Discrete Fourier Transform Computation Using Neural Networks

Rosemarie Velik
Vienna University of Technology
velik@ict.tuwien.ac.at

Abstract

In this paper, a method is introduced how to process the Discrete Fourier Transform (DFT) by a single-layer neural network with a linear transfer function.
By implementing the suggested solution into neuro-hardware, advantage can be taken of actual parallel processing of spectral components of different frequencies and of different coefficients of each spectral line. When computing the DFT due to input data pre-processing for a certain neural network solution, a stand alone solution of neural networks without the necessity of additional computational resources can be achieved.

1. Introduction

Neural networks are used in many different application domains in order to solve various information processing problems. They have proven to be successful in pattern recognition, pattern classification, function approximation, prediction, optimization, and controlling [6]. The basic processing elements of neural networks are artificial neurons. In order to perform complex tasks, these neurons have to be interconnected in an adequate way by arranging them in an appropriate architecture and setting the weights of their interconnections. The setting of weights is performed during a training phase. In a supervised learning process, input data and target data pairs are presented to the network and the weights of the connections are altered by a learning algorithm as long as for (ideally) all input data, the output values of the network match to the target data [5].
To find an efficient solution to a certain problem, pre-processing of input data is essential to reduce the complexity of the network and the time needed for training [1], [3], [7].
In many applications, time signals of physical values serve as input raw data for neural networks. To reduce the size of the network, it is often recommendable not to use the time signal itself but to extract certain features from this time signal. In a large number of cases, using a part of the frequency spectrum of time signals as actual input data for a neural network turns out to be suitable. Using the spectrum of a time signal can reduce the size of the input vector of a neural network and can therefore diminish the complexity and the size of the neural network necessary for solving a certain problem.
In [8], an extensive overview about applications using frequency spectra as input data for neural networks is given. Applications are various and include domains like acoustic signal analysis, speech analysis, analysis of bio-signals, vibration monitoring, analysis of energy supply systems, or image recognition.
For discrete time signals, the frequency spectrum of these signals can be calculated with the Discrete Fourier Transform (DFT). With a digital computer, the DFT can be calculated quite easily. However, by using a computer for this calculation, the potential advantage of parallel processing of different frequency components is lost and a stand alone solution of a neural network is not achievable.
In this paper, a solution is presented how to process the DFT of discrete time signals by a neural network, which offers the advantage of actual parallel processing and for the application of data pre-processing for neural networks the possibility of a neural network stand alone solution.

2. Related work

Until now, not many investigations have been performed concerning the calculation of frequency spectra by neural networks. In [4], the advantage of parallel processing of digital cellular neural networks (CNNs) is exploited to calculate the Fast Fourier Transform (FFT). However, the disadvantage of the FFT is that it can only be applied to time signals, which consist of 2^N sampling points, where N is a positive integer value.
[2] describe how to substitute the FFT calculation of harmonic distortions in power systems by neural

networks. However, these networks do not perform a FFT calculation.
In literature, up to now, no solution can be found how to compute a DFT by neural networks.

3. Frequency spectra of discrete time signals

When calculating the frequency spectrum of discrete time signals, it has to be distinguished between two cases: aperiodic and periodic time signals. Their differences will be discussed in the following.

3.1. Aperiodic signals

When sampling an aperiodic signal with dirac impulses of a distance of a sampling time Ts, the spectrum of the signal can be calculated with the formula

$$X(j\omega) = T_S \sum_{k=-\infty}^{\infty} x(kT_S) e^{\frac{-jk\omega}{T_S}} .$$

Figure 1 illustrates a discrete time signal and its frequency spectrum.

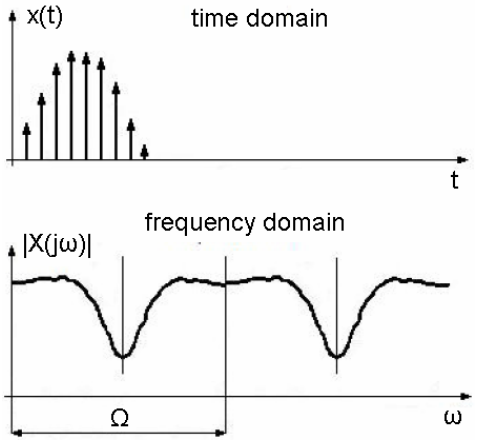

Figure 1: Frequency spectrum of discrete, aperiodic time signals

As can be seen from this figure, the spectrum is periodic with a period of $\Omega = \frac{T_S}{2\pi}$. By using a low pass filter, the periodically repeating components of the frequency spectrum can be eliminated and there only remains the first period, which is symmetric round the frequency f = 0. Therefore, the Nyquist theorem has to be fulfilled, according to which the bandwidth of the signal has to be smaller than half the sampling frequency:

$$bandwidth < \frac{\omega_s}{2}$$

3.2. Periodic signals

The form of the spectrum of a discrete periodic signal is shown in figure 2.

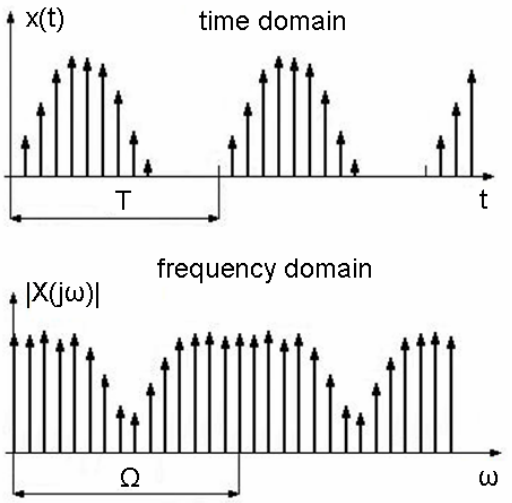

Figure 2: Frequency spectrum of discrete, periodic time signals

Assuming that the time period of the sampling signal is an integer multiple of the sampling time Ts, the frequency spectrum is a periodic line spectrum with lines at multiples of

$$\omega_0 = \frac{2\pi}{N \cdot T_S} = \frac{\omega_s}{N} .$$

The spectrum of the signal is calculated by the discrete Fourier Transform (DFT). The k^{th} spectral line can be calculated by the formula

$$X(k) = \sum_{n=0}^{N-1} x(n) \cdot e^{-jkn\frac{2\pi}{N}}$$

where k = 0, 1, 2, 3, …, N-1.

As can be seen from this formula, N spectral lines of a periodic sampling signal can be calculated from N sampling values. In sum, for a calculation of N spectral lines, N^2 complex multiplications are necessary. For realistic values of N = 1000, this results in a non-neglectable computation effort.
The advantage of the DFT is that the whole information of a measurement signal is represented in N sampling values and spectral lines, respectively.

Precondition for its application is that the measurement signal is representative within the constrained time interval $N \cdot T_S$. One problem with the application of the DFT is that most measurement signals are not periodic. Therefore, for a continuous signal, there has first to be cut out a certain "window" within which the continuous signal is substituted by its sampling signals at the points of time $N \cdot T_S$. For applying the DFT, it is then assumed that the windowed, sampled signal is continued periodically. With certain constraints, the DFT can also be applied to aperiodic signals in order to deliver information about its spectrum at multiples of ω_0. As for a neural network the input vector always has to consist of discrete values, the application of the DFT for data pre-processing is justified.

4. Neural network for DFT calculation

The aim of this paper is to present a solution how to process the DFT by using neural networks. Therefore, principally two different approaches are conceivable. The first possibility is to train a neural network with a number of "input data"-"target data"-pairs of time signals and their correlating frequency spectra and "hope" that the weights of the networks are altered in an adequate way by the learning algorithm in order to be able to generalize also over unseen signals after the training process. However, experiments performed with different network architectures, different numbers of layers and neurons in these layers, as well as neurons with different transfer functions showed that with a realistic effort, neural network structures could not learn to process DFT calculations for unseen input signals.

As it turned out, in order to achieve that a neural network processes a DFT, a strategy has to be applied which maps the mathematical formula of the DFT to the structure of a neural network. This method does not require network training. The concept shown in the following:

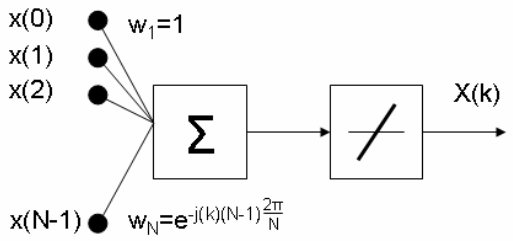

Figure 3 Neuron for calculation of spectral value X(k)

As outlined in section 3.2, by applying the DFT, the k^{th} spectral line of a discrete time signal is calculated by the formula

$$X(k) = \sum_{n=0}^{N-1} x(n) \cdot e^{-jkn\frac{2\pi}{N}} . \quad (1)$$

This section now investigates how a neural network can be used to calculate the DFT of a discrete time signal by considering this formula.

Regarding the neural network as black box in a first step, the input vector of the neural network is the time signal x(n) at N discrete points of time and the output shall be the spectral value X(k). From N time values, there can be maximally calculated N spectral values. To get the spectral value X(k), each of the N time values x(n) has to be multiplied by the factor $e^{-jkn\frac{2\pi}{N}}$ and all these products have to be summed up. It now turns out that this task can be performed by a neuron without bias, with a linear transfer function, and with the values $e^{-jkn\frac{2\pi}{N}}$ as weights of the input signals. The structure of a neuron that performs this task is illustrated in figure 3.

To calculate a whole spectrum, for each required frequency value, one such neuron needs to exist. If for instance 20 spectral values shall be calculated, 20 parallel neurons are necessary, each of them receiving the same input values and only differing in their weights depending on the value k. The resulting network structure is illustrated in figure 4. To preserve the overview, the values of the weights of the particular connections are not depicted. As already mentioned, the values of the weights are not learned from examples but result from the factor $e^{-jkn\frac{2\pi}{N}}$ of formula 1 and differ from each other by the values n and k. All connections coming from the input x(0) always have the weight $e^{-jk\cdot 0\cdot\frac{2\pi}{N}} = 1$.

Connections coming from the input x(1) have the values

$$[e^{-j\,0\cdot 1\cdot\frac{2\pi}{N}}, e^{-j\,1\cdot 1\cdot\frac{2\pi}{N}}, \ldots, e^{-j(k-1)\cdot 1\cdot\frac{2\pi}{N}}, e^{-jk\cdot 1\cdot\frac{2\pi}{N}}] =$$

$$[1, e^{-j\frac{2\pi}{N}}, \ldots, e^{-j(k-1)\cdot\frac{2\pi}{N}}, e^{-jk\cdot\frac{2\pi}{N}}]$$ depending on the output k they contribute a value to. For the remaining connections, the same principle of weights calculation is applied by just altering the value of n.

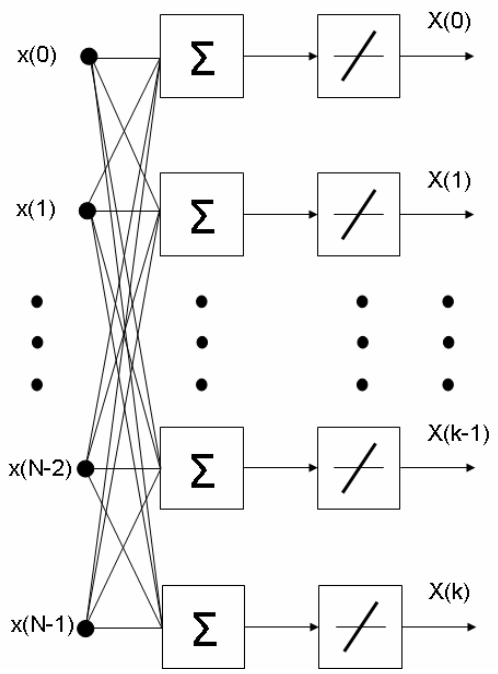

Figure 4 Neural network for calculation of k spectral values

5. Simulation and testing

To test the introduced neural network DFT calculation concept, it was tested in a MATLAB simulation by comparing the output values of the neural network solution with the values calculated when directly applying the DFT to the same input data.

By using the MATLAB neural network toolbox, a single-layered neural network with a linear transfer function can easily be realized and its weights, which are stored in a matrix, can be set to the adequate values.

In order to calculate the DFT algorithmically, the MATLAB command *fft* can be applied to the input data vector. The function *fft* applies the FFT if its input vector consists of 2^N values (were N is a positive integer value), and it applies the DFT for all other input vector sizes.

For test purposes, from time signals with 1001 measurement points, 100 spectral values were calculated by using a one-layer neural network with consisting of 100 neurons with a linear transfer function. The weights of the input layer corresponded to the e-function of formula 1.

A comparison to the results of the *fft*-function of MATLAB showed the output values of the neural network were exactly equal to the spectral values provided by the neural network, which proves that the neural network is suitable for performing DFT calculations.

6. Conclusion

In this paper, a concept was outlined how to calculate the Discrete Fourier Transform (DFT) with a single-layered neural network with a linear transfer function. This offers an efficient method for calculating the DFT. When being implemented in neuro hardware, the neural network offers the advantage of actual parallel processing of spectral components of different frequencies and of different coefficients of each spectral line.

The application of the introduced concept is particularly recommendable for data pre-processing of input data of neural networks as with this method, there is no necessity for additional computational resources for the pre-processing stage.

7. References

[1] Howard Demuth and Mark Beale, *"Neural Network Toolbox – For Use with MATLAB",* The MathWorks, user's guide version 4 edition, 2005. ISBN 3-528-05428-X.

[2] W. W. L. Keerthipala, Low Tah Chong, and Tham Chong Leong, *"Artificial Neural Network Model for Alalysis of Power System Harmonics"*, IEEE.

[3] Gesellschaft für Mess- und Automatisierungs-technik, *"Computational Intelligence – Neuronale Netze, evolutionäre Algorithmen und Fuzzy Control im industriellen Einsatz"*, VDI Verlag, 1998, ISBN 3-18-091381-9.

[4] Martin Perko, Iztok Fajfar, Tadej Tuma, and Janez Puhan, *"Fast Fourier Transform Computation using a Digital CNN Simulator"* IEEE, 0-7803-4867, 1998, pp. 230-235.

[5] Raúl Rojas, *"Theorie der neuronalen Netze - Eine systematische Einführung"*, Springer Lehrbuch, 1996 ISBN 3-540-56353-9.

[6] Andreas Scherer, *"Neuronale Netze - Grundlagen und Anwendungen"*, Vieweg Verlag, 1997, ISBN 3-528-05465-4.

[7] Alberto Sanfeliu, José Francisco Martínez Trinidad, and Jesús Ariel Carrasco Ochoa, *"Progress in Pattern Recognition, Image Analysis and Applications"*, Springer Verlag, 2004, ISBN 3-540-23527-2.

[8] Rosemarie Velik, "Neuronale Muster- und Objekterkennung durch Analyse im Zeit- und Frequenzbereich", Master Thesis, Vienna University of Technology, 2006.