Soft Input Feature Selection within Neural Prosody Generation

Çağlayan Erdem[(1,2)] *& Hans Georg Zimmermann*[(2)]

(1) Dresden University of Technology, D-01062 Dresden, Germany
(2) Siemens AG , Corporate Technology, D- 81730 Munich, Germany
{Caglayan.Erdem;Georg.Zimmermann}@mchp.siemens.de

Abstract

The analysis and selection of input features within machine learning techniques is an important problem if a new system has to be established or the system has to be trained for a new task. Within a Text-to-Speech (TTS) application this task has to be handled while adapting a system to a new language or a new speaker. In this paper a parameterized data-driven weight decay [1] is presented and applied in order to systematically analyze phonetic and linguistic input features of a neural network (NN). The NN models an acoustic prosody generation module of our TTS system *Papageno*. The original NN is enhanced by an additional preprocessing unit. The input features are propagated by a diagonal matrix to a preprocessing cluster. This diagonal matrix is the only one which utilizes the weight decay technique. So the elements of this matrix describe a weighing of the input features. The application resulted in an evaluation of input parameters and a strong reduction of input features without performance loss. Within our f0-contour generation module the squared error of the NN is remarkably reduced by 13%.

1. Introduction

Within data-driven prosody generation modeling by NNs was established first by [2] and became state-of-the-art technique in the following [3], [4]. The commonly used NN architecture is the multi-layer-perceptron with a direct recurrence [2] or without [3] [4]. The advantage of neuro modeling is that it allows fast and easy adaptation to new languages, speakers and speaking styles [3]. Our TTS system *Papageno* has a language and speaker independent core with loadable knowledge bases like lexica and NNs for special tasks. The f0-contour generation module is such a loadable one [4]. Building up a NN for such a task for the first time or during the adaptation to a new language or a new speaker there is a dilemma of using as many inputs as available and at the same time avoiding high input dimensions of a NN. It is a known problem that NNs with a high input dimension tend to over-fitting ([5]). This results in lower generalization abilities. This problem was overcome using expert knowledge (knowledge about the importance of the input parameters) or time consuming heuristic approaches [6] (testing different input parameter constellations to get the best performance). But with the proposed parametric weight decay method it is possible to overcome the difficulties by a data-driven technique that means fast adaptation without language expertise. This parametric weight decay systematically analyses the input vector of a NN. In an additional preprocessing unit of the original NN the input parameters are propagated by a diagonal matrix to a preprocessing cluster. The weight decay concept is applied only to these diagonal elements. These elements represent a weighing of the according input, which then allows an evaluation of input parameters. This paper is organized as follows. Sec. 2 presents the standard weight decay method and the parametric weight decay. In Sec. 3 the application of the parametric weight decay within the f0-generation module is presented. Experiments and results are shown in Sec. 4.

2. Weight Decay

The weight decay concept helps training NN models with a reduced degree of freedom by adding a penalty term to the error function (see eq. (1) and (4)). The first term $E_0(w)$ is the original error function and the penalty is given by $\frac{\lambda}{2}\sum_{w_i} w_i^2$, where λ denotes a factor that scales the penalty and w_i denotes the weights of the NN the penalty term is applied on. During minimization of eq. (1)

$$E(w^j) = E_0(w^j) + \frac{\lambda}{2}\sum_{w_i^j}(w_i^j)^2 \longrightarrow \min_{w_i} \qquad (1)$$

small values for the weights w_i are preferred as high values lead to big penalties. So the penalty term encourages smoother NN mappings [1]. In eq. (1) an error function is depicted known as the standard weight decay. j denotes the number of the training epochs of the NN. In eq. (2)

$$w^{j+1} = w^j - \eta\frac{\partial E(w^j)}{\partial w^j} \qquad (2)$$

the according weight adaptation is given with η as a factor that controls the step size during NN training. Substituting $\frac{\partial E(w^j)}{\partial w^j}$ by its partial derivative leads to eq. (3)

$$w^{j+1} = w^j - \eta\lambda w^j - \eta\frac{\partial E_0(w^j)}{\partial w^j} \qquad (3)$$

The weight vector w^{j+1} for the next epoch is computed as the difference of the prior weight vector w_j and the partial derivative of the error function (back propagation). This standard weight decay was also applied in [7] [8]. In eq. (4)

$$E(w^j) = E_0(w^j) + \frac{\lambda}{p}\sum_{w_i^j}|w_i^j|^p \longrightarrow \min_{w_i}, 0 < p \leq 1 \qquad (4)$$

a modified penalty term with its weight adaptation in eq. (5)

$$w^{j+1} = w^j - \lambda\eta sign(w^j)|w^j|^{p-1} - \eta\frac{\partial E_0(w^j)}{\partial w^j} \qquad (5)$$

is shown. This weight decay will be called p-wd according to the power p term in eq. (4). It is utilized within the following experiments. There are different ways to realize the weight decay within a NN training, as one might apply the penalty term to all weights within the NN or to special connection areas. In Figure 2 a sophisticated realization is depicted. The input vector x is propagated by the diagonal matrix $w^{diag} \in \mathrm{R}^{l\times l}$ to the preprocessing layer, which has a $tanh$-activation function (see [9]). This diagonal matrix w^{diag} is the only connection which utilizes the penalty term $\frac{\lambda}{p}\sum_{w_i}|w_i|^p$.

Due to this order the vector element x_i' is given by $x_i' = x_i w_i^{diag}$. So w_i^{diag} gives a weighing of x_i. w_i is bound to the interval [0,1]. After this preprocessing unit the weighed inputs are propagated to the original NN with n hidden neurons and m outputs. The original NN will be explained in detail later in Sec. 3. It is important to initialize w^{diag} with equal values(e.g. $w_i^{start,j=0} = 0.5$), which are incremented or decremented according to the weight adaptation in eq. (5) during training, as will be presented later. This type of realization

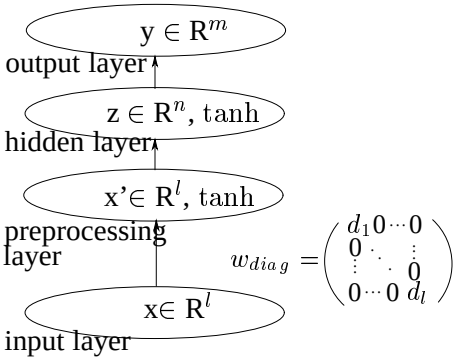

Figure 1: realization of p-wd

of weight decay aims at:

a) Outlier cancellation: the asymptotic behavior of $tanh$ is used in the preprocessing layer as a delimiter. An element of w^{diag} getting to big is limited to an interval [-1;1] at the output side of the preprocessing cluster. Like that inputs are avoided to dominate the mapping. All inputs are limited to the interval [-1;1].

b) Soft input pruning: elements of w^{diag} being pushed to zero are considered to have no significant influence on the NN model. This means that the corresponding input contains no important information for this task with the used database. Inputs being close to zero might be removed afterwards by introducing a threshold and omitting inputs with values in w^{diag} below that threshold. So we obtain a soft input pruning, as there is no ultimate decision made during training. Unimportant inputs are faded out.

c) Separation into subtasks: The original task and the input feature analysis are solved in a parallel manner. The feature analysis does not need a further training phase.

Dealing with highly correlated input features the application of p-wd should be preferred, as it helps to select one of the highly correlated features in contrast to standard wd. wd does not select one of the highly correlated inputs.

This different selection property of p-wd is exemplified using a NN with two highly correlated inputs. This can be formulated in eq. (6).

$$x' = tanh(w_1 x_1 + w_2 x_2) \tag{6}$$

As x_1 and x_2 are highly correlated eq. (6) is equal to eq. (7)

$$x' = tanh((w_1 + w_2) x_1 + 0 x_2) \tag{7}$$

The penalty terms for the standard weight decay are given by eq. (8) (left side of the in-equation according to eq. (6) and the right side according to eq. (7))

$$\frac{\lambda}{2} \sum_w w^2 \quad : \qquad w_1^2 + w_2^2 \leq (w_1 + w_2)^2 + 0^2 \tag{8}$$

The proof of the validity of in-equation 8 is given by the following: During the minimization of the error function in eq. (1) the penalty term ($P(w)$) has to be minimized with the constraint $w_1 + w_2 = a$. This constrained minimization is solved using the Lagrange multiplier method (see eqs. (9) ... (14)).

$$P(w) = \frac{1}{2} w_1^2 + \frac{1}{2} w_2^2 \longrightarrow \min_{(w_1 + w_2 = a)} \tag{9}$$

$$L(\lambda) = \frac{1}{2} w_1^2 + \frac{1}{2} w_2^2 - \lambda(a - w_1 - w_2) \tag{10}$$

$$\frac{\partial L(\lambda)}{\partial w_1} = w_1 - \lambda = 0 \tag{11}$$

$$\frac{\partial L(\lambda)}{\partial w_2} = w_2 - \lambda = 0 \tag{12}$$

$$a = w_1 + w_2 \tag{13}$$

$$\frac{1}{2} a = w_1 = w_2 \tag{14}$$

The result of the Lagrange multiplier method is: both weights have the same sign. Therefore the validity of in-equation 8 is given. As $P(w)$ from eq. (6) is smaller than $P(w)$ from eq. (7) it is obvious that none of the highly correlated inputs are selected by the standard penalty $\frac{\lambda}{2} \sum_w w^2$. So we don't obtain a minimized input feature set.

The in-equation for p-wd is given by eq. (15).

$$\frac{\lambda}{p} \sum_w |w|^p \quad : \qquad |w_1|^p + |w_2|^p \geq |w_1 + w_2|^p + 0^p \tag{15}$$

The proof of the validity is analog to ineq. (8). Through the p-wd penalty function $\frac{\lambda}{p} \sum_w |w|^p$ we achieve a minimized input feature set as the right side of ineq.(15) is smaller than the left side. By the experiments described in the following this behavior of p-wd has been observed.

3. Application

The p-wd method presented above has been applied on the f0-contour generation module which will be explained now. The utilized NN has to map input parameters to an appropriate f0-contour. Such a f0-contour on syllable level is depicted by the solid line in Fig. 3. These contours are parameterized (dashed line in Fig. 3) on syllable level by four values: f0-maximum (p1), f0-maximum-position (p2), f0 at syllable start (p3), and f0 at syllable end (p4). For parameterization a maximum based description is used [10], which mainly says that f0-contours on syllable level for non-tonal languages can be described by a rising on the first part and a falling on the second part of the syllable. The mentioned parameters p1, p2, p3, and p4 are the outputs $y_1, y_2, y_3,$ and y_4 of the NN (see Fig. 3). So the dimension m of the output cluster is $m = 4$. Fig. 3 is identic with the upper part of Fig. 2. f0-contours are known to be influenced by longterm features as the sentence type and breath and by local stress intention. The input parameters must contain information with these local and global characteristics. For a good mapping it is also important to provide contextual information of the syllable. Due to computation reasons the context window length was choosen to be seven to the left (past) and seven to the right (future) of the syllable with the exception of the linguistic categories. The following input features are presented to the NN, which have to be evaluated regarding the importance and the necessary contextual width.

a) phonetic information: The phonetic structure of a syllable to be processed is coded here. The vowel is presented as

an one-out-of-n coded input using the German Sampa phoneme set. Neighboring phonemes of the vowel are assigned to four classes (plosive, fricatives, nasal and liquids) also as an one-out-of-n coded input in a symmetric context window of four phonemes.

b) positional information: Continuous positional information gives time distances of the syllable and its vowel. Discrete information denotes whether this syllable is an initial, medial or final one within the sentence, the phrase, and the word.

c) stress and break information: Flags (one-out-of-n coded input) denote the stress type of a syllable: unstressable, stressable, word stress, phrase second stress, and phrase main stress. Four flags give break information: phrase break, sentence break, question break, and B2 break. B2 breaks give smaller substructeres within phrase breaks.

d) linguistic categories: The used linguistic category set consists of 14 tags, which are one-out-of-n coded and presented in a context of 3 to both sides: numeral, verb, verb particle, pronoune, preposition, noun, particle, determiner, conjunction, adverb, adjective, particle+determiner, interjection, punctuation.

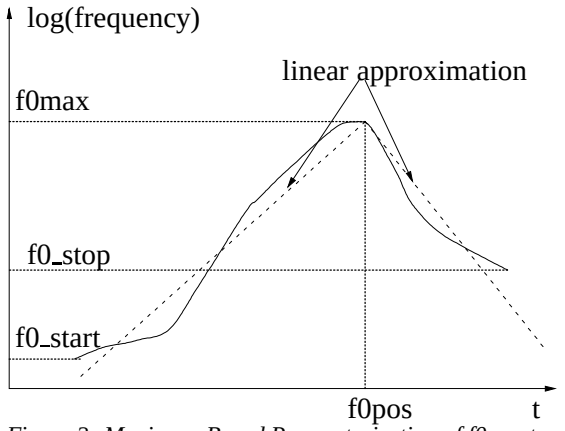

Figure 2: Maximum Based Parameterization of f0-contours

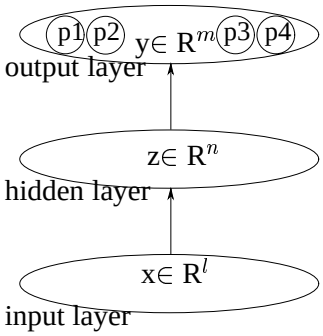

Figure 3: original NN for f0-contour generation

By this input constellation the input dimension of x according to Fig. 3 and Fig. 2 is 567. The p-wd technique is applied to recordings of three hours of a german news speaker reading news from *Frankfurter Allgemeine Zeitung*. The patterns for training (80%) and testing (20%) are separated. A validation set of (20%) is selected randomly from the training set. In the following subsection the parameter of p-wd is optimized by experiments. This NN is then used to analyze the inputs.

4. Experiments & Results

In a first series of experiments NNs with varying parameters p from 0.1 to 2.0 in steps of 0.1 were trained. The optimum parameter was found to be $p = 0.6$ (see Fig. 4). 3537 training epochs were used to train this NN. It was also observed

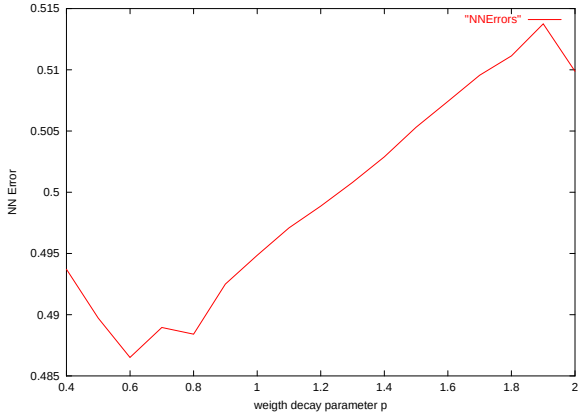

Figure 4: NN squared errors on validation set

that smaller p needed more iterations. Without weight decay 80 training epochs were necessary. In Fig. 4 a three dimensional plot depicts the behavior of w^{diag}. In this plotted range (w^{diag} from 100 to 200 and parameter p from 0.6 to 2.0) it is possible to see how special inputs are evaluated to have a stronger effect within this NN mapping and how other inputs are evaluated to have less impact almost independent of p.

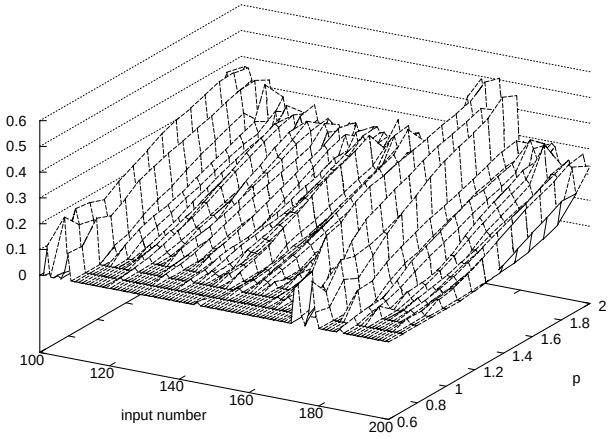

Figure 5: 3-dim. plot of w^{diag} values

In Fig. 4 the elements of w^{diag} are plotted for parameter $p = 0.6$. As can be seen most of the elements are put to zero. A reduction of the NN error on the validation and test set by 13% was observed (see Table 3). The following gives an analysis of the data-driven evaluation of the inputs. The symmetric context window length of seven units was reduced to five units to the left and four to the right.

a) Phonetic information: Only 8 vowels from original 18 vowels were valued to have an impact: diphthongs: (aI and aU), long vowels: (a:, e:), short vowels: (a, I, O, U). Only direct phonetic neighbors of the vowels have a non-zero value. At the left side classes are rated: nasal (=0.0962), liquid (=0.0505), and fricative (=0.0226) different to the right side: liquid (=0.1021) and plosive (=0.0338).

b) Positional information: Continuous information has an unsymmetric context window length (five to the left and three to the right). The vowel position and duration of the central syllable and its direct neighbors is rated non-zero. The syllable position with in the sentence and the phrase was rated with a context of five to the left and three to the right to be important. Discrete positional information flags denoting the syllable to be in the beginning of a phrase, the left neighbor to be a medial syllable in a word, and the left neighbor to be in the final word are rated non zero.

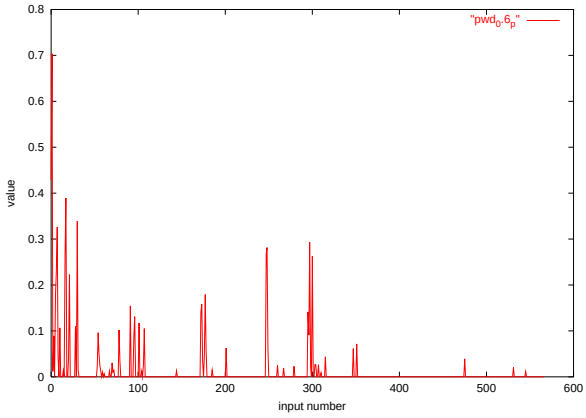

Figure 6: diagonal elements of w^{diag}

c) Stress and break information: this group of information was found to also have an unsymmetric context window length with a left window of two units and a right window of four units. The most important information was rated: main stress within phrase (=0.3896), B2 (=0.3391) , secondary stress within a phrase (=0.2683)

d) categories: only one category was rated non-zero: the participle from the center to the right with window length of two: categ7 (=0.0392), r1categ7 (=0.0204), and r2categ7 (=0.0117)

The different information types have different contextual lenghts as shown in table 1. The dimensionality was strongly

contextual length	right	left
positional information	5	3
stress and breaks	2	4
categories inform.	0	2
phonetic inform.	3	2

Table 1: context window length

reduced from initially 567 to 67 inputs ($p = 0.6$). Tab. 2 gives the reduction rates for parameter p.

NN	$p = 1.0$	$p = 0.8$	$p = 0.6$	$p = 0.4$
reduction	16%	51%	78%	86%

Table 2: reduction of inputs by weight decay parameter p

In Table 3 the squared errors of the trained NN are listed. Optimizing the squared errors does not necessarily result in a naturally sounding voice. Perception is a highly complex process not necessarily modeled appropriately by isolated squared error distances. Therefore informal listening tests were performed. But unlike the squared error tests these tests did not verify an improvement of the NN performance.

NN	normal NN	standard wd	$(p = 0.6)$
validation set	0.554	0.509	0.486
testing set	0.551	0.512	0.488

Table 3: NN squared errors

5. Conclusions

A specialized weight decay method is presented in this paper. Which offers a soft input feature selection even within highly correlated data input vector elements. Its application within the f0-generation task helped to analyze and optimize the input features without linguistic expert knowledge.

Important phonetic and linguistic input features are determined. The less important features are faded out during training and removed afterwards. The necessary contextual widths for each input information type are determined. By this data-driven analysis the initial dimension of the input space is reduced by 78% with a reduction of the squared error by 13% in the test set.

This method gives systematical support for the selection of input features and is well suited to reduce the dimension of input vectors without having performance losses.

6. References

[1] Christopher Bishop. Neural Networks for Pattern Recognition, *Oxford University Press, 1995*

[2] Christof Traber, 1992. F0 generation with a database of natural F0 patterns and with a neural network. In *Talking Machines: Theories, Models and Designs* G. Bailly and C. Benoit, (eds.). Elsevier, North-Holland, 287-304.

[3] Ralf Haury, Martin Holzapfel, 1998. Optimisation of a Neural Network for Pitch Contour Generation, *ICASSP*, Seattle

[4] Çağlayan Erdem, Martin Holzapfel, Rüdiger Hoffmann, 2000. Natural f0-contours with a new Neural-Network-hybrid-approach. *ICSLP*, Beijing

[5] Lutz Prechelt, 1998. Early Stopping - But When?. In *Neural Networks: Tricks of the Trade.* G. B. Orr and K.-R. Müller, (eds.)., Springer Verlag, Berlin, 55-69

[6] Gerit P. Sonntag, Thomas Portele, and Barbara Heuft, 1997. Prosody generation with a neural network: Weighing the importance of input parameters. *ICASSP*

[7] Achim F. Müller, Jianhua Tao, and Rüdiger Hoffmann, 2000. Data-driven importance analysis of linguistic and phonetic information *ICSLP* Beijing

[8] Horst-Udo Hain and Hans Georg Zimmermann, 2001. A Multi-lingual System for the Determination of Phonetic Word Stress Using Soft Feature Selection by Neural Networks. *4th ISCA Workshop on Speech Synthesis*, Perthshire, Scotland

[9] Ralph Neuneier and Hans Georg Zimmermann, 1998. How to train neural networks. In *Neural Networks: Tricks of the Trade.* G. B. Orr and K.-R. Müller, (eds.)., Springer Verlag, Berlin, 373-423.

[10] Heuft, B. et alt.,1995. Parametric Description of F0-Contours in a Prosodic Database. *Proc. ICPHS* Vol. 2, 378-381.