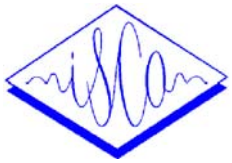

6th European Conference on
Speech Communication and Technology
(EUROSPEECH'99)
Budapest, Hungary, September 5-9, 1999

AUTOMATIC MODELING OF DURATION IN A SPANISH TEXT-TO-SPEECH SYSTEM USING NEURAL NETWORKS

R. Córdoba, J. A. Vallejo, J.M. Montero, J. Gutierrez-Arriola, M.A. López, J.M. Pardo
Grupo de Tecnología del Habla. Departamento de Ingeniería Electrónica. Universidad Politécnica de Madrid
E.T.S.I. Telecomunicación. Ciudad Universitaria s/n, 28040 Madrid, Spain
cordoba@die.upm.es
http://www-gth.die.upm.es

ABSTRACT

Accurate prediction of segmental duration from text in a text-to-speech system is difficult for several reasons. One specially relevant is the great quantity of contextual factors that affect timing and how to model them. There are many parameters that affect duration, but not all of them are always relevant. We present a complete environment in which to decide which parameters are more relevant in different situations and the best way to code them. The system is based in a neural network absolutely configurable, and the main effort is made in the parameters to be used, including the contextual effects using windows of variable length.

Keywords: prosody, duration, text-to-speech, neural networks, parameter selection and coding.

10.21437/Eurospeech.1999-367

1. INTRODUCTION

The primary goal of this study was to develop an automatic system to model duration for a Spanish text-to-speech system and achieve better results than those obtained with our previous system, which used a rule-based approach with a multiplicative model. We will compare them as both have used the same database to train (model) the system.

At the same time, we wanted the system to be very flexible, so we could adapt it easily to new contexts, in special to restricted prosody domains, which will be our next work. Many studies have been carried out lately using neural networks with success [1][4]. Find the parameters that are significant for duration modeling has always been an issue [3][6][7]. We analyze new alternatives for the parameters used and its codification as inputs to the neural network.

Our database consists of five sets of phrases of different lengths and patterns, giving a total of 732 phrases (15,141 phonemes). In its design, the goal was to cover all the different patterns in Spanish [5] and use it to model both fundamental frequency [see a through study in [5]] and duration. So we have considered two types of sentences in the same database:

- Real text: a discourse, with predominance of enunciative patterns, and a colloquial text, with more interrogative and exclamative sentences.
- Laboratory sentences: designed to complement the previous text. It includes at least one sample of all the possible patterns with one or two tonic syllables in independent structures with up to eight syllables.

The segmentation has been automatic, using a continuous speech recognition system with HMM models, followed by a manual revision of the marks.

We have used first about 50% of the database for training and 50% for testing. Then we changed the proportion to 75-25. The division has been made according to phrases, not to phonemes, trying to be as homogeneous as possible.

The system will be included in a text-to-speech system in Spanish [3], that can be tested on-line at the address: http://www-gth.die.upm.es/research/synthesis/synth-form-concat.html.

2. DESCRIPTION

We have focused our work in the following problems:

1) Which topology should we use for our network?
2) Which is the best way to code the parameters?
3) Which are the best parameters that we should use?

2.1 Topology of the neural network

We have used a multilayer perceptron (MLP) and the sigmoid as activation function.

Our basic unit is the phoneme. For each phoneme, we compute a series of parameters, we code them and use those values as inputs to the neural network. There is one output in our network: the duration of the phoneme.

We have made experiments using one or two hidden layers and a variable number of neurons. The necessity of two hidden layers is not always obvious. It is very difficult to know the optimum number of neurons and layers that the net should have. A big number usually provokes overtraining and a small one can be insufficient. Our approach has been to begin from the very beginning: just three neurons.

The best procedure is to increase the number of neurons one by one and observe the test results (training results are not significant for our problem, as they always improve using more neurons), and stop when there is an overtrain symptom (decrease for the test set).

Another important consideration to decide the optimum number of neurons is the size of the training set. As the number of neurons increases, the number of weights to be trained increases too, and the training set can be too small to generalize.

2.2 Codification of the parameters

We have considered different ways present the parameters to the neural network, i.e., the way they are coded, as we have different kind of parameters.

1) **Binary coding**: this is the standard coding for true/false parameters.
2) **One-of-n**. We use n neurons and only one of them is active, the one that corresponds to the class or category. This method has a potential problem: the vectors can be very similar. A solution is to use equilateral coding, where there is one neuron less than the number of classes. All vectors are equidistant and the distance is the maximum possible. This type of coding is useful if you have a big number of classes.

In ordinal values there is a relationship of order between the different values. For example, the position of a unit inside a higher-order unit. For these values we have considered three codifications:

3) **Porcentual transformation**: divide the current value by the maximum value to obtain a percentage. We have a floating-point value as input.
4) **Thermometer**: divide all the possible values in different classes (intervals). We activate all the neurons until we get to the current class and leave the remaining neurons inactive.
5) **Z-Score mapping**: apply a normalization to the floating-point value that takes into account the average and the standard deviation of the variable. It is a good codification for very variable parameters.

2.3 Network evaluation

To evaluate and compare the networks we have considered different metrics for the error (difference between the prediction from the network and the optimum value). The most important metric is the MSE, or the RMS, which is equal to sqrt(MSE). They are both more reliable than the average absolute error.

To make our comparisons it is better to use an adimensional metric, because it will be independent of the way we code the duration, for example the following one:

$$\text{Relative RMS error} = \frac{RMS}{\sqrt{\sum t_i^2}}$$

where t_i are the optimum values.

This metric has a problem: it has an offset. In the same line, it is better to use the following metric, which does not include the offset, and is independent of the average value of the magnitude compared:

$$\text{Relative RMS error}(2) = \frac{RMS}{\sqrt{\sum (t_i - \bar{t})^2}}$$

2.4 Parameters to be used

This is the most important item and will be the vehicle for our experiments. The number of experiments made is too high, so we will only mention the most significant ones and just comment some of the "unsuccessful" ones.

In preliminary tests we found out that it is too difficult to decide which parameters are relevant and the best way to code them using a very big network with many parameters, because the differences in performance are too small and not always coherent. So, we have used a base experiment using only the phoneme identity and the stress, which are the most relevant ones without doubt. Then, we have added the different parameters one by one to see their effect and the significance of each of them. We have considered the introduction of contextual information in some of them, i.e., windowing information.

We have made the experiments in four directions:

- First, we used a fixed number of neurons (3) to test each possible parameter.
- As increasing the number of parameters demands more neurons to the network, we increased the number of neurons until we obtained an overtrain symptom.
- We tested different codifications for non-binary parameters.
- We included all the parameters to get the final network.

2.4.1 Phoneme identity

This is the most obvious one. We have considered a set of 33 phonemes for Spanish and used a one-of-n coding.

Contextual phonemes: we have considered to use the phonemes that are to the right and to the left of the current one. As the number of phonemes is too high (we need 99 inputs for the three phonemes) the results obtained are low, showing that there are not enough examples in the database to train all the possible contexts. The solution is to make clusters of similar phonemes: we divide our set of phonemes in 13 classes and classify the left and right context phonemes in these classes. This way we reduce the inputs to 59 (33+13+13). The results improve with this approach. So we will use it in our reference system. Experiments using a two phonemes context showed a decrease for the test set, so we discarded this option.

2.4.2 The stress

The effect of this parameter is always important. The coding is binary: the phoneme can have stress or not. We have obtained better results using a window of five stress values to include contextual information. See Table 3.

2.4.3 Binary parameters

We show below another binary parameters that have been considered in our experiments. As the coding is

fixed, we have worked in their effect for different numbers of neurons. All of them have shown an improvement over the reference experiment (see Table 1). That is what we expected, as all of them affect the duration.

- Stress in the syllable. This is the best one.
- Syllable beginning with vocal.
- Diphthong.
- Phoneme in a function word.

The last two apply only to special cases, so they do not show a significant improvement in the overall system, but they improve the prediction in their specific cases.

2.4.4 Type of phrase

We have considered at first eight types of phrases. Our first experiment was to use a one-of-n codification (8 inputs), but the results were even worse than the reference experiment. A through examination of the database showed that the distribution of phrases was not uniform, some types had significantly less examples than the others.

The solution was to reduce the number of types to 4 giving a more uniform distribution. Using again a one-of-n codification, the results were really good (Table 1).

2.4.5 Position in the phrase

We have considered different alternatives for position parameters: position of the phoneme in the syllable, word and phrase; the syllable in the word and phrase; and the word in the phrase. In our first approach we made the following steps for the codification:

- Normalize the value of position by the total in the higher-order unit – we obtain a floating point value between 0 and 1.
- This value is coded using 4 classes. The intervals that define these classes are computed automatically looking for uniform distributions.
- The 4 classes use a thermometer-type coding with 3 neurons (always the number of classes minus 1).

The first results using just three neurons showed a very little improvement. Our decision was to increase the number of neurons and use a different number of classes for each parameter, that will be close to its average value. The results are shown in Table 1.

The main conclusion is that we can not use all the parameters at the same time, because they provide similar information to the network. Another conclusion is that the deviation of the parameter "position of phoneme in phrase" is too high to be modeled with enough generalization.

Although the results do not show significant differences between them, we consider the best option to use: phoneme in the syllable, syllable in the word and word in the phrase, as they are more homogeneous when used together, their range of values is smaller and they need less neurons and classes to reach an optimum.

2.4.6 Number of units in the phrase

In a similar way than for positions, we have considered the number of phonemes in the syllable, word and phrase; syllables in the word and phrase; and words in the phrase. We followed these steps for their codification:

- Normalize the value of position by the maximum value – we obtain a floating point value between 0 and 1.
- Apply Z-score (using average and standard deviation) as it is the usual recommendation in the neural network literature [2]: we can restrict at our will the operating range of the parameter, which is too variable.

The conclusions are similar. All parameters referred to phrase provide worse results, what is due to the broad range of values. We have made experiments using the thermometer-type codification instead of the floating point but all were worse.

The summary of most relevant results is shown in Table 1. We have been able to find the right codification for all the parameters, as there is a improvement in all of them.

Experiment	Neu-rons	Rel RMS – Train	Rel RMS – Test
Reference experiment	3	0.77454	0.82536
1- Ref. + Stress in syllable	3	0.76773	0.82420
	4	0.76427	0.82316
2- Ref. + Diphthong	3	0.77078	0.82523
	5	0.75583	0.82300
3- Ref. + Syllable beginning with vocal	3	0.76859	0.82454
	5	0.75679	0.82267
4- Ref. + Function word	3	0.76816	0.82500
5- Ref. + Type of phrase (8 types)	3	0.76830	0.82577
Ref. + Type of phrase (4 types)	3	0.76981	0.82481
	7	0.74368	0.81982
6- Ref. + Pos. of P in S (3	4	0.76628	0.82442
Ref. + Pos. of P in W (5 cl.)	6	0.74796	0.81750
Ref. + Pos. of P in PHR (5 cl.)	6	0.74902	0.81959
7- Ref. + Pos. of S in W (4 cl.)	3	0.76498	0.81971
Ref. + Pos. of S in PHR (6 cl.)	3	0.76038	0.82126
8-Ref. + Pos. of W in PHR (3 cl.)	4	0.75764	0.82037
9- Ref. + Number of P in S	5	0.75433	0.82085
Ref. + Number of P in W	6	0.74908	0.81762
Ref. + Number of P in PHR	5	0.75445	0.82342
10- Ref. + Number of S in W	5	0.75555	0.82207
Ref. + Number of S in PHR	5	0.75381	0.82344
11- Ref. + Number of W in P	3	0.76895	0.82597

Table 1. Summary of results with 50% of the database for training and 50% for testing (P=phoneme, S=syllable, W=word, PHR=phrase).

2.4.7 Modeling of the duration

At this stage, we decided to dedicate 75% of the database for training and 25% for testing. To model the duration, the first decision is if it should be normalized. We found that it is better to normalize it by the duration of the phrase; this way the system is less affected by changes in speed in the database recordings. After that, we can use:

- The duration itself.
- The logarithm of duration (bad results).
- Find the average duration for each phoneme and model the standard deviation.
- The logarithm of the standard deviation.

The results are shown in Table 2. Although the differences are not significant, the best behavior in general corresponds to the last option, which we will use from now on. It is the first experiment that showed an improvement using two layers.

Experiment	Neu-rons	Rel RMS – Train	Rel RMS – Test
Ref. (Duration not normalized)	5	0.78128	0.81588
Duration normalized	8	0.75740	0.79793
Ref., standard deviation	8	0.75629	0.79236
Ref., logarithm of strd. deviation	8	0.75693	0.79504
	12-6	0.74956	0.79334

Table 2. Different ways to code the duration.

2.5 Putting everything together

In Table 3 we can see the summary of results using the parameters together. Numbers refer to items in Table 1. First experiments are windowing experiments that show an improvement and have been used too.

Experiment	Neu-rons	Rel RMS – Train	Rel RMS – Test
Reference	12-6	0.74956	0.79334
12- Ref + window of 5 phonemes	8	0.72431	0.78068
Ref + window of 7 phonemes	8	0.70785	0.78316
13- Ref + window of 5 in stress	8	0.74605	0.78582
12+1+4	8	0.72209	0.77378
12+1+4+6+7+8	8	0.71798	0.77936
12+1+4+6+7+8+9+10+11	8	0.70669	0.77674
12+1+4+6+7+8+9+10+11+5	8	0.70523	0.77916
12+1+4+6+7+8+9+10+11+5+2+3	8	0.70750	0.78048
12+1+4+6+7+8+9+10+11+5+14	8	0.70265	**0.76428**

Table 3. Putting everything together.

The main conclusion is that after a certain point of saturation, the results only improve significantly for the training test. The last experiment includes a new parameter (14): beginning of phrase (till first stress) and ending of phrase (from last stress).

In our best experiment the average absolute error is 229 samples, equivalent to 14.3 ms, which is really close to the maximum accuracy of the segmentation of our database (10 ms steps).

2.6 Comparison to rule-based system

Our previous rule-based system had a relative RMS equal to 0.90547, which is clearly worse than our best result.

We have observed that the results provided by the net are very accurate in the trend (means that the training is correct), but sometimes cannot reach the peak values, which strongly affects to the average results.

3. WORKING ENVIRONMENT

We will not give a detailed description, but we have developed a complete environment in Matlab oriented to:

- Prepare the inputs to the NN. In the first stage, using just the labeling of the phrase, we decide the absolute values for all parameters considered. Then, the system prepares the values of the inputs according to the design specified in a Matlab model.
- This design is easy to prepare, as we have black boxes for all the codifications mentioned in this paper. You just have to select each parameter and its corresponding codification using a box.
- Train the neural network.
- Evaluate the results. We generate detailed reports and charts with the comparison between the durations obtained and the target values.

4. CONCLUSIONS

Compared to our previous rule-based system:

- We have a system that can be easily adapted to specific contexts and/or new databases. We are going to apply the same system to a system with a restricted prosody. We expect a major improvement in it, as the database will be more homogeneous.
- The results are definitely better.

It is difficult to find the optimum topology of the network. It is better to begin with a low number of neurons and increase it step by step. The same applies to the inclusion of parameters: it is better to decide their best coding in small networks. We have found a good compromise between network topology and parameters considered, with good results that are stable.

5. REFERENCES

[1] De Tournemire, S., “Identification and automatic generation of prosodic contours for a text-to-speech synthesis system in French”, Eurospeech 97.

[2] Masters, T., “Practical neural networks recipes in C++”, Academic Press Inc.

[3] Möbius, B., J.P.H. van Santen, “Modeling segmental duration in German text-to-speech synthesis”, ICSLP 96.

[4] Morlec, Y., G. Bailly, V. Aubergé, “Synthesising attitudes with global rhythmic and intonation contours”, Eurospeech 97.

[5] Vallejo, J.A., "Improvement of the fundamental frequency in text-to-speech conversion". Doctoral Thesis, ETSIT, Madrid, UPM, 1998.

[6] van Santen, J.P.H., “Prosodic modeling in text-to-speech synthesis”, Eurospeech 97.

[7] van Santen, J.P.H., “Assignment of segmental duration in text-to-speech synthesis”, Computer Speech and Language (1994)8, pp. 95-128.