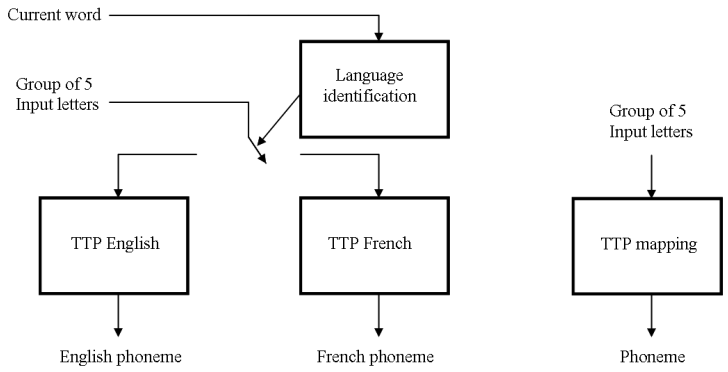

Fig. 2. The block diagrams of the bilingual (left) and monolingual (right) TTP mapping systems.

In this paper, we address the problem of bilingual TTP mapping for isolated words. The written words can be either English or French and the pronunciation of the current word is independent of the previous and next input words. We propose a hybrid system composed of multilayer perceptron neural networks and decision rules and we study its performance, in terms of phoneme accuracy, in several different scenarios.

2 Bilingual Text-to-Fhoneme Mapping

In bilingual TTP mapping, for isolated words, the first step is the language identification (in our case either English or French as shown in Fig. 2). After the language of the whole input word was estimated all its letters are transcribed into phonemes. The transcription is done in the TTP mapping block corresponding to the identified language (either ”TTP English” or ”TTP French” block). In the mapping process, besides the current input letter, also context information from 4 adjacent letters (2

previous and 2 next letters) is taken into account. Thus, the TTP system takes as input a group of 5 adjacent letters.

For the approaches based on decision trees and analogical reasoning there is no need to translate the letters and phonemes to numerical values. For processing by a neural network, the input letters, the output phonemes and the language tags must be translated into numerical values. Moreover, when monolingual or multilingual TTP mapping systems are implemented, they must be trained first on some set of words for which the phonetic transcription and the language are known. As a consequence, prior to implementation and training of a TTP mapping system, the available database must be pre-processed. In this section we review the steps which we have followed for database pre-processing and we describe the encoding method for letters, phonemes and language. The dictionaries used for training and testing the neural networks in our experiments were the Carnegie Mellon University (CMU) pronunciation dictionary [16] for the English words and the Brulex dictionary [17] for the French words. We denote these dictionaries as *cmu.dic* and *brul.dic* respectively. The *cmu.dic* contains 108080 English words and the *brul.dic* contains 32245 French words. Both dictionaries were pre-processed in the following manner:

1. The words and their phonetic transcriptions were aligned, such that there is a one-to-one correspondence between letters of each word and its phoneme symbols [18]. Corresponding to the letters that have no pronunciation a so-called *null phoneme* (_) is introduced in the phonetic transcription in order to have equal numbers of letters and phonemes for a given word. In the case when the phonetic transcription of a single letter consists of 2 or more phonemes they are combined together in a compound phoneme. For instance the word *ox* have the phonetic transcription *A: c s*. In this case the phonemes 'c' and 's' are combined together in the compound phoneme 'cs' that corresponds to the letter 'x'. After the alignment, the number of the letters and the number of phonemes are equal for each entry in the dictionary.
2. In order to eliminate the ambiguity that can occur for multiple pronunciations of the same word, only one phonetic transcription was chosen for each input in the dictionary.
3. Both *cmu.dic* and *brul.dic* were split into two parts. From the *cmu.dic* we randomly chosen 80% of the words for training (each word with a single phonetic transcription). The obtained training dictionary was denoted as *train_en.dic* and contained 86464 words. The remaining 20% (21616 words) from the "cmu.dic" formed the testing dictionary "test_en.dic". In the same manner *brul.dic* was split into *train_fr.dic* (25796 words) and *test_fr.dic* (6449 words). In addition we obtained the file *train_en_fr.dic* by concatenation of

train_en.dic and *train_fr.dic*. The files *train_en.dic*, *test_en.dic*, *train_fr.dic* and *test_fr.dic* were used for training and testing the neural networks responsible for TTP mapping of a single language (English or French). The "train_en_fr.dic" was used to train the neural network for language recognition.

4. The order of the words in the training and in the testing dictionaries was randomized. This was done to increase the modeling capability of the NN modules [19]. After that, each letter in a word was encoded using binary vector codes as shown in Table 1 or random codes. Letter encoding together with phoneme and language encoding are further described in the next subsection.

2.1 Letter, phoneme and language encoding

We begin first with the details of the input letter encoding. The English language contains 26 letters and the French language contains 39 letters and we note that the French alphabet contains all the English letters. As a consequence, it is enough to have an encoding for the French alphabet. We have used in this paper two letter encoding schemes: binary orthogonal codes and randomly generated codes. Other codes were studied for instance in [11] for the monolingual case. Since there are 39 letters in the French alphabet the binary vectors must have length 40 to encode 39 letters plus the *graphemic null* (denoted as \0). The *graphemic null* is used to represent the spaces between words. Examples of the binary letter codes are shown in Table 1. Several studies have concluded that orthogonal codes provide better phoneme accuracy than non-orthogonal codes when neural networks are used for TTP mapping [9]. This is why we selected these orthogonal encoding schemes in our approach.

Table 1. Binary codes used to encode the input letters. The elements of the vectors are zeros except one which equals unity and is placed on the position corresponding to the letter index.

Letters	Corresponding binary codes of length 40
\ 0	1 0 0…0
a	0 1 0…0
b	0 0 1…0
⋮	…
ü	0 0 0…1

The letter codes shown in Table 1 are not the only orthogonal codes that can be implemented (see for instance [11] and the references therein). Another possibility

for letter encoding is to randomly generate the input letter codes. In this case, for each of the 39 letters and the *graphemic null*, we have selected the elements of the vectors representing the letter codes from a random zero-mean Gaussian-distributed sequence with unity variance. The length of the code vectors was in this case 40.

Table 2. Binary codes used to encode the phonemes. The elements of the vectors are zeros except one which equals unity and is placed on the position corresponding to the phoneme index.

Phonemes	Corresponding binary codes of length 62
_	1 0 0...0
A	0 1 0...0
A:	0 0 1...0
⋮	...
ä	0 0 0...1

In our system we use NN for both text-to-phoneme transcription and language identification. Since the outputs of the neural networks are numerical values, the phonemes and the language tags must be numericaly encoded. In this paper, we used a similar encoding approach as for the input letters. The phonemes were encoded using binary vectors with all 0's except one unity element which corresponded to the phoneme index (there are 62 English and French phonemes together). Examples of phoneme encoding are shown in Table 2 where in the first line the code of the *null phoneme* is shown. For language encoding we have used the following binary codes: for English $[0\ 1]$ and for French $[1\ 0]$.

2.2 Multilayer perceptron for text-to-phoneme mapping

Once the database is prepared and the letters, phonemes and languages are encoded we can build the bilingual TTP mapping system. In this subsection we describe in detail the TTP mapping blocks for English and French words transcription (denoted as TTP English and TTP French in Fig. 2). In the next subsection we describe the implementation and the functionality of the language identification block and then the overall bilingual TTP mapping is introduced and we show how it is used for letter to phoneme translation. To implement the TTP English and TTP French blocks we have used a three layer MLP neural network with one input layer, one hidden layer and one output layer. Such a network is depicted in Fig. 3 where the synaptic connections and the activation functions are also shown.

The multilayer perceptron was trained, to learn the letter to phoneme correspondences, using the error back-propagation with momentum algorithm (cf. [19]). In detail the algorithm does the following:

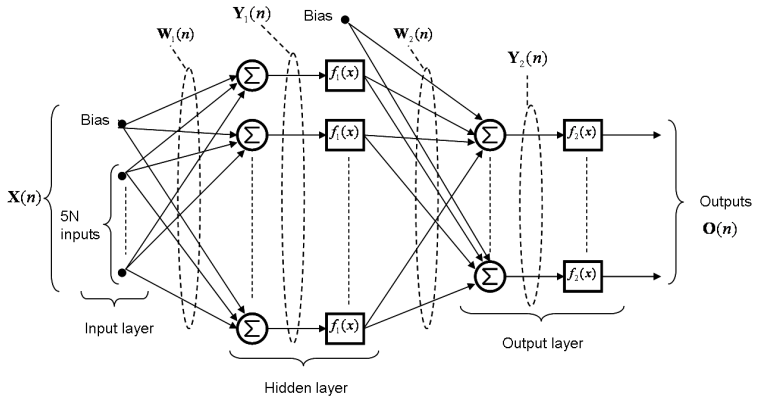

Fig. 3. A detailed diagram of the multilayer perceptron neural network used in our experiments.

1. Generate the inputs of the neural network by concatenating the vectors corresponding to 5 adjacent letters (as we have seen in the previous subsection all letters from the dictionary were encoded as vectors):

$$\mathbf{X}(n) = \left[\mathbf{L}_1^t(n)\ \mathbf{L}_2^t(n)\ \mathbf{L}_3^t(n)\ \mathbf{L}_4^t(n)\ \mathbf{L}_5^t(n)\ 1\right]^t, \tag{1}$$

where n is the iteration number, $\mathbf{x}^t$ represents the transposed of vector $\mathbf{x}$, $\mathbf{L}_i(n)$ is the vector corresponding to the i^{th} input letter in the neural network (for instance for the letter "a" we have $\mathbf{L}_i(n) = [0\ 1\ 0 \ldots 0]$ if we use the binary orthogonal codes from Table 1), 1 stands for the input bias and $\mathbf{X}(n)$ is the vector of the inputs. Since we deal with isolated word TTP mapping, the first letter of each word goes in the middle position of the input vector. In this case $\mathbf{L}_1(n)$ and $\mathbf{L}_2(n)$ correspond to the *graphemic null*. Similarly, the last letter of the current word goes always in the middle of the input vector and the last two vector codes, $\mathbf{L}_4(n)$ and $\mathbf{L}_5(n)$ are set equal to the *graphemic null*. In this manner, the context dependence between adjacent words during training and testing is not taken into account. Of course, in applications where the context dependence is important, such as the case of continuous TTP mapping, the first 2 letters and the last 2 letters are taken from the previous and next word respectively. The input window $\mathbf{X}(n)$ of 5 adjacent letters is shifted over the entire current word and the subsequent training steps are followed for each position of the input window.

2. Compute the induced local field of the hidden layer:

$$\mathbf{Y}_1(n) = \mathbf{W}_1(n)\mathbf{X}(n), \tag{2}$$

where $\mathbf{W}_1(n)$ is an $M \times (5N+1)$ matrix containing the synaptic connections between the inputs and the hidden neurons. The length of the input vector is $5N+1$, where N is the length of an input letter code and 1 stands for the input bias and M is the number of hidden neurons.

3. Compute the output of the hidden layer by applying the activation function $f_1(z)$ to every element of the vector $\mathbf{Y}_1(n)$:

$$\mathbf{Z}(n) = f_1\left(\mathbf{Y}_1(n)\right). \tag{3}$$

4. Compute the induced local field of the output layer:

$$\mathbf{Y}_2(n) = \mathbf{W}_2(n)\left[\mathbf{Z}^t(n) \quad 1\right]^t, \tag{4}$$

where $\mathbf{W}_2(n)$ is an $P \times (M+1)$ matrix containing the synaptic connections between the hidden neurons and the outputs, P is the number of neural network outputs and $M+1$ is the number of hidden neurons plus the bias of the hidden layer.

5. The outputs $\mathbf{O}(n)$ of the neural network are then obtained by applying an activation function to $\mathbf{Y}_2(n)$:

$$\mathbf{O}(n) = f_2\left(\mathbf{Y}_2(n)\right). \tag{5}$$

The hyperbolic tangent activation function $f_1(x)$ was implemented in the hidden layer and the softmax function $f_2(x)$ has been used at the output of the neural network:

$$f_1(x_i) = \frac{1-\exp(x_i)}{1+\exp(x_i)}, \qquad f_2(x_i) = \frac{\exp(x_i)}{\sum_{j=1}^{P}\exp(x_j)}. \tag{6}$$

The softmax activation function gives a good approximation of the class posterior probabilities [19]. As a consequence, for every input pattern presented at the input of the NN, at the output we obtain a vector of length 62 that has a maximum value in the position corresponding to the recognized phoneme. For example if the phoneme A : correspond to a certain input pattern the 3^{rd} element of the output vector will be the maximum (see Tab. 2).

The synaptic weights are modified by back-propagation of the output error through the neural network, as follows:

1. Compute the output error vector:

$$\mathbf{e}(n) = \mathbf{D}(n) - \mathbf{O}(n). \tag{7}$$

with $\mathbf{O}(n)$ being the output of the neural network computed in (5) and $\mathbf{D}(n)$ is the vector of the corresponding known phoneme (during training of the NN's words with known phonetic transcription are used. For each letter of a word the vector $\mathbf{D}(n)$ of its corresponding phoneme is known.).

2. For the output neurons the weight changes are given by:

$$\Delta W_{2_{ij}}(n) = \alpha \Delta W_{2_{ij}}(n-1) + \lambda \delta_{2_i}(n) Z_j(n), \tag{8}$$

with $\Delta W_{2_{ij}}(n)$ being the correction applied to the ij element of the matrix $\mathbf{W}_2(n)$, $\alpha = 0.9$ is the momentum constant, $\lambda = 0.1$ is the learning rate parameter:

$$\delta_{2_i}(n) = e_i(n) f_2'(Y_{2_i}(n)), \tag{9}$$

where $e_i(n)$ is the i^{th} element of $\mathbf{e}(n)$ obtained in (7), $f_2'(x)$ is the first derivative of $f_2(x)$ and $Z_i(n)$ is defined in (3).

3. For the hidden neurons the weight changes are given by:

$$\Delta W_{1_{ij}}(n) = \alpha \Delta W_{1_{ij}}(n-1) + \lambda \delta_{1_i}(n) X_j(n), \tag{10}$$

with $\Delta W_{1_{ij}}(n)$ being the correction applied to the ij element of the matrix $\mathbf{W}_1(n)$ and:

$$\delta_{1_i}(n) = e_{h_i}(n) f_1'(Y_{1_i}(n)), \qquad \mathbf{e}_h = \mathbf{W}_2^t(n) \delta_2(n) \tag{11}$$

with $f_1'(x)$ being the derivative of $f_1(x)$.

The above described training algorithm is applied to the MLP neural network in on-line mode where at each iteration a vector containing the codes of 5 adjacent letters are presented to the network. The error between the output of the neural network (estimated phoneme corresponding to the center input letter) and the desired known phoneme is computed. The error is back-propagated through the network and the synaptic weights are modified. The input window is then shifted one letter and again 5 adjacent letters (current letter, 2 letters at left and 2 letters at right of the current letter) are input into the neural network. The error between the NN output and the vector of the known corresponding phoneme is computed and back-propagated through the neural network in order to update the synaptic weights. The above mentioned training algorithm is used to update the synaptic weights of both MLP neural networks that perform the phonetic transcription of English words and the phonetic transcription of the French words.

After the synaptic weights of the MLP neural networks are trained, they can be used in the TTP mapping framework. When the phoneme transcription of an input word must be generated for each letter of the word the corresponding neural

network is computed using (1)-(5). For each output network the corresponding recognized phoneme C_c is then calculated using the following criterion:

$$C_c = \underset{i=1,\ldots,62}{\operatorname{argmax}} \left(\mathbf{O}(n)\right). \tag{12}$$

where C_c is the index of the recognized phoneme into the list of total phonemes (see Table 2).

For example, if the second element of $\mathbf{O}(n)$ is the maximum, then $C_c = 2$ and the recognized phoneme is *A*.

2.3 The hybrid system for language identification

We describe here the sub-system implemented for language identification. The block diagram of this sub-system in the learning mode and in the recognition phase is depicted in Fig. 4. The block denoted as MLP5 is composed of a three layer MLP neural network trained using a similar method as the one described above. The difference between this neural network and the one used for TTP mapping relies on the fact that the output of this network represents the estimated language and not the corresponding phoneme. As a consequence the output vector $\mathbf{O}(n)$ of this neural network will have length 2. For training this NN words that belong to known languages are used. At each iteration, a group of 5 adjacent letters are taken as inputs to the neural network and the output vector of length 2 is generated. The error between the output vector of the neural network and the known language vector of the current letter is computed. This error is back-propagated and the synaptic weights are computed as described in the previous subsection. While the training of this neural network is done in on-line mode, when the network is used for language identification of unknown words it operates in batch mode. This functionality will be described in more details in the sequel. We mention here that a similar equation as (12) is used at the output of this neural network to identify the language of new words in the recognition phase. If the first element of the output vector is maximum the assigned language is French otherwise it is English.

Besides the MLP5 neural network, the language identification subsystem contains also a decision rule block. This is implemented to increase the language recognition accuracy and to estimate the language of the entire word. The MLP5 assigns a language tag to each letter of the current word whereas the decision rule have two roles. On one hand it identifies situations when the language of the entire word can be estimated with unity probability by some if/else rules. On the other hand in situations when the language of the current word cannot be estimated with unity probability by some simple if/else rules, the decision block analyzes the language tags of each letter, generated by the MLP5, and finally assigns the language

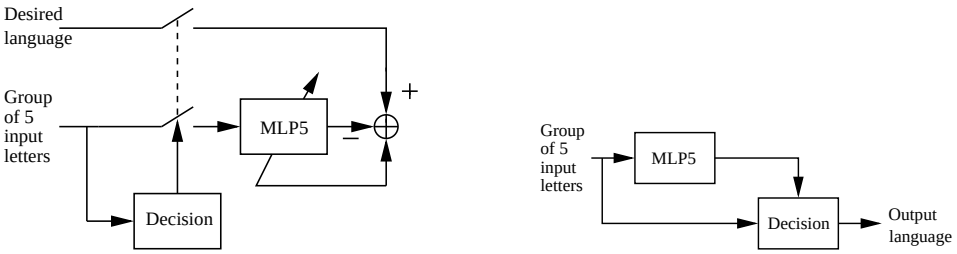

Fig. 4. The block diagram of the language identification system in training mode (left) and in recognition mode (right).

of the entire word. The language identification of unknown words is done in the following manner: 5 adjacent letters are presented to the input of the system (the first input pattern contains the first letter of the word in the middle position and the last input vector contains the last letter of the word in the middle position). The input vector goes first into the block denoted as ”Decision” where it is checked if the language can be identified with certainty. If so, the language tags of all letters from the current word are set accordingly and the language identification of the current word is stopped. If not the input vector is presented to the neural network which assigns a language tag for the central letter (current letter of the word). Next, the input window is shifted one letter forward and the process continues. In this manner, if the language corresponding to one of the letters from the current word is identified with probability 1, the entire word is assigned with that language and the language identification stops. If none of the letters could be classified with probability 1 as English or French, the neural network is used to assign them a language tag. When the end of the word is reached the number of letters that are assigned to English and the number of letters that are assigned to French are counted. If there are more English letters then French letters the entire word is assigned to English and to French otherwise. One can argue that such a strategy of language identification implemented for each word separately and independent of the adjacent words is incorrect. The reader should keep in mind that the problem addressed in this paper is isolated TTP mapping in which isolated words are pronounced by the machine. This is different than continuous TTP mapping addressed in [5] where better language accuracy can be obtained by analyzing larger portions of text.

The block denoted as ”Decision” is implemented using some decision rules based on the input letters. First we note again that the English alphabet is contained in the French alphabet which includes in addition some specific letters. If one of the French specific letters are found in the current input vector $\mathbf{X}(n)$ the entire current word is classified as French. Moreover, we have performed a set of tests to verify if groups of several adjacent letters could be directly classified to English

or French with certainty. Due to memory limitations we have tested only groups of 4 adjacent letters and we found out that 74% of the total number of possible groups can be directly classified as either English or French. As a consequence the "Decision" block works as follows: if one of the input letters is a French specific letter the language of the current word is French. Otherwise, the first 4 letters of the input vector ($\mathbf{L}_1(n)$, $\mathbf{L}_2(n)$, $\mathbf{L}_3(n)$ and $\mathbf{L}_4(n)$) are checked if they have a unique language correspondence (English or French). If this group of 4 letters have a unique correspondence the language of the current word is assigned accordingly with probability 1.

As we can see from Fig. 4 the decision block influences also the training of the MLP5 neural network such that not all input patterns are used to train the MLP neural network for language recognition. If some input patterns can be a priori classified with probability 1, as belonging to either English or French language, they are not used to modify the synaptic weights of the neural network. In this manner, those patterns that can be classified by some decision rules and represent noise for the training of the NN are discarded from training.

3 Simulations and Results

In this section we present experimental results obtained with the proposed bilingual TTP mapping system and comparisons with some previously published approaches. In order to have a fair comparison, the training parameters and the sizes of the neural networks for both methods were equal. In the training process the synaptic weights of all neural networks were initialized with random values uniformly distributed in the range $[-1,\ 1]$ and the training algorithm was error back-propagation with momentum described in Section 2. The learning rate parameter was set equal to 0.1 and the momentum constant was $\alpha = 0.9$.

First, we did a set of experiments where we varied the size of the neural networks responsible for language recognition and TTP mapping for English and French. The results, in terms of phoneme accuracy, for both English and French language are shown in Table 3 for the case when the input letters were encoded using binary codes and for the case of random encoding the input letters. Comparing the results shown in Table 3 we can see that better phoneme accuracy for French language is obtained with random codes for sizes larger than 10500 synaptic weights while English words are transcribed more accurate when binary codes are used. We note also that the phoneme accuracy of French words is about 5% larger when random codes were used to encode the input letters compared to the case of binary codes while the phoneme accuracy of the English words only drops by 2% (see for example the values obtained for 10500 synaptic weights). The reason can

be found in the language recognition step which provides better results in the case of random codes. This is also demonstrated by the results reported in [11] where it was shown that random codes can improve the phoneme accuracy of small sized neural networks for monolingual TTP mapping (in our experiments the smaller neural network was used in the language recognition sub-system). This leads us to the conclusion that different input letter encoding schemes can be used for the 3 neural networks involved.

Table 3. Phoneme accuracy, for different number of synaptic weights, obtained with the proposed hybrid approach.

	Random letter codes		Binary letter codes	
Number of synaptic weights	English	French	English	French
2100	49.42%	44.23%	65.92%	65.03%
10500	77.64%	79.84%	79.44%	74.81%
16000	78.68%	80.62%	80.51%	75.32%
22000	78.57%	81.05%	80.53%	79.09%

Based on the above observations, we implemented the final bilingual TTP mapping system in a slightly different manner. The architecture of the system is similar to the one depicted in Fig. 2 with the main difference that we used random vectors to encode the inputs of the language recognition and French TTP mapping modules. In the TTP mapping module responsible for the translation of the English words we have used binary encoding of the input letters. As a consequence, the final bilingual system must also contain a module that translates the binary codes into random codes which we have implemented using a look-up table.

In Table 4 we show the comparative results obtained with the proposed approach when the input letters into all 3 neural networks were encoded using binary and random codes. We also show in this table the results of the hybrid approach from [20] and the phoneme accuracy obtained when binary codes were used in the neural networks responsible for English TTP mapping and random codes in the French TTP mapping and language recognition. We can see from these results that improved performance is obtained with the proposed hybrid approach especially for the French language.

The results shown in Table 4 might look on the low side since in [13] phoneme accuracy levels around 90% was reported for English language. However, the results reported in [13] were obtained in a different framework. In that publication the single language approach was studied while in this paper we address the problem of bilingual TTP mapping. As one can expect the phoneme accuracy in our approach, for both English and French words, drops due to the imperfect language identification. There are also differences in the topology of the NN implemented

Table 4. Comparison between the proposed bilingual hybrid system and the approach from [20].

Language	Binary letter codes	Random letter codes	Binary letter codes for English NN. Random letter codes for French NN. Random letter codes for language NN.	Hybrid [20]
English	80.53%	78.57%	80.05%	80.04%
French	79.09%	81.05%	81.21%	73.86%

and in the selection of the training dictionary. For instance in [13] non-symmetric windows were used to include context dependence between adjacent letters while in our approach we have used symmetric ones.

4 Conclusions

In this paper we have studied the problem of bilingual TTP mapping implemented by a combination of neural networks and decision rules. Through extensive simulations we have studied the influence of the neural network size and input letter encoding into the phoneme accuracy of the system. Based on the results of this study we introduced a new hybrid approach to the problem of bilingual text-to-phoneme mapping. The proposed system was implemented in the context of isolated word transcription from text where the input words can be either English or French. Our bilingual TTP mapping is composed of a hybrid language recognition part implemented by means of a small neural network and decision rules. It also contains two TTP sub-systems that are responsible for the phoneme transcription of the two languages. The bilingual approach introduced here shows better performance, in terms of phoneme accuracy, for both English and French words compared with a previously published method.

References

[1] T. Vitale, "An algorithm for high accuracy name pronounciation by parametric speech synthesizer," *Computational Linguistics*, vol. 17, no. 3, pp. 257–276, 1991.

[2] E. Wong and S. Sridharan, "Three approaches to multilingual phone recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2003*, Hong Kong, Apr. 2003, pp. 44–47.

[3] T. Ogawa and T. Kobayashi, "Hybrid modeling of PHMM and hmm for speech recognition," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'03*, Hong-Kong, apr 2003, pp. 140–143.

[4] W. Wang and D. Vergyri, "The use of word n-grams and parts of speech for hierarchical cluster language modeling," in *Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'06*, France, May 2006, pp. 1057–1060.

[5] W. B. Cavnar and J. M. Trenkle, "N-gram based text categorization"," in *Proc. of 3rd Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, US, 1994, pp. 161–175.

[6] Y. Marchand and R. I. Damper, "A multistrategy approach to improving pronunciation by analogy," *Computational Linguistics*, vol. 26, pp. 195 – 219, June 2000.

[7] A. van den Bosch and W. Daelemans, "Data-oriented methods for grapheme-to-phoneme conversion," *European Chapter of ACL*, pp. 45–53, 1993, utrecht.

[8] T. J. Sejnowski and C. R. Rosenberg, "Parallel networks that learn to pronounce english text," *Complex Systems*, vol. 1, pp. 45–168, 1987.

[9] E. B. Bilcu, "Text-to-phoneme mapping using neural networks," Ph.D. dissertation, Tampere University of Technology, Tampere, Finland, Jan. 2006.

[10] E. B. Bilcu and J. Astola, "A hybrid neural network for language identification from text," in *Proc. of IEEE International Workshop on Machine Learning for Signal Processing, MLSP 2006*, Ireland, Sept. 2006, pp. 253–258.

[11] E. B. Bilcu, J. Astola, and J. Saarinen, "Comparative study of letter encoding for text-to-phoneme mapping," in *Proc. of the 13th European Signal Processing Conference, EUSIPCO 2005*, Turkey, Sept. 2005.

[12] R. A. Cole, J. W. T. Inouye, Y. K. Muthusamy, and M. Gopalakrishnan, "Language identification with neural networks: A feasiblity study," in *Proc. of IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, jun 1989, pp. 525–529.

[13] M. Embrechts and F. Arciniegas, "Neural networks for text-to-speech recognition," in *Proc. of the IEEE International Conference on Systems Man and Cybernetics*, 2000, pp. 3582–3587.

[14] J. Tian and J. Suontausta, "Scalable neural network based language identification from written text," in *Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2003*, Hong Kong, Apr. 2003, pp. 48–51.

[15] I. T. Podolak and S.-W. Lee, "Hybrid neural system for ponematic transformation," in *Proc. of the 15th International Conference on Pattern Recognition, ICPR 2000*, Spain, sep 2000, pp. 44–47.

[16] The Carnegie Mellon University. [Online]. Available: www.speech.cs.edu

[17] F. Content, P. Mousty, and M. Radeau, "Brulex: Une base de donnees lexicales informatisee pour le francais ecrit et parle," *L'Annee Psychologique*, pp. 551–566, 1990.

[18] R. I. Damper, Y. Marchand, J. D. Marsters, and A. I. Bazin, "Aligning text and phonemes for speech technology applications using an em-like algorithm," *International Journal of Speech Technology*, vol. 8, no. 2, pp. 49–162, 2000.

[19] S. Haykin, *Neural Networks - A Comprehensive Foundation.* New York, USA: Prentice-Hall, 1999.

[20] E. B. Bilcu, J. Astola, and J. Saarinen, "A hybrid neural network rule/based system for bilingual text-to-phoneme mapping," in *Proc. of the 14th IEEE International Workshop on Machine Learning for Signal Processing, MLSP 2004*, Brazil, Sept. 2004, pp. 345–354.