Neural Networks for Text-to-Speech Phoneme Recognition

Mark J. Embrechts (embrem@rpi.edu) and Fabio Arciniegas
Department of Decision Sciences and Engineering Systems
Rensselaer Polytechnic Institute, Troy, NY 12180

Abstract

This paper presents two different artificial neural network approaches for phoneme recognition for text-to-speech applications: Staged Backpropagation Neural Networks and Self-Organizing Maps. Several current commercial approaches rely on an exhaustive dictionary approach for text-to-phoneme conversion. Applying neural networks for phoneme mapping for text-to-speech conversion creates a fast distributed recognition engine. This engine not only supports the mapping of missing words on the database, but it can also mitigate contradictions related to different pronunciations for the same word. The ANNs presented in this work were trained based on the 2000 most common words in American English. Performance metrics for the 5000, 7000 and 10000 most common words in English were also estimated to test the robustness of these neural networks.

1 Introduction

This paper describes two different artificial neural network (ANN) approaches for text-to-speech conversion: Staged Backpropagation Neural Networks (SBPNNs) and Self-organizing Maps (SOMs). Both neural networks can be applied to large general texts and build on the pioneering work of Sejnowski and Rosenberg (i.e., NetTalk [10]).

NetTalk uses a single neural network to deal with all phoneme cases. The work presented in this paper is different in the sense that multiple stages were applied and, that a different alignment structure for the mapping between letters-to-phonemes was used for the backpropagation as well as the SOM neural networks.

In both SBPNNs and the SOMs, the first neural network stage distinguishes between single and dual phoneme cases (i.e., one letter is mapped to two phonemes). In the second stage two different neural networks are used in parallel to deal with one and two-phoneme cases separately.

Several current commercial methods for text-to-phoneme conversion rely on an exhaustive dictionary approach. However, not all words are included in the dictionaries and therefore the phoneme recognition performance is compromised. One common approach to mitigate this problem is to mine a database for determining the best matches between the word's syllables and those contained in the database. However, this process requires large amounts of memory.

The use of soft computing in lieu of hard computing to implement text-to-speech conversion is a matter of trade-off. Applying neural networks for phoneme mapping for text-to-speech conversion creates a fast distributed recognition engine. This engine not only supports the mapping of missing words in the database, but it also reduce contradictions related to different pronunciations for the same word.

NetTalk introduced a slicing window where the letter to be mapped to phonemes are placed in the middle of the window (i.e., central window positioning [2]). In the alignment approach presented in this paper the window positioning structure is changed to Second Position Asymmetric Windowing (SPAW), where the number of spaces before and after the object letter (i.e., the letter which will be mapped to a phoneme) are not equal. It is called "Second Position" because the object letter is located in the second space from the middle of a central window (Fig. 1). Second Position Asymmetric Windowing will be labeled "N-M SPAW" for notation purposes, where N and M refer to the number of spaces before and after the central window position. This concept will be explained in more detail later in this paper.

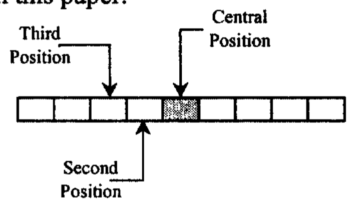


Fig 1. Different Windowing Positions

The human brain uses sets of semantic rules or routes to read. Two different lines of study for analyzing these semantic rules have been proposed: the single and the dual semantic route. The single route uses only a dictionary based lookup procedure (lexical route) for text-to-phoneme conversion. The dual route uses a letter to phoneme rule procedure (nonlexical route) in addition to the dictionary lookup procedure. Following the NetTalk approach, only the single semantic route was used in this work.

This paper is organized as follows: section 2 introduces the text-to-phoneme conversion approach, section 3 describes the staged backpropagation and SOM neural network models and discuss the neural networks performance, and in section 4 the conclusions are summarized.

2 The Text-to-Phoneme Conversion

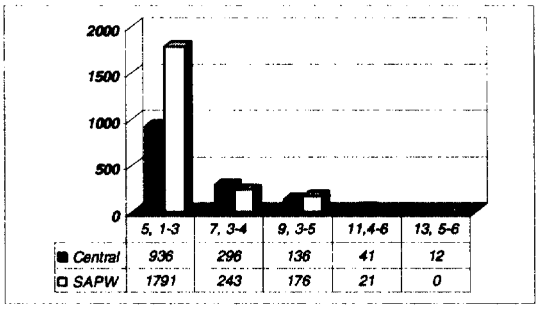
Three fundamental issues need to be addressed for text-to-phoneme mapping with staged neural networks: training data, window alignment, and context.

The 2000 most common American English words [14] were selected for developing a text-to-phoneme staged neural network and performance metrics were estimated for the 5000, 7000 and 10000 most common words in American English. The 1000 most common words in American English were also used in order to compare our results with NetTalk [10]. We used the 1000 most common words from the CUE practice set [13], which is different from the 1000 most commonly words used by NetTalk, based on The Webster's Pocket Dictionary. The Carnegie Mellon Pronouncing Dictionary (CMPD) was utilized for generating a windowed training dataset [12]. CMPD is a publicly available web-based machine-readable pronunciation dictionary for North-American English that contains over 100,000 words and their phonetic transcriptions. The implemented CMPD phoneme set contained 39 phonemes, for which the vowels may carry lexical stress (0 for no stress, 1 for primary stress, and 2 for secondary stress). As of yet, voice stress-related features were not implemented. A 40th phoneme was added to represent the blank or punctuation marks. The staged networks are trained based on the 2000 most common words in American English.

Window alignment is important because a unique map from text to phonemes is needed to generate the training/test sets for the neural network. Words are sliced starting with the first letter in the window and shifted to the left until the whole word has been passed by. This implies that the number of pattern per word will be equal to number of letters in the word. Issues related to the choice for the appropriate window size are discussed by Bullinaria [4]. Bullinaria considered only central windowing. The first issue relates to the proper choice for the window size in order to accommodate any long-range dependencies. A large window size implies that many units and connections would be vastly underutilized because of the prevalence of empty window spaces. The use of a series of recurrent connections in lieu of a sliding window was also addressed by Bullinaria [3], but did not offer any improvement in performance.

Different arrangements for central windowing and Second Position Asymmetric Windows (SPAW) alignments were investigated. Sometimes two different phonemes can be mapped from the same information window. This is considered an inconsistency. The main idea is to explore how much information (spaces) is needed in order to minimize the number of inconsistencies per arrangement. Inconsistencies are counted for different window representations of the objective letter for both central and SPAW alignments (Fig. 2). It has been found that Second Position Asymmetric Windowing alignments lead to fewer inconsistencies. A SPAW representation requires also less information (i.e., shorter window size).

Pronunciation of a particular letter or string of letters depends on the context. Therefore, context information should be used to generate pronunciation rules with a realistic generalization of unknown words and non-words [5]. Context related issues were not taken into consideration in this study.



	5, 1-3	7, 3-4	9, 3-5	11,4-6	13, 5-6
■ Central	936	296	136	41	12
□ SAPW	1791	243	176	21	0

Fig. 2. Number of Inconsistencies found in the 2000 Most Common Words in American English using both Central and SAPW Windowing Alignments.

3 The Staged Neural Network Model

For both neural network approaches, a categorical representation (or class representation) was used to represent the input information. This categorical representation assigns to each window space a set of 26 binary values, one for each letter of the English alphabet (i.e., a window with 7 spaces results in 182 inputs for the neural network). For the phoneme output representation 40 categorical values were used (one for each phoneme and the 40th neuron represents the blank space). For the Backpropagation ANN, output values were encoded as 0.1 or 0.9 to avoid saturation problems.

Initially the neural networks were trained on 80 percent of the 2000 words considered and tested on the remaining 20 percent. This results in a training set of 10,251 patterns when all the possible window positions are considered (Fig. 2). For the set of phonemes used in this study, 0.83% of the samples were cases in which the match between letters and phonemes was not one-to-one (single phoneme case) but one-to-two (dual phoneme case) [1].

First attempts to deal with dual phoneme cases used one or two slabs of 40 categorical neurons for representing the phonemes. However, for the Backpropagation ANN, no successful networks resulted from this approach (i.e., overall recognition was around 77%, but none of the two-phoneme cases was really recognized). Hence, it was decided to follow a staged neural network approach. The neural network in the first stage distinguished between single and dual phoneme cases. The second stage consists of two parallel neural networks to deal with the one and the two-phoneme cases separately.

3.1 Backpropagation Neural Network

Different neural network configurations (i.e., windowing, number of neurons in hidden layer, and number of hidden layers) were tested to obtain the best model for the text-to-phoneme mapping for both stages. Different combinations of inputs, alignment structures, size, and outputs configurations were analyzed for each neural network (Fig. 4). For the first stage, all window sizes were tested. However, for the ANN's for the single phoneme case in the second stage, only large windows were used (4-6 and 5-6 SPAW and 11 and 13 central windowing). All neural networks trained were halted using an early stopping criterion.

The first stage ANN initially utilized two categorical output neurons with target values [0 1] or [1 0] depending on whether the outcome was a single or dual phoneme. The best neural network employed a 1-4 SPAW window and one hidden layer with 11 neurons. It was able to recognize 100% and 67% of single and dual cases, respectively. A second approach used 40 neurons in the output layer. For this approach a 1-5 SPAW and a neural network with two hidden layers (43 and 67 neurons, respectively) was able to reach 100% recognition for both single and dual phoneme cases.

For the second stage different ANN's with different window sizes were considered as well. Based on the results obtained for the first stage, only one output layer with 40 neurons and categorical inputs were used for the second stage. For the single phoneme case ANN, a 4-7 SPAW and two hidden layers was large enough to accommodate all long-range dependencies. Different numbers of neurons were tried for both layers. A weak optimum was found with 67 and 91 neurons for the first and second hidden layer, respectively. This ANN achieved 97% accuracy for a test set using 20 percent of the 2000 most common words in American English. The best net for the dual phoneme case used a 3-5 SPAW and two hidden layers (with 43 and 67 neurons, respectively) and achieved 100% accuracy.

Sejnowski, and Rosenberg trained NetTalk using the 1000 most commonly occurring words from the Webster's Pocket Dictionary based on frequency counts in the Brown corpus [10]. The best performance achieved was 98% on the 1000 word corpus, and 80% and 91% without and with additional training on the 20,012 words in the Webster's corpus, respectively. The word corpus used in this paper, which is based on the CMPD [12] dictionary from Carnegie Mellon University and other word frequency counts referred in the Carnagie Mellon's web site ([13] and [14]), were different to the one used by NetTalk. For the 1000 most common words, the difference in words was approximately 200 (The recognition reached was 99%).

In order to test the robustness of the staged neural network model, performance metrics are compared for the 2000, 5000, 7000 and 10000 most common words in American English obtained from [12], [13] and [14] (Table II). For the first staged neural network, the recognition level was 100% for the 2000 and 5000 most common words. However, for the 7000 and 10000 most common words the first stage performance was 99% due to the presence of a new phoneme case. Once that case was introduced into the training process, the recognition level was 100% again. Also, many of the new words do not contain dual phoneme cases, which contributes to the robustness of the first stage of the model.

For the second stage single case neural network there is a decrease in the level of recognition for the ANN trained on the 2000 most frequent words, resulting in 94%, 91% and 85% recognition for the 5000, 7000 and 10000 most common words in American English, respectively [1].

Table II. Staged Backpropagation ANN Performance Results.

Training Set	***Performance over the Most Common Words in American English***			
	2000	5000	7000	10000
2000	97%	94%	91%	85%
5000	(98%)	95%	92%	84%
7000	(94%)	(91%)	89%	82%

Note: Corpus and word frequency counts were obtained from the CMPD, and the LOB and ACL_DCI corpus.

3.2 Self-organizing Maps

The SOM is used for its visualization capabilities in order to explore relationships between letters and phonemes and to see whether phoneme clusters can be clearly delineated.

Self-Organizing Maps (SOM) were developed by Professor T. Kohonen of Helsinki University in Finland [6]. A SOM is an unsupervised feedforward neural network trained by competitive learning. The neurons are usually arranged on a virtual 2-D grid and the self-organizing characteristics of Kohonen's training algorithm cause likewise inputs (molecules) to activate nearby neurons on the Kohonen map.

The SOM is a topology-preserving map from a high-dimensional input descriptor space to a two-dimensional grid or plane. A SOM describes relative faithfully the distribution of data points embedded in a high-dimensional space onto a plane. The property of preserving topology implies that a SOM groups similar input data to nearby neurons in the map. Therefore, the SOM can serve as a visual clustering tool for high-dimensional data .

The SOMs were generated with Viscovery, a commercial software package for SOM visualization [11]. Viscovery does not impose a limit to the number of samples or variables and exhibits outstanding visualization options. Key SOM (heuristic) parameter settings for Viscovery are the size of the map, the map ratio, the map tension and the variable priority. The size of the map determines the number of nodes in the map (in this case a hexagonal grid). A *heuristic* recommendation is to use ten times as many nodes as number of input vectors. However, the map size was set in 20,000 neurons, which is the maximum number of neurons possible in Viscovery. Heuristics for map display can be obtained from the userforum e-mail list from Eudaptics Gmbh (viscovery-userforum@eudaptics.com).

The map ratio describes the relation between the width and the height of the map. The map ratio was derived from the ratio of the *principal plane* of the source data set. The principal plane is spanned by the two longest eigenvectors of the autocorrelation matrix of the data distribution. The map ratio is chosen automatically by Viscovery.

The map tension represents the reach of the neighborhood function at the end of the training process. Its value depends on the objectives of the study. For instance, maps with low tension will be more detailed. An initial heuristic value of 0.2 was used.

The variable priority gives additional weight to a component by multiplying its internal scale by that factor. If the variable priority setting exceeds unity, the corresponding component will have more impact on the distance metric [11]. Priority factors were used for the objective letters and phonemes in order to bias the topological structure. For the maps presented in this paper, priority factors of 2 were used (Priority factor greater than 2 resulted in similar performance).

A Kohonen self-organizing map (SOM) was trained using the 2000 most common words in American English by passing the words through a 4-7 SAPW. As in the Backpropagation case, the first SOM distinguishes between phoneme cases and then two parallel SOMs handle each phoneme case separately. SOMs are unsupervised and often do not use separate training and validation sets. However, in order to test the phoneme map, random training and validation sets were generated (80% and 20%, respectively).

The test set for the 2000 most common words was presented to the SOM. The phonemes will be estimated, once the SOM is generated, from the K nearest neighbors based on local learning or association. Starting from the reference point P, the weights of the K nearest neighbors ($w_1 \ldots w_K$) are used to determine the corresponding distances from the nearest neighbors ($d_1 = |w_1 - P_1|, \ldots, d_j, \ldots, d_K = |w_K - P_k|$), and which are then sorted in ascending order. A new distance weighted value for the phonemes are then estimated based on the distance-weighted responses of the K nearest neighbors. Viscovery provides several options for the weight function: uniform (i.e., all neighbors equally weighted), gaussian (i.e., neighbors with medium distance are weighted more), linear (i.e., neighbors are weighted according to the inverse of the distance) and, quadratic [11].

Viscovery uses a Gaussian weight function by default and reduces the size of the neighborhood when the training progresses. The number of neighbors was selected using a visualization tool provided by Viscovery (i.e., the *neighbor page*, which shows the neighborhood for any cell in the map). Only neighbors within the same cluster were considered.

Using the test set obtained from the 2000 most common words, for the first stage SOM a 100% recognition level was reached. For the dual phoneme case SOM in the second stage, also 100% recognition was obtained. However, for the single phoneme case SOM (Fig. 3), only 87% recognition level was obtained.

These results in the single phoneme case SOM are lower than those obtained using the Backpropagation NN and NetTalk. One reason for the poorest results might be the number of neurons used to create the map (less than 2 neurons per sample).

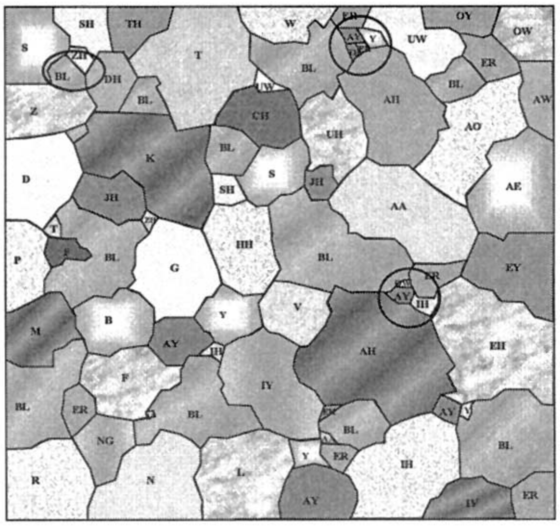


Fig. 3. Phoneme Self-Organizing Map

Having lower amounts of neurons per sample might result in a decrease on the topological ordering capabilities of the SOM. Looking at Fig. 3, there are many regions that might indicate the presence of inconsistencies in the information being used (i.e., similar letter patterns can lead to different phonemes). This can be clearly seen in the three regions marked in Fig. 3 (i.e., phonemes /AY/, /Y/, and /UW/ are coming from similar letter structures, as well as phonemes /AH/, /AY/, /IH/, /OW/, /SH/ and /ZH/). If more neurons would be added to the SOM, the Kohonen algorithm would then create clusters for each case, avoiding these inconsistencies.

The information being used in this paper does not include context and intonation. As it was noted by Bagshaw [2], many words share the same structure but they are pronounced different depending on their context. Therefore, it is possible also that these inconsistencies might be due to the no inclusion of the context.

On the other hand, Fig. 4 shows the overall organization of the SOM. It can be seen that vowels are located in the left side and consonants in the right side of the map. The SOM, also, clearly distinguish phonemes only related with one letter (i.e., /DH/ and /TH/ are only related with 'T'). However, for some phonemes such as /AH/, /UH/ and /UW/, more than three different vowels can be related to them (i.e., all vowels are related to /AH/). Some vowel phonemes are not located inside the vowel region. These phonemes (/AY/, /IH/, /IY/, and /ER/), are related with the letters 'Y', 'W', and 'R' so that there were correctly located in the SOM.

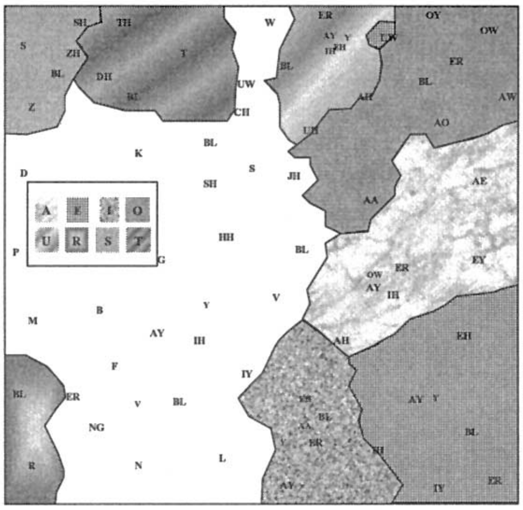


Fig. 4. Distribution of Vowels and Consonants

Finally, SOMs for the 5000, 7000 and 10000 most common words in American English could be not obtained due to the limitations in the number of neurons in Viscovery. However, the levels of recognition obtained using the 2000 most common words SOM were more consistent than those obtained using a backpropagation NN. The results are shown in Fig. 5.

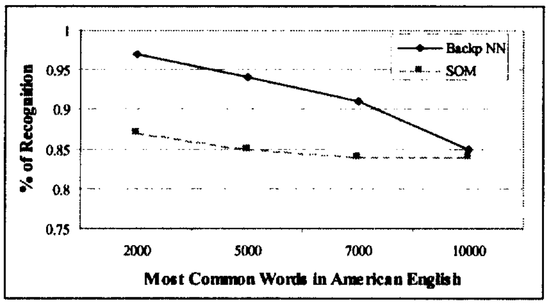


Fig. 5. SBPNNs and SOMs Recognition Levels

4 Conclusions

This paper described two different implementations for text-to-speech conversion with a two-staged backpropagation neural network and Self-Organizing Maps. It is novel in the sense that staged neural networks were used and a better window alignment structure (SPAW) was applied. This staged approach first recognizes whether the letter shown has to be mapped into a single (one-to-one) or a dual (one-to-two) phoneme case and does the actual phoneme matching in the second stage.

For the 2000 most common words in American English, 100 % recognition in the first stage and the dual phoneme case in the second stage was obtained for both neural network approaches. For the single phoneme case, 97% and 87 % of recognition were reached for the backpropagation and SOM, respectively. Good recognition levels (91 % backpropagation approach), however, can be achieved for up to 7000 of the most common words in American English. For more than 7000 words it was found that many of the new words either do not belong to the American English (i.e., are exotic) or show complicated structures and are therefore hard to learn by the ANN. SOMs, also, are more robust than SBPNNs.

5 References

[1] Fabio Arciniegas, and Mark J. Embrechts, "Artificial Neural Networks (ANNs) for Phoneme Recognition for Text-to-Speech Applications," Proceedings of the 2000 IEEE-INNS-ENNS International Joint Conference on Neural Networks, Como, Italy.

[2] Paul C. Bagshaw, "Phonemic Transcription by Analogy in Text-tot-Speech Synthesis: Novel Word Pronunciation and Lexicon Compression", Computer Speech and Language, Vol. 12, pp. 119-142, 1998.

[3] J. A. Bullinaria, "Internal Representations of a Connectionist Model of Reading Aloud," Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society, pp. 84-89, 1994.

[4] J. A. Bullinaria, "Neural Network Learning from Ambiguous Training Data," Connection Science, Vol. 7, pp. 99-122, 1995.

[5] J. A. Bullinaria, "Modeling Reading, Spelling, and Past Tense Learning with Artificial Neural Networks," Brain and Language, Vol. 59, pp. 236-266, 1997.

[6] T. Kohonen, "The Self-organizing Map," in Proceedings of the IEEE, Vol. 78, No. 9, pp. 1464 - 1480, 1990.

[7] M. Norris, "Time, Memory, Change and Structure in the NETtalk text-to-speech network," Proceedings of ACNN'96 Cognitive Models, Workshop Case Study, Australia, 1996.

[8] M. Patel, "Using Neural Nets to Investigate Lexical Analysis," Proceedings of PRICAI'96: Topics in Artificial Intelligence, pp. 241-252, 1996.

[9] M. Seidenberg and J. McClelland, "A Distributed, Developmental Model of Word Recognition and Naming," Psychological Review, Vol. 96, No. 4, pp. 523 - 568, 1989.

[10] T. J. Sejnowski, & C. R. Rosenberg, "Parallel Network that Learn to Pronounce English Text," Complex Sytems, Vol. 1, No. 1, pp. 145-168, 1987.

[11] Viscovery SOMine Lite Manual, Version 2.1. Eudaptics Software Gmbh, Austria 1998.

[12] www.speech.cs.cmu.edu/cgi-bin: The CMPD Pronouncing Dictionary. Carnegie Mellon University.

[13] http://www.uri.edu/comm_service/cued_speech/1000most.html. CUE practice with the 1000 most common words.

[14] http://gopher://gopher.sil.org/11/gopher_root/linguistics/info/. LOB and ACL_DCI corpus frequency list.