Movement Noise Suppression by Neural Networks

Report about Research Internship

Alexander Schmidt

March 2015

Supervisor: Hendrik Barfuss

Contents

Chapter 1

Introduction

When a robot is moving, its recordings are highly corrupted by movement noise. This noise is non-stationary, as it depends on e.g the motor angle, the walking surface, ... A classical e.g. Wiener Filter approach to suppress this noise is therefore not suitable. In [1] a different method is proposed, in which the movement-noise spectrum is estimated using a neural network and then subtracted from mixed speech-noise-spectrum. The only thing what should remain after subtraction is the pure speech spectrum.
The mentioned paper gives simulation results for AIBO ERS-210, a small robot dog. During the research internship it should be investigated if such an approach for noise suppression is suitable for the NAO robot and, if yes, how well it works.
In the following the investigation results are presented. The report is therefore organized as follows: first, on the basis of above mentioned paper, the experiment set-up is explained. Here it is especially pointed out what is done differently from the paper and where the difficulties and challenges with the NAO robot lie. In a second part the results are presented and, afterwards discussed.

Chapter 2

Experiment Set-up

2.1 Conceptual Overview

A typical scenario for mentioned approach would be a person talking to NAO while it is walking or pitching the head. For simplicity we consider only the latter movement. The (movement) noise corrupted speech recordings are considered in the spectral domain. From this spectrum the predicted (ideally pure) noise spectrum is subtracted by so called 'Motion dependent spectral Subtraction' (MDSS). Its output, back transformed to time, should ideally contain only speech. The spectral estimates are output of a neural network which is fed by sensor data of NAO's head motors. Figure 2.1 gives an overview about the concept.

For implementation MATLAB was used, which offers a nice and easy accessible toolbox for neural networks.

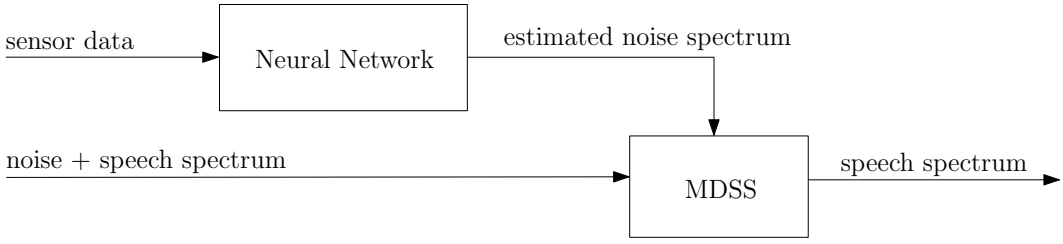

Figure 2.1: Concept of movement-noise reduction using neural network

2.2 Sensor Data and Recordings

Using the Choregraphe-Suite Interface provided for the use of NAO, recordings and sensor samples of executed movements can be taken. NAO is therefore connected via Ethernet to a computer client.

The audio recordings have a samplings frequency of $f_s = 48kHz$ and consist of four channels, where each does represent one of NAO's four microphones. They are saved in *.wav* format. It was observed that NAO stops recording after $40s$, although a longer recording time is wished.

For the pitch movement of NAO, data of four sensors can be used: angle of pitch motor, angle of corresponding joint, hardness and motor's current. Sensor data can be collected with sampling time of minimum $T = 0.02s$, all values below caused problems in the Ethernet connection. Sensor recordings are saved in *.csv* format.

Two other issues complicated the collection of train and test data: Recording of sound and sensor data could not be started simultaneously and, what is more, time stamps of sensor data are highly irregular. First point required a post-recording synchronization ($\rightarrow$ `preprocessing.m`). Second issue was solved by selecting simply the nearest available sampling instant for a given time stamp.

After this, collected samples can be depicted like in Figure 2.2. Clearly, changes in e.g. angle goes ahead with a pattern in the audio recording. It is also obvious that there is no difference between angle of motor and angle of joint (what is not surprising). The hardness is some kind of absolute first derivative of the angles. The current has quite a twitchy development which does not bear too much information.

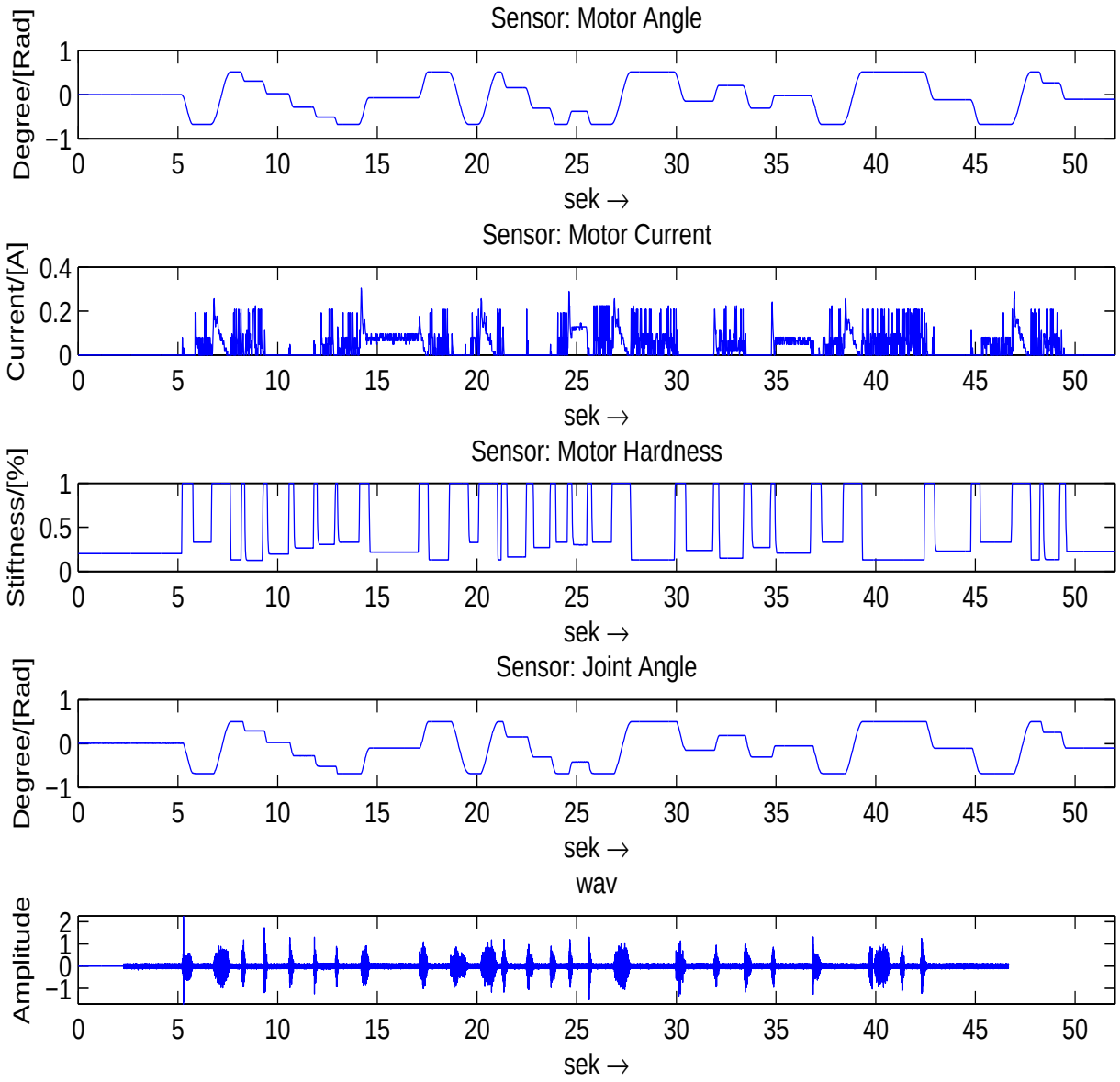

Figure 2.2: *.wav* recording (very bottom) aligned with different sensor data: motor angle, motor current, motor hardness, joint angle (from top to bottom)

2.3 Spectrum/Filterbank

A provided filterbank was used to transform time data to spectral domain and, vice-versa, shift it back to time domain (analysis/synthesis). Reconsidering Figure 2.1, the filterbank-analysis part is placed on the bottom left-hand, just before noise-speech spectrum. The synthesis part would conclude on the bottom right hand side the speech spectrum flow.

To guarantee alias-free analysis/synthesis and a sliding data window shift which matches approximately sensor sampling time T, a filter bank output size of 157 banks is chosen. However, this size would also be the number of output nodes of the neural network (see below). This implies, as a consequence, a big network architecture with many layers, nodes per layer, ... what, as a consequence again, would require a huge amount of training data. To avoid this, the obtained 157-bin spectrum is reduced to 25 bins, by simple averaging several bands together. Before synthesis, the 25 bins are enlarged back to 157 by repeating and lowpass filtering.

Note that the band size 25 is somehow arbitrarily chosen and just adopted from mentioned paper.

2.4 Neural Network (c.f. → `train.m`, `test.m`)

As described before, MATLAB was used for this purpose, as it offers a nice neural network toolbox. It provides easy configuration, training and testing of a network (c.f. [2]).

After the number of layers and the number of nodes per layer are chosen, training requires matrices of input and output data. The number of rows of the matrices does represent the number of inputs/outputs nodes to the neural network. The different columns represent the sequential character of the data, i.e. the development over time, time frames in our case. While the amount of input and output nodes, and hence the number of rows of input and output matrices, can differ, the number of columns must be identical.

In the following we have a closer look on the individual input and output matrices. The input matrix is constructed in such a way that it does not only contain the required sensor data itself, but also delays (in positive like negative direction) as well as derivatives. Let, for example, $s(k)$ be a required sensor value at time stamp k. With forward-backward delay of $D = 1$ the following input sensor matrix S is constructed:

$$S = \begin{pmatrix} s(0) & s(1) & s(2) & ... \\ s(1) & s(2) & s(3) & ... \\ 0 & s(0) & s(1) & ... \\ 0 & \frac{s(2)-s(0)}{2} & \frac{s(3)-s(1)}{2} & ... \\ 0 & \frac{s(3)-s(1)}{2} & \frac{s(4)-s(2)}{2} & ... \\ 0 & \frac{s(1)-0}{2} & \frac{s(2)-s(0)}{2} & ... \end{pmatrix}.$$

Here, the first line is the original sensor data, second line shifted by one time step to the left/future, third line shifted by a time step to the right/past. The last the rows contain differences of lines above, fourth row is related to first row, fifth row to second, and so on. The derivative at time step k is calculated as halved difference of values at $k + 1$ and $k - 1$. The third column, for example, incorporates such knowledge of 5 sampling instants.

The output matrix contains the absolute STFT spectras, i.e. the 25 absolute frequency bins of the filterbank column-wise, and the development over time frames row-wise.

2.5 Motion dependent spectral Subtraction: MDSS

(c.f. → `MDSS.m`)

Once a spectrum is estimated by the neural network, it must be subtracted from the noise-corrupted one. This is done via 'Motion dependent spectral Subtraction' (taken from [1]). For the n-th frequency bin we calculate

$$|S_{MDSS}(f_n)|^2 = \max \begin{cases} |S(f_n)|^2 - \alpha \cdot |\tilde{S}(f_n)|^2 \\ \beta \cdot |S(f_n)|^2, \end{cases}$$

with $\alpha \geq 1$, $0 \leq \beta \leq 1$. $S(f_n)$ is spectrum of original noise corrupted speech signal, $\tilde{S}(f_n)$ is the estimated noise spectrum.

The value of $\alpha \geq 1$ does imply that we actually underestimate the spectrum, what turned out to be true during simulations.

Note, that this MDSS is nothing but a normal and very well-known spectral subtraction. The authors of [1] gave it a new name as one of its inputs is dependent on the motion of the robot.

After subtraction, any phase estimation is lost. Before the synthesis part of the filterbank is performed, the pure real valued subtraction outcome $|S_{MDSS}(f_n)|$ must therefore be phase corrected. For this, it is multiplied with the phase of the original noise-corrupted spectrum. The input to the synthesis filter $S_{clear}(f_n)$ reads therefore

$$S_{clear}(f_n) = |S_{MDSS}(f_n)| \cdot \exp(j \angle S(f_n)),$$

where $\angle$ denotes the angle operator. This phase correction is not mentioned in Figure 2.1.

Chapter 3

Results

3.1 Scenario & Simulation Results

To make statements about the goodness of the estimation and noise suppressionas a whole, we compare word recognition rates for an unprocessed and a processed movement-noise corrupted speech signal at different SNRs.
The results are based on a testing scenario described in the following:

- Recordings contain pure movement noise. In Matlab manually speech was added with different amplification constants to obtain required SNRs. As mentioned before only the pitchment of the head was regarded.
- The used neural network has 1 layer with 40 nodes, training data consists of ≈ 6500 feature vectors.
- Out of the four available sensors, only the motor angle was chosen as input. As it was described above, the other three hardly provides more information: joint angle is just the same as motor angle, electric current is very twitchy and hardness is implicitly taken into account by the derivatives of motor angle.
- $D = 2$, 25 output frequency bins
- Speech recognizer: CHiME Challenge/PocketShpinx

To find a good parameter constellation α, β for MDSS, a brute force simulation was started, checking different combinations of both constants. For this a SNR had to be fixed (here SNR $10dB$). The result is shown in Figure 3.1. It is clearly observable that there are different parameter constellations resulting in equal high recognition rates. $\alpha = 1.5,\ \beta = 0.12$ was chosen for further considerations.
The final result is shown in Figure 3.2. Over a large SNR area the processed signal results in higher recognition rates. However, the difference to the unprocessed reference is not very significant ($\approx 3 - 5\%$). This is highly unsatisfactory.

3.2 Discussion

The presented results require discussions about the reasons for the only slight improvement of word recognition rate. There are several possible explanations for the bad performance which should be presented shortly in the following.

- Estimated spectrum: Figure 3.3 shows on the top the spectrum over time for some pitching of NAO's head (no speech). On the bottom the estimated spectrum for comparison is shown. It is clearly visible that the neural network predicts at least the temporal structure of spectrum quite well, as nonzero components are time aligned in top and bottom spectras. However, this good prediction does not hold for the fine structure. Figure 3.4 is a zoom of one movement in Figure 3.3. The very fine structure of top noise spectrum is badly estimated by the neural network. Its output is some kind of blurred. From this point of view it does not come as surprise that a spectral subtraction cannot perform well and the overall result is not better than expected.

- Architecture and convergence of neural networks, trainings data: The above presented results are based on a parameter constellation (i.e. number of layers, number of nodes per layers, delays D, ...) which is a guess in the blue. Perhaps there are better constellations, but it stays the problem that there is know systematic

way known how to derive them. To make things worse, all parameters are most probably dependent on each other. When e.g. changing the delay D, the input size of the network changes. This requires a larger more layers/nodes per layer. In turn more trainings data is needed.

Of course different e.g. network architectures were tested. However, no other constellation resulted in a clear better performance. For all constellations it remained a uncertainty about the overall convergence of the networks. This is a point for closer investigations.

- Synchronization: As mentioned in the beginning, sensor data and corresponding audio recordings of NAO are heavily unsynchronized. This problem was tried to be solved by a synchronization routine, using simple edge detection. Of course such a very first approach is perhaps not that accurate as required. More sophisticated synchronization routines may result therefore in a better e.g. convergence of network and, hence, a better overall performance.

  In this context it is desirable that Aldebaran Robotics solves this issue in an upcoming software release.

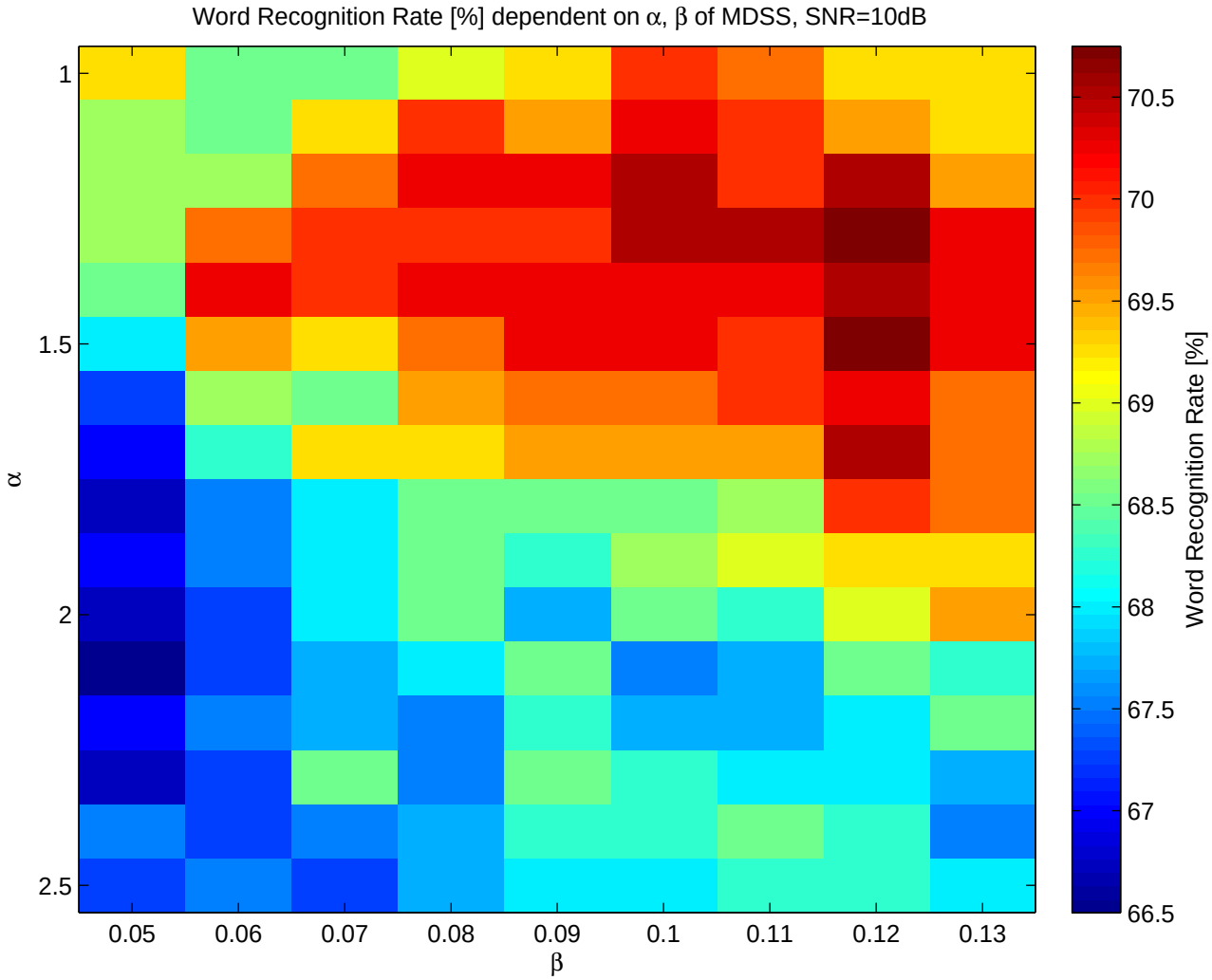

Figure 3.1: Word recognition rate dependent on different constellations of α, β. $SNR = 10dB$.

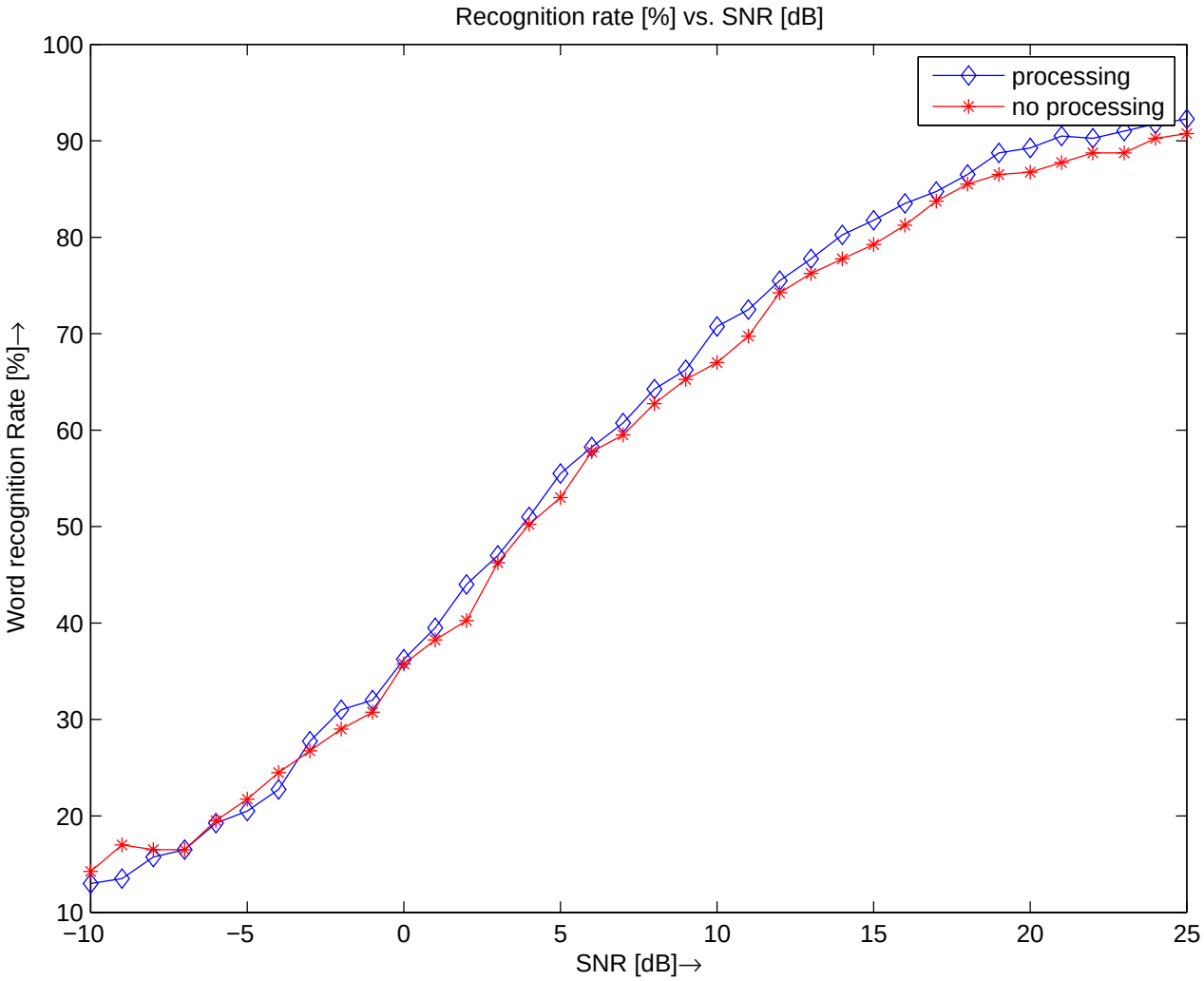

Figure 3.2: Word recognition rate dependent on the SNR for no processing (red) and processing by described method (blue).

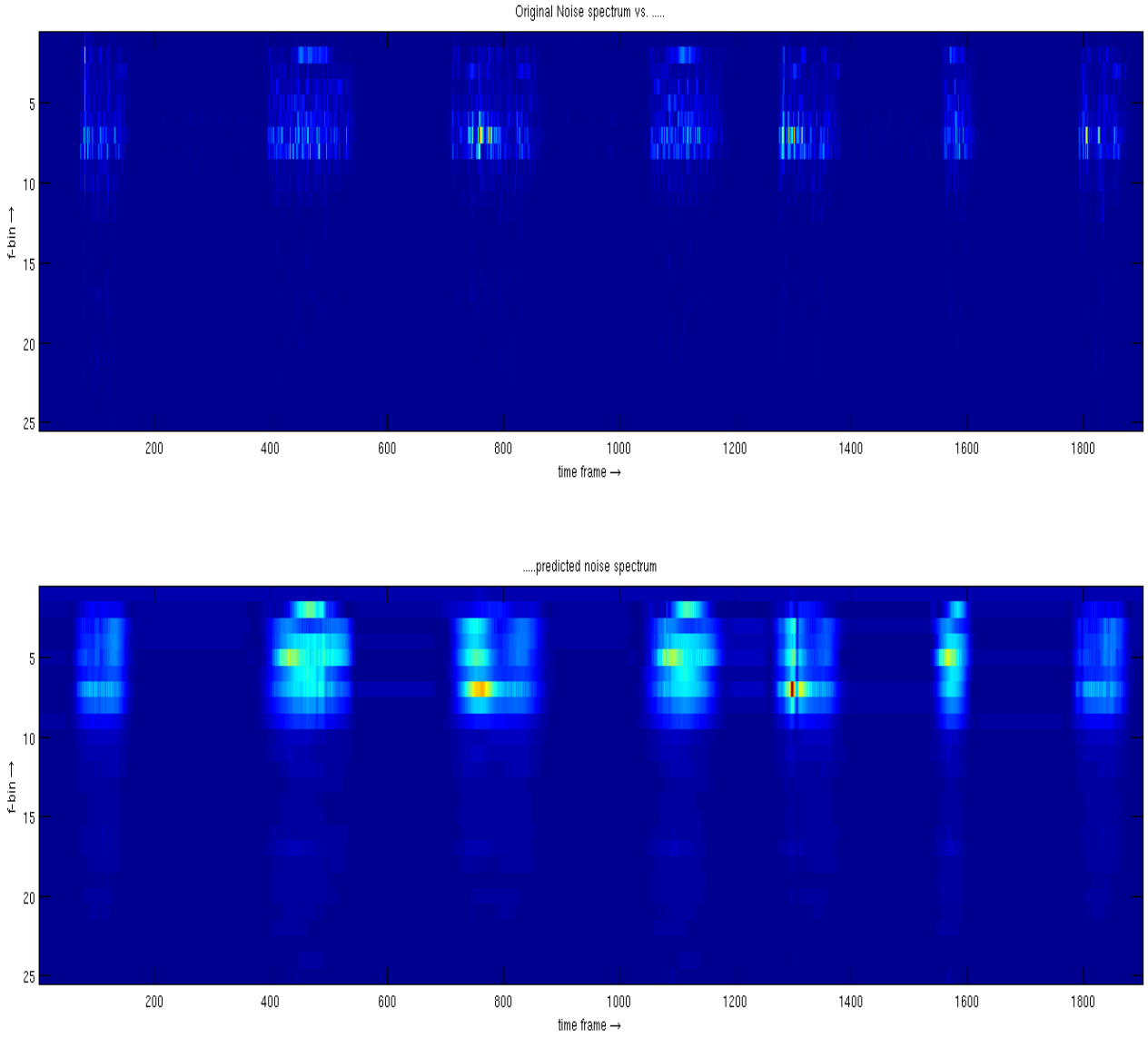

Figure 3.3: STFT spectrum of a pure noise recording (top). Estimated noise spectrum (bottom). It is clearly visible that at least the rough structure is well predicted.

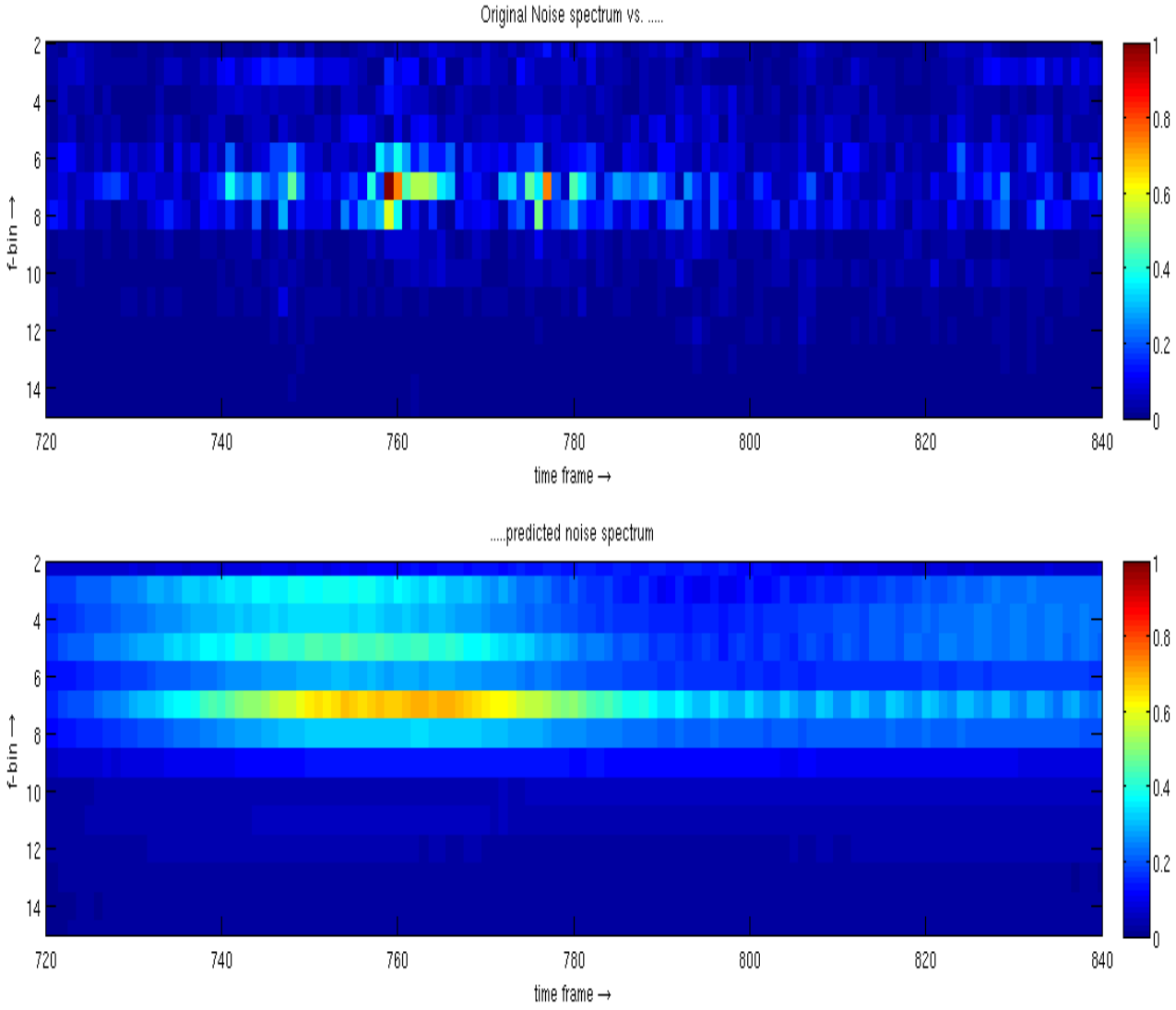

Figure 3.4: Zoom of Figure 3.3. The fine structure, however, is not well estimated.

Chapter 4

Conclusion & Acknowledgments

From the beginning, this research internship had a quite exploring character as it was unknown if the presented approach can succeed at all. But despite the quite unsatisfactory results, the project was not fully useless. First of all the author's personal aim to collect experiences with the use of neural networks was met. Furthermore problems and challenges could be identified on which, if work is continued, one has to take special care on. Overall a more structured approach is needed then.
Of course the presented neural network idea is very attractive, particularly because one has not to understand in full detail what the network is doing. But perhaps it got clear that such 'black-box' thinking is not always that fruitful. Best results are definitely archived when the influence of all parameters and their effects are well understood.

The author wants to thank Hendrik Barfuss, M.Sc., for the great supervision and help to clarify and discuss open questions. Many thanks also to Christian Hümmer, M.Sc., for his consultancy in questions concerning neural networks.

Bibliography

[1] Akinori Ito et al. Internal noise suppression for speech recognition by small robots. Interspeech 2005, Lisbon, Portugal, 2014.

[2] *The MathWorks, Inc.* Neural Network Toolbox, User's Guide 2014a, 2014.

[3] Christopher M. Bishop. *Pattern Recognition and Machine Learning.* Springer, Berlin, Heidelberg, corr. 2nd edition, 2011.