Annales UMCS Informatica AI XII, 2 (2012) 39–48
DOI: 10.2478/v10065-012-0014-2

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI
http://www.annales.umcs.lublin.pl/

# Automatic disordered sound repetition recognition in continuous speech using CWT and kohonen network

Ireneusz Codello[1]*, Wiesława Kuniszyk–Jóźkowiak[1]†, Elżbieta Smołka[1]‡, Adam Kobus[1]§

[1] *Institute of Computer Science, Maria Curie-Skłodowska University*
*pl. M. Curie-Skłodowskiej 1, 20-036 Lublin, Poland*

**Abstract** – Automatic disorders recognition in speech can be very helpful for a therapist while monitoring therapy progress of patients with disordered speech. This article is focused on sound repetitions. The signal is analyzed using Continuous Wavelet Transform with 16 bark scales. Using the silence finding algorithm, only speech fragments are automatically found and cut. Each cut fragment is converted into a fixed-length vector and passed into the Kohonen network. Finally, the Kohonen winning neuron result is put on the 3-layer perceptron. Most of the analysis was performed and the results were obtained using the authors' program WaveBlaster. We use the STATISTICA package for finding the best perceptron which was then imported back into WaveBlaster and used for automatic blockades finding. The problem presented in this article is a part of our research work aimed at creating an automatic disordered speech recognition system.

## 1 Introduction

Speech recognition is a highly important branch of computer science nowadays – oral communication with a computer can be helpful in real-time document writing, language translation or simply in using a computer. Therefore the issue has been analyzed for many years by the researchers which resulted in creating many algorithms, such as the Fourier transform, Linear Prediction, spectral analysis. Disorders recognition in speech is quite a similar issue – one attempt to find where speech is not fluent instead of trying

*irek.codello@gmail.com
†jozkowiak@gmail.com
‡esmolka@tytan.umcs.lublin.pl
§kobus.adam@gmail.com

to understand the speech, therefore the same algorithms can be used. Automatically generated statistics of disorders can be used as a support for therapists in their attempts at estimating therapy progress.

Several methods for the disordered speech detection have been used by researchers for disordered speech recognition, like: Fourier Transform, third octave filters, fuzzy logic [**1**], Hidden Markov Models, MFCC coefficients [**2**], Linear Prediction [**3**] or Kohonen networks [**4**]. In this paper a relatively new algorithm is used – Continuous Wavelet Transform (CWT) ([**5, 6, 7**]) as - by using it - the most suitable scales (frequencies) can be chosen. Fourier transform and Linear Prediction [**8**] are not so flexible – we have to choose if we want to have more precise time scale (small window) and more precise frequencies or the opposite - for the whole spectrogram. In CWT such a decision can be made for each scale separately. The bark scales set was taken, which is, besides the Mel scales and the ERB scales, considered as a perceptually based approach[**9**]. Using only the speech finding algorithm, the utterance fragments were found and cut (automatically). Each cut fragment was converted into the fixed-length window (which contained several vectors eq. 5) and passed into the Kohonen network which received the 3D data and produced the 2D data (see Fig. 3). Such a dimensionally reduced signal was passed to a 3-layer perceptron with a mark: containing a blockade or not.

Perceptron learning was performed by the STATISTICA's 'Neural Network' package and its tool – Intelligent Problem Solver. Once found, the best network was imported back again into WaveBlaster and then it was used for the automatic disorders finding. Two–result statistics were presented: learning statistics of the best perceptrons from the STATISTICA package and recognition statistics obtained by WaveBlaster using these perceptrons.

## 2 Input signal processing by CWT

### 2.1 Mother wavelet

Mother wavelet is the heart of the Continuous Wavelet Transform:

$$CWT_{a,b} = \sum_t x(t) \cdot \psi_{a,b}(t), \qquad \text{where} \qquad \psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right) \tag{1}$$

where $x(t)$ – input signal, $\psi_{a,b}(t)$ – wavelet family, $\psi(t)$ – mother wavelet, $a$ – scale (multiplicity of mother wavelet), $b$ – offset in time. The Morlet wavelet represented by equation (2) was used ([**10**]):

$$\psi(t) = e^{-t^2/2} \cdot \cos(2\pi \cdot 20 \cdot t) \tag{2}$$

which has the center frequency $F_C = 20$Hz. Mother wavelets have one significant feature: length of the wavelet is connected with $F_C$ which is a restraint. The Morlet wavelet is different because the length can be chosen and then its $F_C$ can be set by changing the cosines argument.

### 2.2 Scales

For frequencies of scales, a perceptually based approach was assumed – because it is considered to be the closest to the human way of hearing. The Hartmut scales were chosen [**11**]:

$$B = \frac{26.81}{1 + 1960/f} - 0.53, \quad f\text{– freq. in Hz.} \tag{3}$$

The frequency $F_a$ of each wavelet scale $a$ was computed from the equation

$$F_a = F_C F_S / a, \quad F_S\text{– sampling frequency.} \tag{4}$$

Due to the discrete nature of the algorithm, it was not always possible to match scale $a$ with scale $B$ perfectly (Table 1). During the research some Hartmut scales were found as insignificant in the recognition process. Therefore eventually only 16 scales were used.

Table 1. 16 scales a with the corresponding frequencies $f$ and the bark scales $B$.

| *a* [scale] | *f* [Hz] | *B* [bark] |
|---|---|---|
| 57 | 7736 | 20.9 |
| 68 | 6485 | 20.1 |
| 83 | 5313 | 19.1 |
| 100 | 4410 | 18 |
| 119 | 3705 | 17 |
| 140 | 3150 | 16 |
| 163 | 2705 | 15 |
| 190 | 2321 | 14 |

| *a* [scale] | *f* [Hz] | *B* [bark] |
|---|---|---|
| 220 | 2004 | 13 |
| 256 | 1722 | 12 |
| 297 | 1484 | 11 |
| 347 | 1270 | 10 |
| 408 | 1080 | 9 |
| 479 | 920 | 8 |
| 572 | 770 | 7 |
| 700 | 630 | 6 |

### 2.3 Smoothing scales

Because the CWT values are similarity coefficients between the signal and wavelet, their sign are therefore irrelevant, in all computations, the following modules are taken – $|CWT_{a,b}|$. We went one step further and the $|CWT_{a,b}|$ was smoothed by creating a contour (see Fig. 1) because of its good recognition ratio influence [**12**].

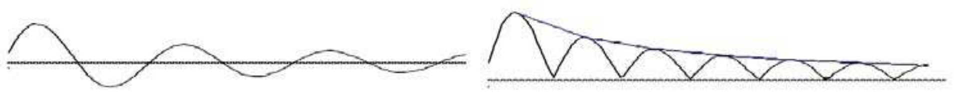


Fig. 1. Left: Cross-section of one $CWT_{a,b}$ scale. Right: Cross-section of one $|CWT_{a,b}|$ scale and its contour (smoothed version).

### 2.4 Windowing

Thus the spectrogram consists of 16 smoothed bark scales vectors. Then the spectrogram was cut into 23.2ms frames (512 samples when $F_S$=22050Hz), with a 100% frame offset. Because each scale has its own offset – one window of fixed width (e.g. 512 samples) will contain a different number of CWT values (CWT similarity coefficients) in each scale (see Fig. 3), therefore the CWT arithmetic mean of each scale value was taken.

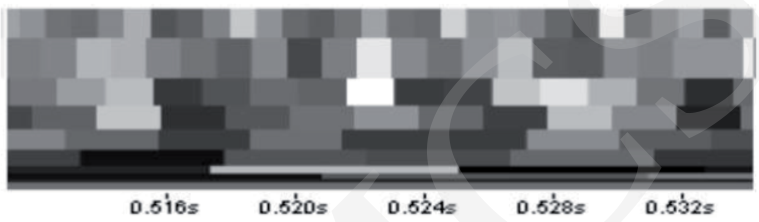


Fig. 2. One CWT window (512 samples when $F_S$=22050Hz).

From one $i$–th window the vector $V$ of the form presented in eq. (5) was obtained. Such consecutive vectors were then passed into the Kohonen network.

$$\vec{V} = \{mean\left(|CWT_{57,i}|\right), mean\left(|CWT_{68,i}|\right), \ldots, \\ mean\left(|CWT_{572,i}|\right), mean\left(|CWT_{700,i}|\right)\} \tag{5}$$

## 3 Modified kohonen network algorithm

The Kohonen network ([**13, 14, 15, 16, 17, 18**]) (or "self-organizing map" or SOM, for short) was developed by Teuvo Kohonen. The basic idea behind the Kohonen network is to establish a structure of interconnected processing units ("neurons") which compete for the signal. While the structure of the map may be quite arbitrary, rectangular maps were used in the research.

Let us assume that:

- Kohonen network has $K$ neurons
- $n$ is the dimension of each input vector $X$
- each element $x_i \in X$ is connected to all $K$ neurons, so we have $K \times n$ connections. Each connection is represented by its weight $w_{ij}$, $i = 1, \ldots, n$, $j = 1, \ldots, K$ which is adjusted during the training.

The Kohonen neurons were numbered by rows from the top to the bottom

| | | | | |
|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 | 9 |
| 10 | 11 | 12 | 13 | 14 |

For every 2D CWT vector (see eq. (5)) one winning neuron is obtained. Therefore the Kohonen network is used to convert the 3–dimension CWT spectrogram (which consists of 2D CWT vectors situated one next to another) into the 2–dimension winning neuron contour as depicted in Fig. 3 ([**4, 19**]). Such reduction of data, from 3D into 2D, which is later passed on to MLP, occurred to have a positive impact on the non-fluencies recognition ratio ([**4, 19**]) (the whole 3D spectrogram seems to be too large for MLP to find general features).

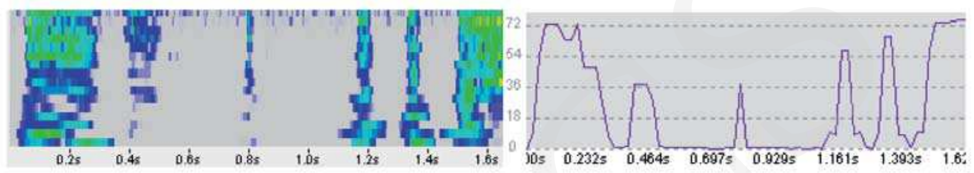


Fig. 3. Converting 3D CWT (Left picture. $Y$ axis: the bark scale, $X$ axis: the time) into the 2D Kohonen winning neuron contour (Right picture. $Y$ axis: winning neuron, $X$ axis: the time). In this example the Kohonen network was of the size $8 \times 9$ giving 72 neurons.

The standard training algorithm [**16, 18**] was used with one modification – i.e. $0^{\text{th}}$ neuron clearing [**20**].

## 4 Automatic disordered sound repetitions recognition

### 4.1 Input data

The Polish speech recordings of 9 stuttering persons were taken of the summary length equal to 9 min 44 s divided into 3 files: allblknn1, allblknn2, allblknn3 containing 294 disordered repetitions of the sounds: b,d,g,k,n,o,p,t. The statistics were the following:

Table 2. Disordered sound repetition fragments counts.

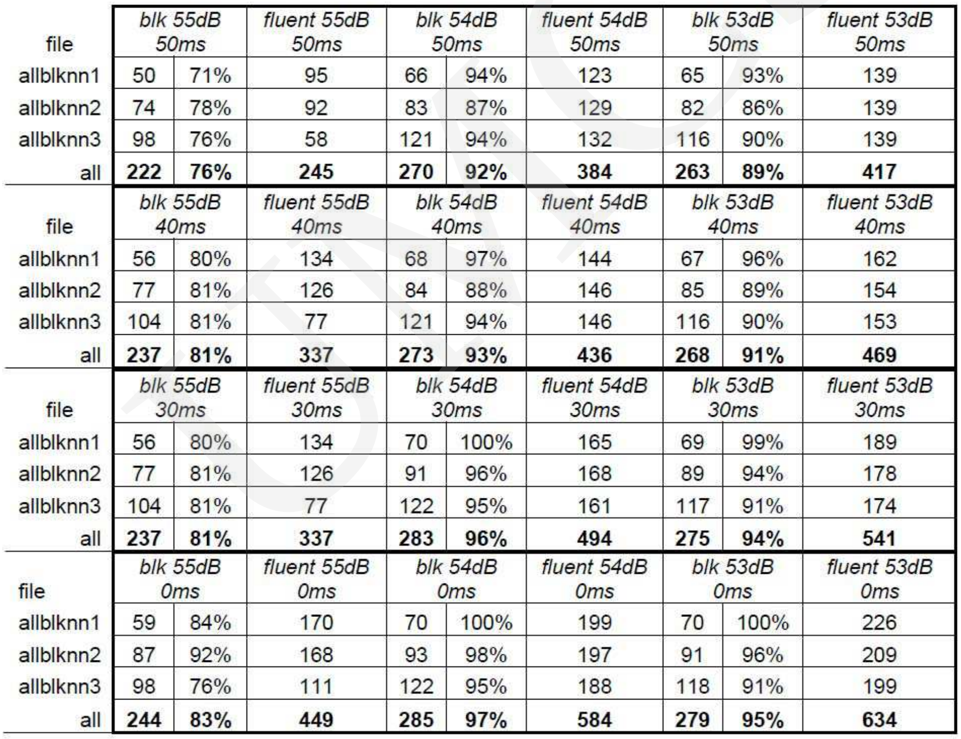
| *file* | *b* | *d* | *g* | *k* | *n* | *o* | *p* | *t* | ***sum*** |
|---|---|---|---|---|---|---|---|---|---|
| allblknn1 | 3 | | 3 | 33 | | | 20 | 11 | **70** |
| allblknn2 | 1 | 3 | | 15 | | | 17 | 59 | **95** |
| allblknn3 | 19 | 3 | 7 | 34 | 2 | 1 | 29 | 34 | **129** |
| **all** | **23** | **6** | **10** | **82** | **2** | **1** | **66** | **104** | **294** |

### 4.2 Automatic blockades cutting

Input files were automatically divided into words by a simple algorithm. We divided the CWT scalogram into 22ms windows with 11ms offset. Each window was marked as speech if it contained at least one value above the **threshold**: -53dB, -54dB or

-55dB (maximum CWT value was assigned as 0dB). Because we were looking only for disordered blockades, which are always short, words longer than 200ms were removed. Moreover, we observed that cutting algorithm was so sensitive, that it found silence in fluent words and divided them into pieces. Therefore we added the second parameter – **distance**: 50ms, 40ms, 30ms, 0ms. If two words were closer than the distance, then they were treated as one longer word and removed. Based on these two parameters, we created blockades cutting statistics containing a number of correctly cut blockades and a number of fluent words:

Table 3. Blockades cutting statistics (number of words) for the threshold and distance parameters.



| file | *blk 55dB 50ms* | | *fluent 55dB 50ms* | *blk 54dB 50ms* | | *fluent 54dB 50ms* | *blk 53dB 50ms* | | *fluent 53dB 50ms* |
|---|---|---|---|---|---|---|---|---|---|
| allblknn1 | 50 | 71% | 95 | 66 | 94% | 123 | 65 | 93% | 139 |
| allblknn2 | 74 | 78% | 92 | 83 | 87% | 129 | 82 | 86% | 139 |
| allblknn3 | 98 | 76% | 58 | 121 | 94% | 132 | 116 | 90% | 139 |
| all | **222** | **76%** | **245** | **270** | **92%** | **384** | **263** | **89%** | **417** |
| file | *blk 55dB 40ms* | | *fluent 55dB 40ms* | *blk 54dB 40ms* | | *fluent 54dB 40ms* | *blk 53dB 40ms* | | *fluent 53dB 40ms* |
| allblknn1 | 56 | 80% | 134 | 68 | 97% | 144 | 67 | 96% | 162 |
| allblknn2 | 77 | 81% | 126 | 84 | 88% | 146 | 85 | 89% | 154 |
| allblknn3 | 104 | 81% | 77 | 121 | 94% | 146 | 116 | 90% | 153 |
| all | **237** | **81%** | **337** | **273** | **93%** | **436** | **268** | **91%** | **469** |
| file | *blk 55dB 30ms* | | *fluent 55dB 30ms* | *blk 54dB 30ms* | | *fluent 54dB 30ms* | *blk 53dB 30ms* | | *fluent 53dB 30ms* |
| allblknn1 | 56 | 80% | 134 | 70 | 100% | 165 | 69 | 99% | 189 |
| allblknn2 | 77 | 81% | 126 | 91 | 96% | 168 | 89 | 94% | 178 |
| allblknn3 | 104 | 81% | 77 | 122 | 95% | 161 | 117 | 91% | 174 |
| all | **237** | **81%** | **337** | **283** | **96%** | **494** | **275** | **94%** | **541** |
| file | *blk 55dB 0ms* | | *fluent 55dB 0ms* | *blk 54dB 0ms* | | *fluent 54dB 0ms* | *blk 53dB 0ms* | | *fluent 53dB 0ms* |
| allblknn1 | 59 | 84% | 170 | 70 | 100% | 199 | 70 | 100% | 226 |
| allblknn2 | 87 | 92% | 168 | 93 | 98% | 197 | 91 | 96% | 209 |
| allblknn3 | 98 | 76% | 111 | 122 | 95% | 188 | 118 | 91% | 199 |
| all | **244** | **83%** | **449** | **285** | **97%** | **584** | **279** | **95%** | **634** |

Based on these statistics we decided to get only the configurations: 50ms-55dB, 50ms-54dB, 30ms-54dB.

### 4.3 Training algorithm

The procedure of finding sound repetitions in the file was the following:

1. The CWT spectrogram of the continuous speech was computed.
2. The CWT signal was divided into 22ms windows with 50% offset, and only the words that match criteria (see 4.2) were chosen. The distance and threshold parameters

were applied to the algorithm (see 4.2), and the most suitable ones were used: 50ms-55dB, 50ms-54dB, 30ms-54dB.

3. If the speech fragment passed the above verification, it was cut with a surrounding according to the **window length** parameters: 700ms, 1000ms, 1500ms, 2000ms, 2500ms, 3000ms (each window always contained the 500ms prefix, speech fragment and the postfix of variable length so that we would obtain a desired window length).
4. Each window which consisted of 16-element vectors was automatically passed into the Kohonen network. After the training process a winning neuron graph was obtained (Fig 3). The 5x5 Kohonen network was used with the following parameters: 100 epochs, learning coefficient 0.20-0.10, and neighbour distance 2.5-0.5.
5. Each graph was marked as fluent or non-fluent (this information was 'the teacher' in the perceptron learning algorithm).
6. Using STATISCTICA, the perceptron with the best recognition ratio was found. The input vectors were divided randomly by STATISTICA into teaching set (50%), verifying set (25%) and testing set (25%). (Only the allblknn2 and allblknn3 files were passed to the STATISTICA). The best perceptron (see Table 4) was imported back into WaveBlaster and all three files took part in the finding process.

### 4.4 Finding algorithm

1. Steps 1–4 were repeated from the previous paragraph.
2. The obtained Kohonen vector was passed into the perceptron (imported from STATISTICA) and its output was checked.
3. Based on the output the speech fragment was marked as fluent/non-fluent.

## 5 Results

The recognition ratio was calculated with the use of these formulas:

$$sensitivity = \frac{P}{A}; \quad predictability = \frac{P}{P+B} \tag{6}$$

where $P$ is the number of correctly recognized disorders, $A$ is the number of all disorders and $B$ is the number of fluent sections mistakenly recognized as disorders.

## 6 Conclusions

As we can see in Table 4 all perceptrons distinguish blockades really well (97%-100%), even in veriication and testing sets (test vectors do not take part in teaching at all). That is because of speech cutting algorithm – on the perceptron only speech fragments that begin with the utterance were passed, therefore the perceptron does not have to straggle with fragments that have sometimes blockade in the middle and sometimes at the end. Such results would suggest that this method of cutting blockades is very good.

Table 4. Best perceptron recognition ratio in % for allblknn2 and allblkn3 files. STATISTICA randomly divided vectors into learning (50%), verifying (25%) and testing set (25%). In 'net' column we have a number of neurons on each layer. Learning algorithm: BP100 – back propagation with 100 epochs, CG20b – continuous gradients with 20 epochs.

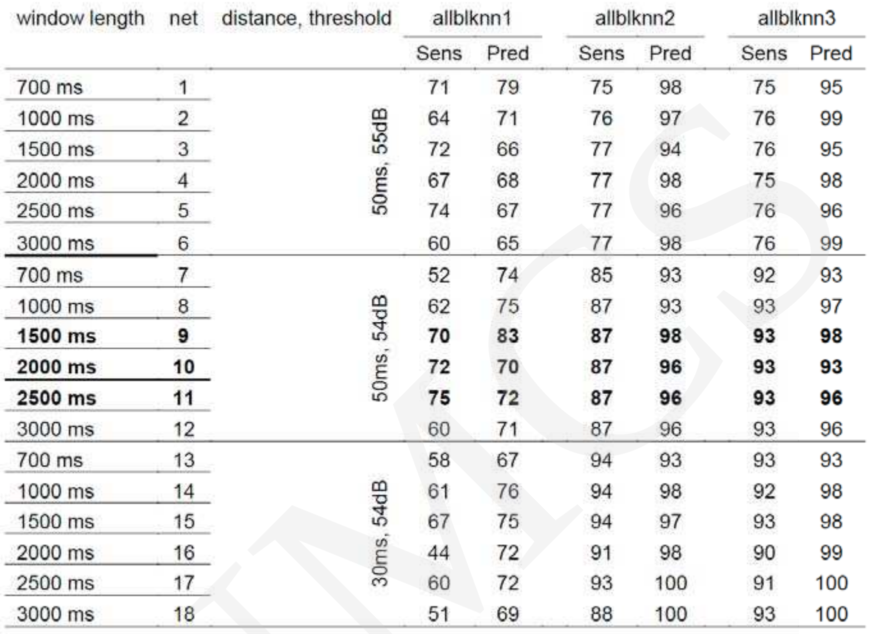
| window length | net | | learning algorithm | | All | | U | | W | | T | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | blk | fluent | blk | fluent | blk | fluent | blk | fluent |
| 700 ms | 1 | 31-130-1 | BP29b | 50ms, 55dB | 98.2 | 97.7 | 99.4 | 99.0 | 97.6 | 96.4 | 96.5 | 96.4 |
| 1000 ms | 2 | 44-91-1 | BP100,CG20b | | 98.8 | 99.2 | 99.6 | 99.9 | 97.9 | 98.8 | 98.1 | 98.0 |
| 1500 ms | 3 | 65-78-1 | BP100,CG37b | | 99.5 | 98.0 | 100.0 | 99.8 | 99.4 | 96.5 | 99.8 | 96.5 |
| 2000 ms | 4 | 87-87-1 | BP100,CG28b | | 99.3 | 99.2 | 100.0 | 100.0 | 98.5 | 98.9 | 98.9 | 98.0 |
| 2500 ms | 5 | 108-74-1 | BP100,CG44b | | 99.7 | 99.3 | 100.0 | 100.0 | 99.2 | 99.4 | 99.7 | 97.9 |
| 3000 ms | 6 | 130-130-1 | BP100,CG15b | | 99.1 | 99.8 | 99.8 | 100.0 | 98.4 | 99.4 | 98.6 | 99.8 |
| 700 ms | 7 | 31-130-1 | BP33b | 50ms, 54dB | 97.1 | 97.7 | 97.9 | 99.0 | 95.5 | 96.2 | 97.0 | 96.5 |
| 1000 ms | 8 | 44-91-1 | BP100,CG20b | | 99.8 | 98.4 | 100.0 | 99.6 | 99.5 | 97.7 | 99.7 | 96.6 |
| 1500 ms | 9 | 65-83-1 | BP100,CG28b | | 99.4 | 99.5 | 100.0 | 100.0 | 98.5 | 98.9 | 99.0 | 99.2 |
| 2000 ms | 10 | 87-74-1 | BP100,CG55b | | 99.8 | 98.6 | 100.0 | 100.0 | 99.6 | 96.7 | 99.7 | 97.7 |
| 2500 ms | 11 | 108-100-1 | BP100,CG42b | | 99.8 | 98.8 | 100.0 | 100.0 | 99.4 | 97.5 | 99.9 | 97.5 |
| 3000 ms | 12 | 130-98-1 | BP100,CG49b | | 99.5 | 99.7 | 100.0 | 100.0 | 99.2 | 99.3 | 99.0 | 99.5 |
| 700 ms | 13 | 31-130-1 | BP14b | 30ms, 54dB | 97.3 | 98.0 | 98.1 | 99.1 | 96.1 | 96.5 | 96.8 | 97.4 |
| 1000 ms | 14 | 44-130-1 | BP30b | | 98.8 | 98.8 | 99.8 | 99.7 | 97.7 | 98.0 | 98.0 | 97.7 |
| 1500 ms | 15 | 65-101-1 | BP100,CG25b | | 99.3 | 99.4 | 100.0 | 100.0 | 98.6 | 98.2 | 98.5 | 99.6 |
| 2000 ms | 16 | 87-99-1 | BP100,CG42b | | 99.9 | 97.7 | 100.0 | 100.0 | 99.8 | 95.1 | 99.7 | 95.9 |
| 2500 ms | 17 | 108-130-1 | BP95b | | 99.9 | 98.2 | 100.0 | 100.0 | 99.8 | 96.5 | 99.8 | 96.4 |
| 3000 ms | 18 | 130-98-1 | BP100,CG56b | | 100.0 | 97.8 | 100.0 | 100.0 | 99.9 | 94.6 | 99.9 | 96.9 |

Unfortunately our speech cutting algorithm has a weakness – it misses some of the blockades and by making it more sensitive, it cuts disproportionately more fluent fragments (see Table 3). Maybe a more complex and more smart algorithm should be used.

As for automatic blockades recognition results in the fluent speech (see Table 5), we need to remember that they can only be as good as speech cutting efficiency. Nets 1-6 work on the set that has only 71%-78% blockades cut (see Table 3 *blk 55dB 50ms* section) so their results are significantly lower than sets 7-12 having 87%-94% blockades cut (see Table 3 *blk 54dB 50ms* section) or sets 13-18 having 95%-100% blockades cut (see Table 3 *blk 54dB 30ms* section). Files allblknn2 and allblknn3 have very good results. Of course these files were used in teaching the perceptron but we should remember that only 50% of fragments took direct part in teaching (learning set) while 25% of the fragments were not used at all (testing set).

We tested one file that was not used in teaching at all – allblknn1. As we can see the results are significantly lower but still good. After closer investigation it occurred that the file has a few series blockades that occur very fast one after another (like "p p p p publication"). Though the cutting algorithm cut them correctly, perceptron decided

Table 5. Disordered sound repetition recognition results in % in continuous speech using nets from Table 4. The best results are marked as bold.



| window length | net | distance, threshold | allblknn1 | | allblknn2 | | allblknn3 | |
|---|---|---|---|---|---|---|---|---|
| | | | Sens | Pred | Sens | Pred | Sens | Pred |
| 700 ms | 1 | 50ms, 55dB | 71 | 79 | 75 | 98 | 75 | 95 |
| 1000 ms | 2 | | 64 | 71 | 76 | 97 | 76 | 99 |
| 1500 ms | 3 | | 72 | 66 | 77 | 94 | 76 | 95 |
| 2000 ms | 4 | | 67 | 68 | 77 | 98 | 75 | 98 |
| 2500 ms | 5 | | 74 | 67 | 77 | 96 | 76 | 96 |
| 3000 ms | 6 | | 60 | 65 | 77 | 98 | 76 | 99 |
| 700 ms | 7 | 50ms, 54dB | 52 | 74 | 85 | 93 | 92 | 93 |
| 1000 ms | 8 | | 62 | 75 | 87 | 93 | 93 | 97 |
| **1500 ms** | **9** | | **70** | **83** | **87** | **98** | **93** | **98** |
| **2000 ms** | **10** | | **72** | **70** | **87** | **96** | **93** | **93** |
| **2500 ms** | **11** | | **75** | **72** | **87** | **96** | **93** | **96** |
| 3000 ms | 12 | | 60 | 71 | 87 | 96 | 93 | 96 |
| 700 ms | 13 | 30ms, 54dB | 58 | 67 | 94 | 93 | 93 | 93 |
| 1000 ms | 14 | | 61 | 76 | 94 | 98 | 92 | 98 |
| 1500 ms | 15 | | 67 | 75 | 94 | 97 | 93 | 98 |
| 2000 ms | 16 | | 44 | 72 | 91 | 98 | 90 | 99 |
| 2500 ms | 17 | | 60 | 72 | 93 | 100 | 91 | 100 |
| 3000 ms | 18 | | 51 | 69 | 88 | 100 | 93 | 100 |

that they were so close to each other that it had to be one fluent word. Because such a decision was applied to all blockades in one series (not only one), this lowered the recognition ratio heavily.

The last conclusion is connected with the result for the file allblknn1. Nets 7-12 which received 71%-78% blockades had better results than those 13-18 which received 95%-100% blockades. This means that perceptron cannot receive too many fluent fragments (nets 7-12 received 123 and nets 13-18 received 165) because it makes more mistakes though it has more blockade patterns to learn on.

## References

[1] Suszyński W., Kuniszyk–Jóźkowiak W., Smołka E., Dzieńkowski M., Automatic recognition of non-fluent stops, Annales UMCS Informatica (2004): 183.

[2] Wiśniewski M., Kuniszyk–Jóźkowiak W., Smołka E., Suszyński W., Improved approach to automatic detection of speech disorders based on the Hidden Markov Models approach, Journal Of Medical Informatics & Technologies 15 (2010): 145.

[3] Kobus A., Kuniszyk–Jóźkowiak W., Smołka E., Codello I., Speech nonfluency detection and classification based on linear prediction coefficients and neural networks, Journal Of Medical Informatics & Technologies 15 (2010): 135.

[4] Szczurowska I., Kuniszyk–Jóźkowiak W., Smołka E., Speech nonfluency detection using Kohonen networks, Neural Computing and Application 18(7) (2009): 677.

[5] Akansu A. N, Haddad R. A., Multiresolution signal decomposition, Academic Press (2001).

[6] Codello I., Kuniszyk–Jóźkowiak W., Wavelet analysis of speech signal, Annales UMCS Informatica AI 6 (2007): 103.

[7] Nayak J., Bhat P. S., Acharya R., Aithal U. V., Classification and analysis of speech abnormalities, Elsevier SAS 26(5-6) (2005): 319.

[8] Gold B., Morgan N., Speech and audio signal processing, JOHN WILEY & SONS, INC (2000).

[9] Smith J., Abel J, Bark and ERB Bilinear Transforms, IEEE Transactions on Speech and Audio Processing (1999).

[10] Goupillaud P., Grossmann A., Morlet J., Cycle-octave and related transforms in seismic signal analysis', Geoexploration 23 (1984–1985): 85.

[11] Traunmüller H., Analytical expressions for the tonotopic sensory scale, J. Acoust. Soc. Am. 88 (1990): 97.

[12] Codello I., Kuniszyk-Jóźkowiak W., Smołka E., Kobus A., Prolongation Recognition in Disordered Speech, Valencia, Spain, Proceedings of International Conference on Fuzzy Computation (2010): 392.

[13] Garfield S., Elshaw M., And Wermter S., Self-orgazizing networks for classification learning from normal and aphasic speech, In The 23rd Conference of the Cognitive Science Society, Edinburgh, Scotland (2001).

[14] Horzyk A., Tadeusiewicz R., Self-optimizing neural networks, Advances in neural networks - ISNN 2004, pt 1, Lecture notes in computer science 3173 (2004): 150.

[15] Horzyk A., Tadeusiewicz R., Mechanisms, symbols and models underlying cognition, pt 1, Proceedings, Lecture notes in Computer Science, 3561 (2005): 156.

[16] Kohonen T., Self-Organizing Maps 34 (2001): 2173.

[17] Tadeusiewicz R., Elementarne wprowadzenie do sieci neuronowych z przykładowymi programami, Akademicka Oficyna Wydawnicza, Warszawa (1998).

[18] Tadeusiewicz R., Sieci neuronowe, Akademicka Oficyna Wydawnicza, Warszawa (1993).

[19] Szczurowska I., Kuniszyk–Jóźkowiak W., Smołka E., Application of Artificial Neural Networks In Speech Nonfluency Recognition, Polish Jurnal of Environmental Studies 16(4A) (2007): 335.

[20] Codello I., Kuniszyk–Jóźkowiak W., Smołka E., Kobus A., Disordered sound repetition recognition in continuous speech using CWT and Kohonen network, Journal Of Medical Informatics & Technologies 17 (2011): 123.