information

Article

Design and Analysis of Binary Scalar Quantizer of Laplacian Source with Applications

Zoran Peric [1], Bojan Denic [1,*], Milan Savic [2] and Vladimir Despotovic [3]

[1] Faculty of Electronic Engineering, University of Nis, Aleksandra Medvedeva 14, 18000 Nis, Serbia; zoran.peric@elfak.ni.ac.rs
[2] Faculty of Sciences, University of Pristina—Kosovska Mitrovica, Ive Lole Ribara 29, 38220 Kosovska Mitrovica, Serbia; milan.savic1@pr.ac.rs
[3] Department of Computer Science, University of Luxembourg, Avenue de la Fonte 6, L-4364 Esch-sur-Alzette, Luxembourg; vladimir.despotovic@uni.lu
* Correspondence: bojan.denic@elfak.ni.ac.rs; Tel.: +381-1852-9367

Received: 15 September 2020; Accepted: 20 October 2020; Published: 27 October 2020

Abstract: A compression method based on non-uniform binary scalar quantization, designed for the memoryless Laplacian source with zero-mean and unit variance, is analyzed in this paper. Two quantizer design approaches are presented that investigate the effect of clipping with the aim of reducing the quantization noise, where the minimal mean-squared error distortion is used to determine the optimal clipping factor. A detailed comparison of both models is provided, and the performance evaluation in a wide dynamic range of input data variances is also performed. The observed binary scalar quantization models are applied in standard signal processing tasks, such as speech and image quantization, but also to quantization of neural network parameters. The motivation behind the binary quantization of neural network weights is the model compression by a factor of 32, which is crucial for implementation in mobile or embedded devices with limited memory and processing power. The experimental results follow well the theoretical models, confirming their applicability in real-world applications.

Keywords: quantization; Laplacian distribution; clipping factor; signal to noise ratio; pulse code modulation; delta modulation; neural network

1. Introduction

Quantization can be classified into two categories: scalar and vector [1,2]. The classification has been made based on whether only one sample is quantized at a time, using a fixed number of bits per sample (scalar quantization), or a number of samples is quantized at a time (vector quantization). The main advantage of scalar quantization is the reduced design complexity, which may be a crucial point in applications where processing delay is a critical parameter, such as speech coding. In scalar quantization, the real line is divided into a certain number of non-overlapping cells and, for each cell, the representative level is defined [1–3]. Hence, the input data is mapped into the appropriate representative level, depending on the cell where the input belongs.

The binary quantizer is the simplest scalar quantization model, where each symbol is represented by only one bit. The main benefit achieved with this model concerns data compression, rather than achieved signal quality. Besides being widely used in speech [1–9] and image coding [1,10–13], recently, the most prominent application became the compression of neural networks [14–21]. Although extensively exploited, a detailed analysis of binary quantization from the viewpoint of signal (data) processing by assuming known data distribution, including a design for the reference variance and performance analysis in a wide dynamic range of variances, is not available in the literature. This motivated the authors to perform a detailed analysis of such a quantization model. The data is assumed to follow the Laplacian probability density function (PDF), which is known to model various real data well, including speech [1–3,22], differences between neighboring pixels in an image [1], or weights in neural networks [23,24]. In brief, the main contributions of this paper can be summarized as follows:

- We introduce two types of binary non-uniform scalar quantizers, named binary quantizer type 1 and binary quantizer type 2, and provide detailed descriptions of the design methods. Furthermore, we investigate the effect of clipping with the aim of reducing the quantization noise. Quantizers are designed for the memoryless Laplacian source with zero-mean and unit variance.
- We conduct a detailed analysis of both binary quantizers and provide recommendations for quantizer selection in applications where the non-optimal design is required.
- We analyze the performance of both quantizers in a wide range of input data variances and investigate the robustness property.
- We propose a method to improve the performance in a wide dynamic range that is based on the forward adaption technique.
- We verify the correctness of the theoretical quantizer models by applying them to several real data, including speech, image, and neural network parameters.

The rest of the paper is organized as follows. Section 2 deals with the overview of state-of-the-art applications of scalar quantization. In Section 3, two design approaches for the non-uniform binary scalar quantization are described in detail, and an appropriate comparison of the models is given. In Section 4, an analysis in a wide dynamic range is provided. In Section 5, application to speech coding, image coding, and neural networks is presented, and the obtained results are discussed. Finally, Section 6 concludes the paper.

2. Previous Work

Scalar quantization models have been used in various data processing applications and, hence, play an important role in signal processing and compression. In this section, we provide a brief overview of state-of-the-art applications in several research areas, including speech coding, image coding and neural network compression.

Speech belongs to the class of time-varying signals. Therefore, for its efficient processing, adaptive schemes that operate on a frame-by-frame basis are recommended [1,2]. Frame here refers to the group of consecutive samples of finite lengths. We are particularly interested in quantization techniques such as delta modulation (DM), where compression is more important than the preserved quality of the speech signal. DM is a well-known predictive coding technique where the difference between successive samples is encoded using a single bit [1–3,25,26]. The improved version, known as adaptive delta modulation (ADM), has been proposed to overcome the issues of DM, known as overload and granular distortion. The algorithms based on ADM can be classified into instantaneous (where the adaptation is done at the sample level) [6] and frame-based (where adaptation is done at the frame level) [7–9]. Although in its original implementation ADM assumes the use of a one-bit (two-level) quantizer, some recent works propose substantial performance improvements using two-bit [6,7], two-digit [8], and three-level ADMs [9], with only a minimal increase in complexity.

Block truncation coding (BTC) [10–13] is a lossy compression algorithm widely used for the compression of monochrome images, which divides images into non-overlapping blocks, estimates the statistical parameters (mean and variance) for each block, and uses a binary quantizer to represent pixels, based on the block statistics [10,11]. Modifications that assume the utilization of high-rate non-uniform quantizers are also analyzed in literature [12,13].

Neural networks (NNs) are a research area where quantization has recently taken an important role [14–21,23,24,27–35]. The main motivation for introducing quantization of NN parameters is to provide model compression when compared with full precision cases, which is of crucial importance for implementation in portable and edge computing devices with limited memory and processing power, or in latency-critical services. The effects of scalar quantization in NNs have been analyzed so far in numerous papers, and it was shown that the quantization of network weights and activations to 8-bit fixed point representations has only a negligible effect on performance, but a further reduction of precision may lead to severe performance degradation [27–29]. Recent attempts provide competitive performance using bitrates lower than eight bits [23,24,30,31], or even binary quantizers in an extreme case [14–21].

3. Design Methods of Binary Quantizer

The binary scalar quantizer (N = 2 levels) was characterized by three parameters: Decision threshold t and representative levels y_1 and y_2. The model we addressed was zero-symmetrical, which assumed t = 0 and $y_1 = -y_2$. Accordingly, the design target was to find only the representative level in the positive part y_2. Usually, during the design process, the input data is modeled by PDF, and a certain performance criterion is adopted to select the required parameter. In this paper, we supposed that the data followed the Laplacian PDF given by [1–3]:

$$p(x,\sigma) = \frac{1}{\sqrt{2}\sigma}\exp\left(\frac{\sqrt{2}|x|}{\sigma}\right) \tag{1}$$

where σ^2 is the variance of the data.

The signal-to-quantization noise ratio (SQNR) is a widely employed quantizer performance measure defined as [1–3]:

$$SQNR = 10\log_{10}\left(\frac{\sigma^2}{D}\right) \tag{2}$$

where D is the mean-squared error (MSE) distortion:

$$D = 2\int_0^\infty (x-y_2)^2 p(x,\sigma)dx \tag{3}$$

Minimal MSE distortion, or an equivalently maximal SQNR, is the most common performance criteria. In the following subsections, we describe in detail the design of two binary quantizer models. In particular, the design is performed by assuming the unit variance ($\sigma^2 = \sigma_{ref}^2 = 1$); that is, we used $p(x,\sigma = 1)$, which stands for a standard approach in scalar quantization [1].

3.1. Binary Quantizer Type 1

In the design of this quantizer, we introduced the clipping factor x_{clip}, an additional parameter that served as the upper support region threshold, and also the step size $\Delta = 2x_{clip}/N$, assuming that $y_2 = \Delta/2$. Figure 1 illustrates the considered quantizer.

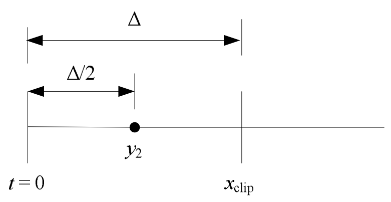

Figure 1. Illustration of binary quantizer type 1.

According to this model, the real line was divided into two regions: An inner region defined as $(-x_{clip}, x_{clip})$ and an outer region defined as $(-\infty, -x_{clip}) \cup (x_{clip}, \infty)$. Therefore, the MSE distortion was composed of two components. The first one, inner distortion D_i, can be evaluated as:

$$D_i = 2\int_0^{\Delta}\left(x-\frac{\Delta}{2}\right)^2 p(x)dx = 1-\frac{\Delta}{\sqrt{2}}+\frac{\Delta^2}{4}-\left(1+\frac{\Delta}{\sqrt{2}}+\frac{\Delta^2}{4}\right)\exp\left(-\sqrt{2}\Delta\right) \tag{4}$$

The outer distortion D_o, the second component, can be evaluated as:

$$D_o = 2\int_{\Delta}^{\infty}\left(x-\frac{\Delta}{2}\right)^2 p(x)dx = \left(1+\frac{\Delta}{\sqrt{2}}+\frac{\Delta^2}{4}\right)\exp\left(-\sqrt{2}\Delta\right) \tag{5}$$

Based on the last two expressions, the total distortion becomes

$$D = 1-\frac{\Delta}{\sqrt{2}}+\frac{\Delta^2}{4} \tag{6}$$

In terms of x_{cilp}, it can be expressed as

$$D = 1-\frac{x_{clip}}{\sqrt{2}}+\frac{x_{clip}^2}{4} \tag{7}$$

In the case of binary quantization, the following $x_{clip} = \Delta$ identity holds.

It can be seen that the MSE distortion is the function of the parameter x_{clip}. Accordingly, the optimal parameter value that minimizes MSE distortion can be obtained by setting the first derivative of distortion, with respect to x_{clip}, to zero:

$$\frac{\partial D}{\partial x_{clip}} = 0 \Rightarrow x_{clip} = x_{clip}^{opt} = \sqrt{2} \tag{8}$$

Hence, the optimal representative level amounts to $y_2^{type\ 1} = 1/\sqrt{2}$. The result obtained in (8) can also be verified by numerical simulation, as shown in Figure 2.

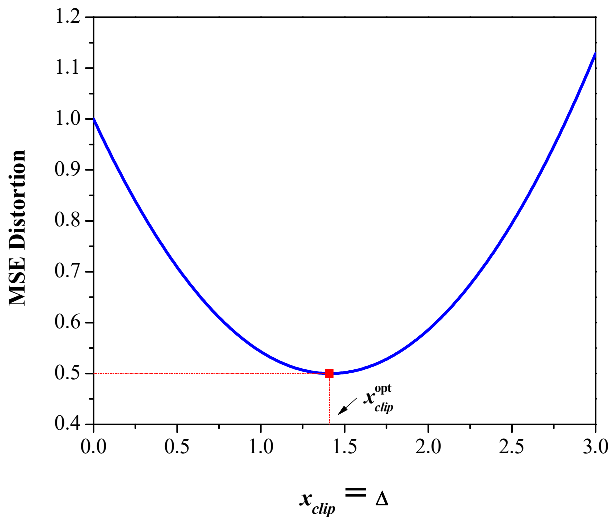

Figure 2. Mean-squared error (MSE) distortion dependence on parameter x_{clip} (Δ) for binary quantizer type 1.

3.2. Binary Quantizer Type 2

The second type of binary scalar quantizer that is extensively used in neural network applications [27,33] is illustrated in Figure 3.

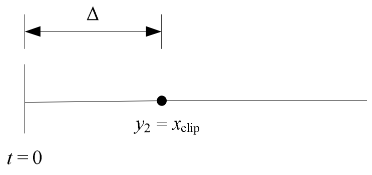

Figure 3. Illustration of binary quantizer type 2.

The representative level in the positive part (0, ∞) was set to $y_2 = x_{clip} = \Delta$, and the following identity holds at $y_1 = -y_2$. The MSE distortion is given as

$$D = 2\int_{0}^{\infty} (x - \Delta)^2 p(x)dx = 1 - \sqrt{2}\Delta + \Delta^2 \tag{9}$$

In terms of x_{clip}, it can be expressed as

$$D = 1 - \sqrt{2}x_{clip} + x_{clip}^2 \tag{10}$$

We can see that the MSE distortion is highly dependent on x_{clip} (or Δ). By optimizing the MSE distortion with respect to x_{clip}, we arrived at

$$\frac{\partial D}{\partial x_{clip}} = 0 \Rightarrow x_{clip} = x_{clip}^{opt} = \frac{1}{\sqrt{2}} \tag{11}$$

This is an intuitively expected result that can be confirmed by directly comparing Equations (3)–(5), (8), and (9), and further verified by numerical simulation, as shown in Figure 4.

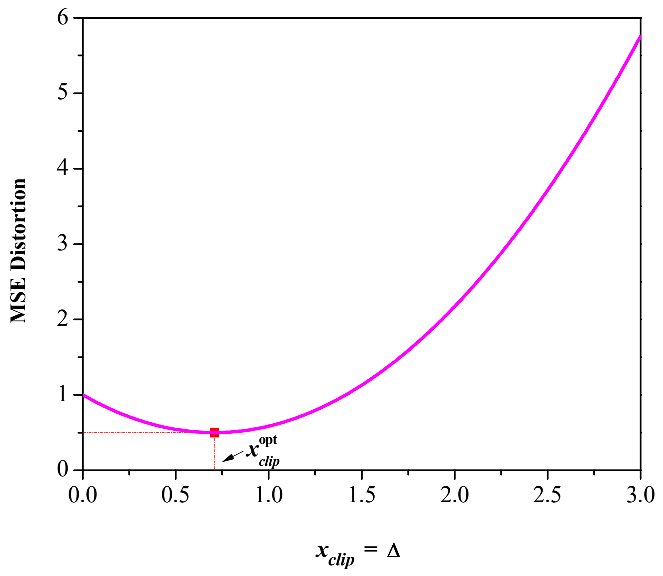

Figure 4. MSE distortion dependence on parameter x_{clip} (Δ) for binary quantizer type 2.

Note that for this specific case (i.e., using the optimal values of x_{clip}), both binary quantizer type 1 and binary quantizer type 2 guarantee the same performance. However, it is also of interest to compare the performances of discussed quantizers for an arbitrary value of x_{clip}, which is investigated in the following subsection.

3.3. Quantizer Performance Evaluation

Figure 5, where the SQNR is plotted as the function of parameter x_{clip}, illustrates the performance of both binary quantizers for a given fixed value x_{clip}. The curves achieve the same maximum SQNR at the optimal values of x_{clip} ($\sqrt{2}$ for type 1 and $1/\sqrt{2}$ for the type 2 quantizer), as expected. If we compare performances for various x_{clip} values, we can perceive that there is a region where the particular quantizer is more efficient. Thus, within the region $x_{clip} \in (0, x_{clip,t})$, where $x_{clip,t} = 0.94$ denotes the point where the curves intersect, binary quantizer type 2 performed better than quantizer type 1. Observe that this range is narrower than the one where binary quantizer type 1 performs better. Observe also that the SQNR values achieved in these ranges are lower than the maximal one, where the SQNR may even take a negative value, meaning that the signal power is lower than the noise power. Finally, we can conclude that if the non-optimal design is necessary to use, a better candidate is binary quantizer type 1.

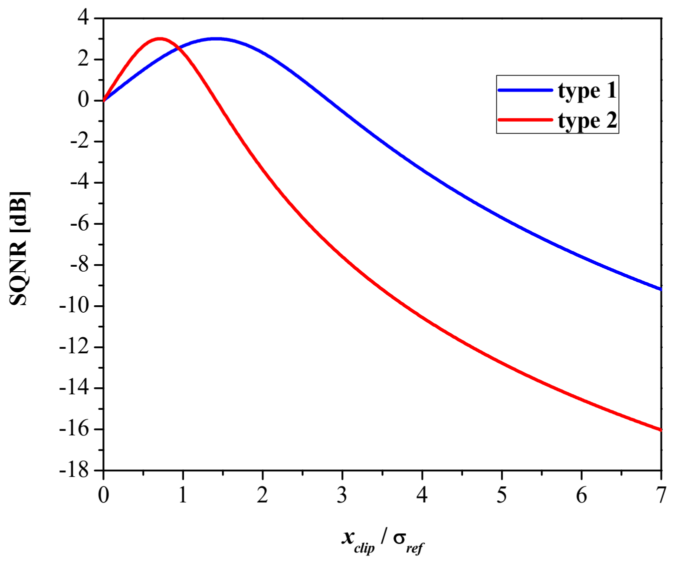

Figure 5. Performance evaluation of binary quantizers type 1 and 2 for a given x_{clip}.

4. Analysis in a Wide Dynamic Range

In the previous section, we designed the binary quantizers for a particular (reference) value of variance and optimized the parameters accordingly. Here, we investigate the performance of optimized quantizers in a scenario where the input data variance differs from the reference one ($\sigma^2 \neq \sigma_{ref}^2 = 1$), or a mismatched quantization [36]. In this case, it is necessary to derive the appropriate expression for performance evaluation using the Laplacian PDF $p(x,\sigma)$ given in Equation (1). Accordingly, for binary quantizer type 1, the following expression for distortion can be derived:

$$D = 2\int_0^\infty \left(x - \frac{x_{clip}(\sigma_{ref})}{2} \right)^2 p(x,\sigma)dx = \sigma^2 - \sigma\frac{x_{clip}(\sigma_{ref})}{\sqrt{2}} + \frac{x_{clip}^2(\sigma_{ref})}{4} \tag{12}$$

where $x_{clip}(\sigma_{ref}) = x_{clip}$ is the value determined for the reference variance σ_{ref}^2.

For binary quantizer type 2, the distortion takes the following form:

$$D = 2\int_0^\infty \left(x - x_{clip}(\sigma_{ref}) \right)^2 p(x,\sigma)dx = \sigma^2 - \sqrt{2}\sigma x_{clip}(\sigma_{ref}) + x_{clip}^2(\sigma_{ref}) \tag{13}$$

Clearly, the SQNR can be estimated using Equation (2).

Figures 6 and 7 depict the SQNR in a wide range of input data variances for the optimal binary quantizer type 1 ($x_{clip} = \sqrt{2}$) and binary quantizer type 2 ($x_{clip} = 1/\sqrt{2}$), respectively. Note that binary quantizers are not robust, since the desired SQNR is attained for the variance matched case ($\sigma^2 = \sigma_{ref}^2$) while, in the rest of the range, the SQNR significantly drops. In these figures, we also include the results for several arbitrary chosen values of x_{clip}, (i.e., $x_{clip} = 2\sigma_{ref}$, $3\sigma_{ref}$, and $4\sigma_{ref}$.) It can be seen that, by increasing x_{clip}, the SQNR curves shift to the right, degrading performance even more. Therefore, quantizers designed for a particular variance may not be useful in non-stationary data processing, such as speech coding applications, and require performance improvement.

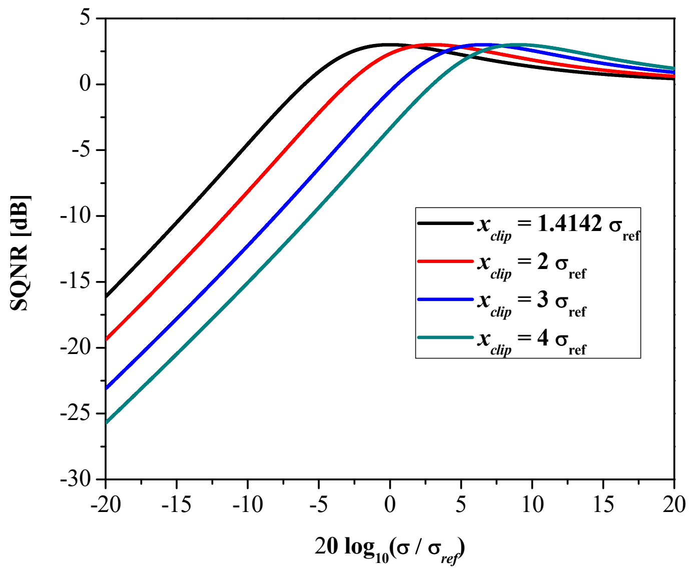

Figure 6. Signal-to-quantization noise ratio (SQNR) as a function of input signal variance for binary quantizer type 1.

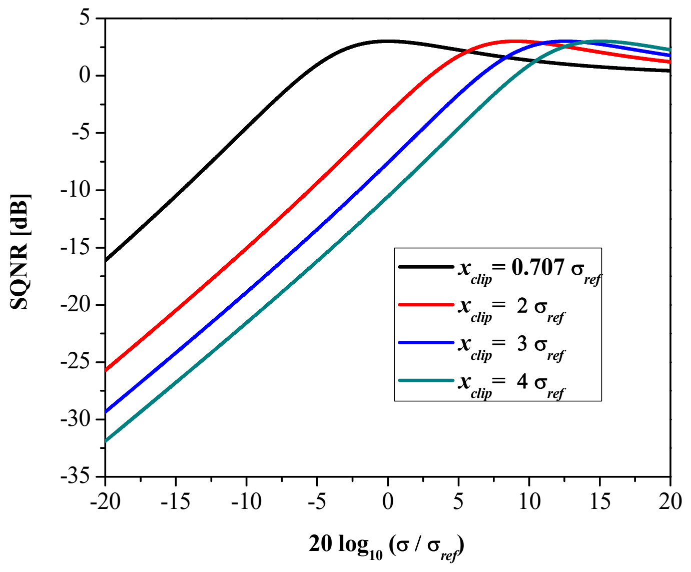

Figure 7. SQNR as a function of input signal variance for binary quantizer type 2.

5. Applications of Binary Quantizer

In this section, we discuss the application of binary quantizers to several research areas, including speech coding, image coding, and neural network compression.

5.1. Speech Coding

Speech belongs to the class of time-varying signals, where the variance tends to change over time [1,3]. Hence, the implementation of a binary quantizer designed for the reference variance to encode the speech may not be the optimal choice, and adaptation is recommended. In this paper, we describe the implementation of a binary quantizer using two frame-wise adaptive techniques (i.e., PCM (Pulse Code Modulation) and ADM).

5.1.1. PCM

The implementation of a binary quantizer in PCM is done using the forward adaptive speech coding algorithm [1,3–5] presented in Figure 8. The forward adaptation technique is introduced to improve the performance of the single quantizer (designed for the particular variance) in a wide dynamic range. The algorithm performs the following steps:

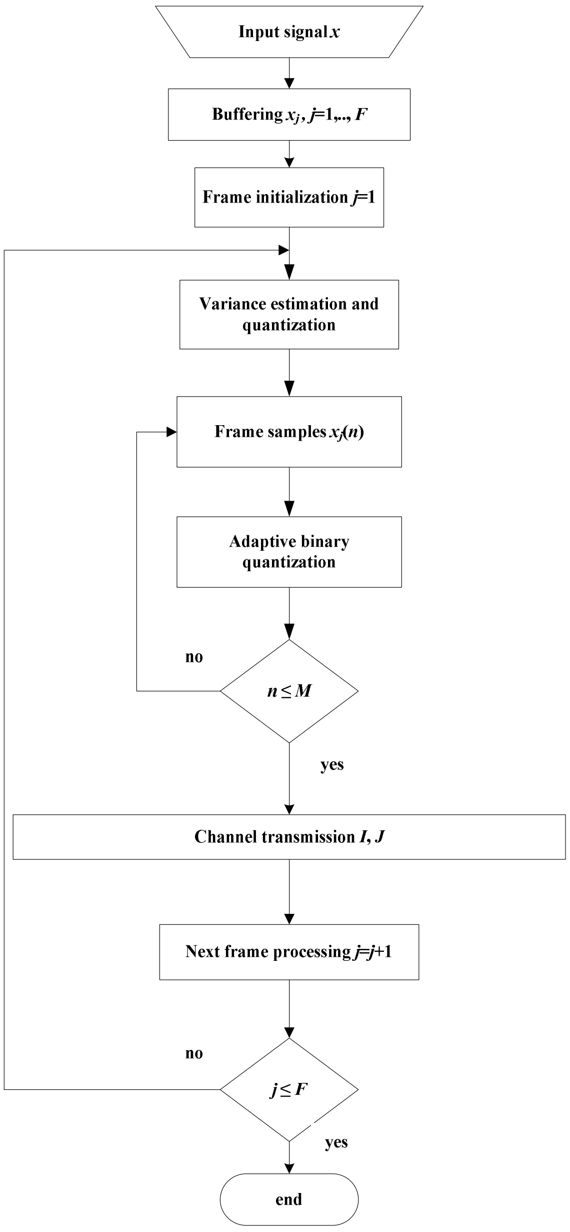
Input signal x
Buffering x_j, j=1,.., F
Frame initialization j=1
Variance estimation and quantization
Frame samples $x_j(n)$
Adaptive binary quantization
no
$n \leq M$
yes
Channel transmission I, J
Next frame processing j=j+1
no
$j \leq F$
yes
end

Figure 8. Pulse code modulation algorithm with a binary quantizer.

Step 1. Buffering. A group of M consecutive samples (i.e., one frame, $x_j(n)$, $n = 1, \ldots, M$, $j = 1, \ldots, F$), is stored within the buffer, where j is the frame index and F is the total number of frames.

Step 2. Variance estimation and quantization. For the stored frame, the variance is estimated by the following equation [1,3–5]:

$$\sigma_j^2 = \frac{1}{M}\sum_{n=1}^{M} x_j^2(n) \tag{14}$$

The log-uniform quantizer is used for variance quantization, which performs uniform quantization in the logarithmic domain [3–5]. In particular, it quantizes the variance V_j (dB) = 10 $\log\sigma_j^2$ to one of L allowed values, defined as

$$V_j = V_k \Big| V_k = V_{\min} + \frac{2k-1}{2}\Delta_L, k = 1, \ldots, L \tag{15}$$

where $\Delta_L = V_{\max} - V_{\min} / L$ denotes the step size and $V_{\max}$ and $V_{\min}$ denote the maximal and minimal estimated variance values. As this information is required at the decoder side, it has to be transmitted once per frame by the index J with $\log_2 L$ bits.

Step 3. Adaptive binary quantization. An adaptive binary quantizer is obtained by multiplying the parameter of the binary quantizer designed for the reference variance value by factor g:

$$y_2^a = g \cdot y_2\left(\sigma_{ref}\right) \tag{16}$$

where g is defined as

$$g = 10^{\frac{V_k}{20}} \tag{17}$$

Each frame sample is quantized using the adaptive binary quantizer, and the output is encoded with a one-bit codeword (index I).

Step 4. Repeat all previous steps until all frames are processed.

Figure 9 depicts the theoretical SQNR (determined by substituting Equation (16) into Equations (12) or (13)) of the forward adaptive binary quantizer for the optimal y_2 value ($y_2 = 1/\sqrt{2}$), and L equals the 32-level log-uniform quantizer. We can see that the adaptive quantizer was robust, as an approximately constant SQNR was achieved in a wide dynamic range.

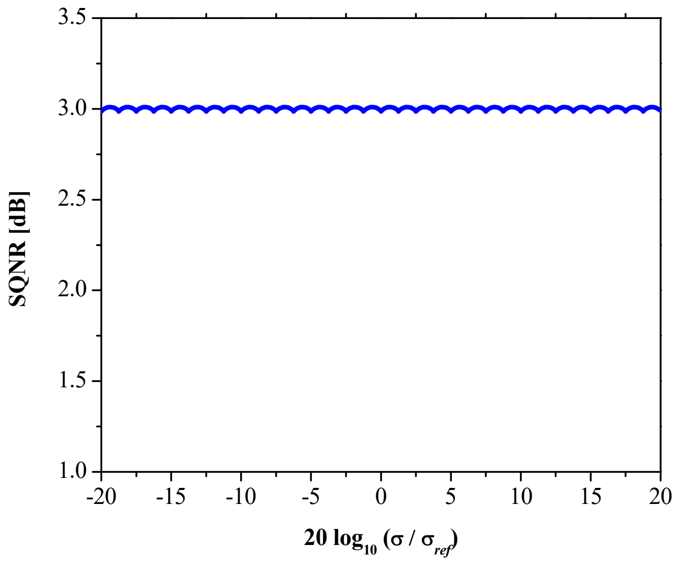

Figure 9. SQNR of the forward adaptive binary quantizer ($y_2 = 1/\sqrt{2}$) in a wide range of input data variances.

The real data experiment was performed with the following sentence: The play seems dull and quite stupid. It was 3 s in length, spoken by a female speaker, sampled at 16 kHz, and extracted from a set of Harvard Psychoacoustic Sentences [37], a collection of phonetically balanced sentences that use specific phonemes with the same frequency of appearance as in the English language. The segmental Signal-to-Noise Ratio (SNR_{seg}) [1,3] was used as the performance measure; that is, the SNR was calculated over each speech frame and then averaged. The frame length was set to 10 ms (160 samples), while we used the log-uniform quantizer with $L = 32$ levels.

Figure 10 shows SQNR over different speech frames for three scenarios. In the first scenario, we use the binary quantizer with optimal representative level $y_2 = 1/\sqrt{2}$(see magenta line). In other two scenarios the performance is investigated for binary quantizer with arbitrary chosen representative levels. Thus, we set x_{clip} as the maximal amplitude of the speech, denoted as x_{max} ($x_{clip} = x_{max} = 0.086$ for the considered speech). This implies $y_2 = x_{clip} = 0.043$ for the binary quantizer type 2 (see red line), and $y_2 = x_{clip}/2 = 0.23$ for the binary quantizer type 1 (see black line). Observe that optimally chosen level ensures the highest performance, while type 2 performs better than type 1 for the established parameter value. If we observe the theoretical results in Figure 5, we can confirm that for the value $x_{clip} = 0.043$ the binary quantizer type 2 is indeed better than type 1. Therefore, we can conclude the experimental results for speech coding follow well the theoretical model.

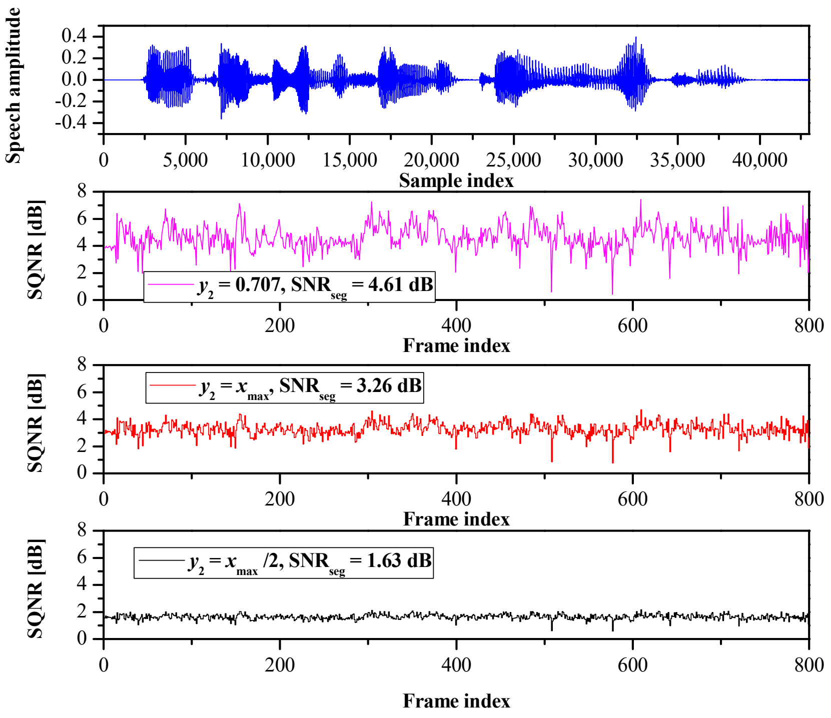

Figure 10. SQNR across speech frames in the case of PCM.

5.1.2. Delta Modulation

The implementation of a binary quantizer in ADM with the first-order linear predictor (both the quantizer and the predictor are forward adaptive) is shown in Figure 11, and can be expressed using these steps:

Step 1. Buffering. This is the same as in Step 1 of the algorithm in Section 5.1.1.

Step 2. Variance estimation and quantization. This is the same as in Step 2 of the algorithm in Section 5.1.1.

Step 3. Estimation of the correlation coefficient and quantization. The correlation coefficient, denoted as ρ, for the current jth frame is estimated as [1,7–9]

$$\rho_j = \frac{1/M \sum_{n=1}^{M-1} x_j(n)x(n+1)}{1/M \sum_{n=1}^{M} x_j^2(n)}, \; j = 1, \ldots, F \tag{18}$$

It is uniformly quantized to one of S available values, given by

$$\rho_j = \rho_k \Big| \rho_k = \rho_{\min} + \frac{2k-1}{2}\Delta_\rho, k = 1, \ldots, S \tag{19}$$

where $\Delta_\rho = \rho_{\max} - \rho_{\min}/S$ denotes the step size and $\rho_{\max}$ and $\rho_{\min}$ denote the maximal and minimal estimated values of the correlation coefficient. This information is also required at the decoder side, and it has to be transferred once per frame by the index K with $\log_2 S$ bits.

Step 4. Determination of the prediction error. For the jth frame, the prediction error can be determined according to

$$e_j(n) = x_j(n) - x_p(n), k = 1, \ldots, L \tag{20}$$

where $x_p(n) = \rho_k \cdot y_j(n-1)$ is the predicted sample value and $y_j(n)$ is the reconstructed value:

$$y_j(n) = x_p(n) + e_q(n), k = 1, \ldots, L \tag{21}$$

where $e_q(n)$ is the quantized value of $e_j(n)$.

Step 5. Adaptive binary quantization. For the jth frame, the scaling factor is given by

$$g^{DM} = g\sqrt{1-\rho_k^2}, k = 1, \ldots, L \tag{22}$$

where g is defined by Equation (17) and ρ_k is given by Equation (19). The adaptive representative level is obtained as in Equation (16). The prediction error signal was quantized using the adaptive binary quantizer and the output was encoded with a one-bit codeword (index I).

Step 6. Repeat all previous steps until all frames are processed.

A real data experiment was performed using the same speech signal, frame length, and quantizer parameters as in the case of PCM. In addition, both the log-uniform quantizer (for the frame variance) and the uniform quantizer (for the correlation coefficient) used 32 levels.

Figure 12 plots the SNR over different speech frames for ADM. Similar conclusions can be drawn as in the case of PCM; that is, binary quantizer type 2 performed better than type 1 for the established parameter value. We also observed that ADM (Figure 12) gave a higher performance for both voiced and unvoiced frames in comparison with PCM (Figure 10).

5.2. Image Coding

The application of a binary quantizer in image coding was analyzed using a BTC algorithm, which we briefly describe.

The algorithm starts by dividing the input picture into a set of non-overlapping pixel blocks of size $m \times m$. For each block, the mean value, denoted as ${x_{av}}^k$, where k is index of the block, is calculated as

$$x_{av}^k = \frac{1}{m \times m}\sum_{i=1}^{m}\sum_{j=1}^{m} x(i,j) \tag{23}$$

where $x(i, j)$ is the pixel intensity, i is the row index, and j is the column index. The mean value is uniformly quantized according to

$$x_{av}^{k} = x_{av,q} \Big| x_{av,q} = x_{av,\min} + \frac{2n-1}{2}\Delta_{av}, n = 1, \ldots, S_1 \tag{24}$$

where $x_{av,\min} = 0$, $x_{av,\max} = 255$, and $\Delta_{av} = (x_{av,\max} - x_{av,\min})/S_1 = 255/S_1$. For encoding, we used $r_{av} = \log_2 S_1$ bits.

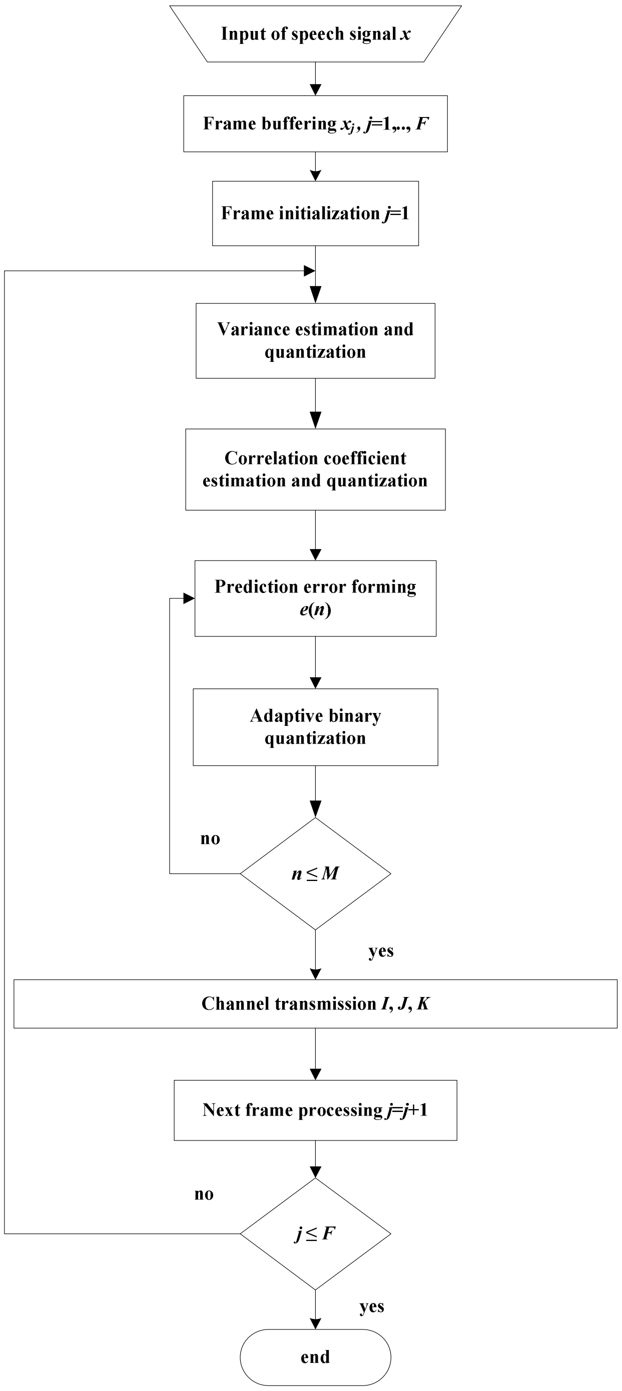
Input of speech signal x
Frame buffering x_j, j=1,.., F
Frame initialization j=1
Variance estimation and quantization
Correlation coefficient estimation and quantization
Prediction error forming $e(n)$
Adaptive binary quantization
no
$n \leq M$
yes
Channel transmission I, J, K
Next frame processing j=j+1
no
$j \leq F$
yes
end

Figure 11. Adaptive delta modulation algorithm with a binary quantizer.

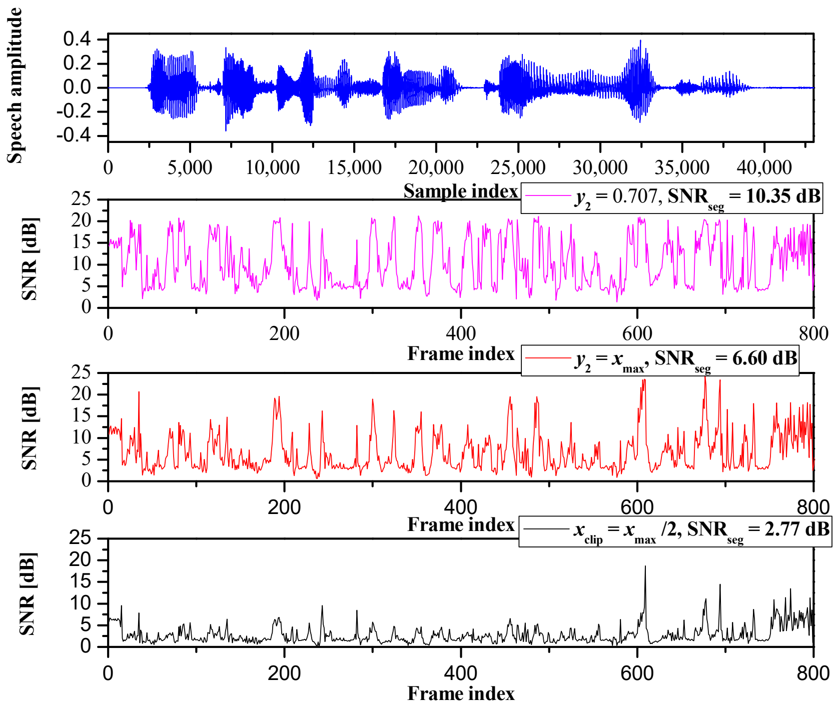

Figure 12. SQNR across speech frames in the case of adaptive delta modulation (ADM).

In the next step, for the kth block, the mean value is subtracted from the original pixel intensities, $x_d(i,j) = x(i,j) - x_{av}{}^k$, and the block with new values is obtained. It actually represents an input to the binary quantizer. To perform efficient quantization, we had to adjust the quantizer to the local block statistics. Thus, for the kth block we estimate the variance as follows:

$$\sigma_k^2 = \frac{1}{m \times m} \sum_{i=1}^{m} \sum_{j=1}^{m} (x_d(i,j))^2 \tag{25}$$

As in previous subsections, we define the binary quantizer for kth block as

$$y_2^k = \sigma_k \cdot y_2\left(\sigma_{ref}\right) \tag{26}$$

Furthermore, the estimated block variance was uniformly quantized and encoded using r_σ bits. A particular block value $x_d(i,j)$ was binarily quantized by the value $x_d{}^q(i,j) \in (-y_2{}^k,\ y_2{}^k)$. The reconstructed pixel intensity for the kth block, denoted as $x_r(i,j)$, can be obtained with

$$x_r(i,j) = x_d^q(i,j) + x_{av,q} \tag{27}$$

The real data experiment was done using the standard test grayscale image of Lena of the size 512 × 512 pixels, which belongs to the USC-SIPI Image Database [38]. The bit rate R and peak signal-to-noise ratio (PSQNR) were used as performance measures [1,12,13]. The results for the BTC algorithm, with two different block sizes (4 × 4 and 8 × 8) and different r_{av} and r_σ values by adopting the optimal binary quantizer ($y_2 = 1/\sqrt{2}$), are summarized in Table 1. It can be seen that a high PSQNR was obtained with satisfactory low bit rates.

Moreover, in Table 2, we present the results in terms of the PSQNR for the BTC algorithm with a binary quantizer, using the optimal and non-optimal (i.e., arbitrarily selected) representative levels for the block size of 4 × 4. Thus, we adopted y_2 = 1.5 and y_2 = 3, which correspond to binary quantizer type 1 and 2, respectively. As expected, BTC performance was degraded when it used the non-optimal quantizer. However, we can see that a better PSQNR was provided when binary quantizer type 1 was employed. This is also in accordance with the theoretical results in Figure 5 since, for the same value of $x_{clip} = 3$, binary quantizer type 1 performed better than type 2.

Table 1. Block truncation coding algorithm with the optimal binary quantizer ($y_2 = 1/\sqrt{2}$) applied to the monochrome image of Lena.

		Block			***Block***	
		4 × 4			**8 × 8**	
r_σ	8	5	4	8	5	4
r_{sr}	8	5	4	8	5	4
PSQNR (dB)	32.06	31.84	31.37	28.59	28.47	28.18
***R* (bpp)**	2	1.5	1.625	1.25	1.16	1.125

Table 2. Performance comparison of the block truncation coding algorithm using optimal and non-optimal binary quantizers, applied to the monochrome image of Lena.

Block		**4 × 4**	
r_σ	8	8	8
r_{sr}	8	8	8
y_2	$1/\sqrt{2}$	1.5	3
PSQNR (dB)	32.06	28.69	20.87
***R* (bpp)**	2	2	2

In Figure 13, we depict the reconstructed images in the cases of an optimal ($y_2 = 1/\sqrt{2}$) and non-optimal level ($y_2 = 3$) for $m = 4$, $r_{av} = 8$ bits/block, and $r_\sigma = 8$ bits/block, showing better quality of reconstruction and reduced granular noise in the case of an optimally chosen level and confirming the above conclusions. Note that BTC with a binary quantizer achieved a compression ratio equal to 4, with respect to the original image (8 bits/pixel).

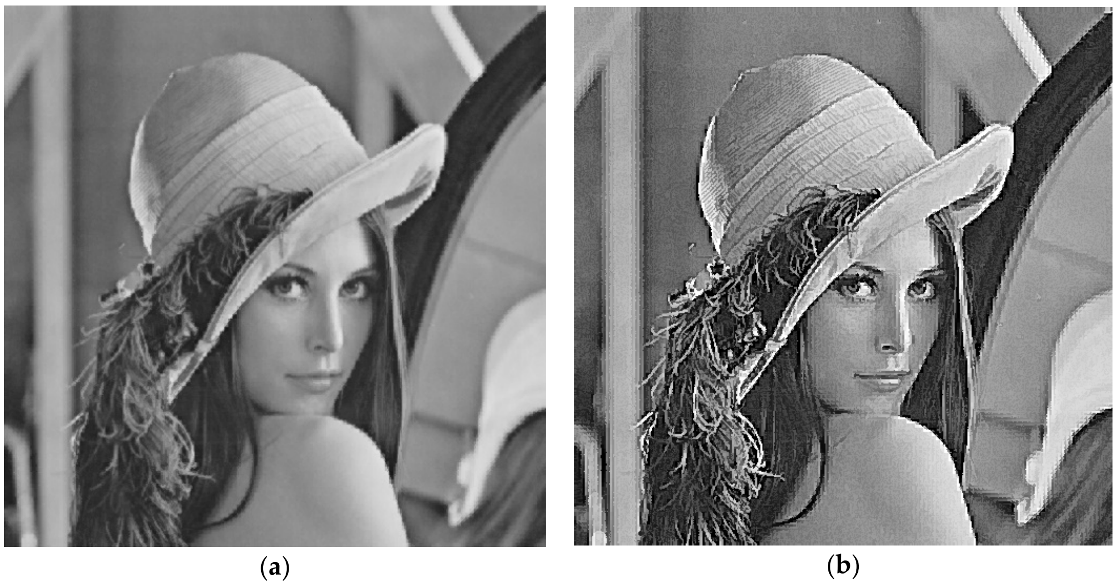

Figure 13. The reconstructed image for block truncation coding (BTC) ($m = 4$, $r_{av} = 8$ bits/block, and $r_\sigma = 8$ bits/block) with (**a**) an optimal binary quantizer ($y_2 = 1/\sqrt{2}$) and (**b**) a non-optimal binary quantizer ($y_2 = 3$).

5.3. Neural Networks Compression

In this paper, the multilayer perceptron (MLP) neural network was employed to investigate the influence of a binary quantizer on performance, measured by prediction accuracy. As elaborated in Section 2, the motivation behind the binary quantization of neural network parameters was to provide model compression when compared to the full precision case, which is crucial for implementation in portable and edge computing devices with limited memory and processing power. MLP is a class of feedforward artificial NNs that consists of three layers: An input layer, a hidden layer, and an output layer. Our goal was to apply the quantization to learned network weights (post-trained quantization).

Training data was taken from the MNIST database [39], which contains 60,000 monochrome images of handwritten single digits sized 28×28 pixels. The network input was the image vector of a size $28 \times 28 = 784$ pixels, and the number of hidden layer nodes was set to 128, whereas the output layer had 10 nodes and corresponded to the number of digits. Rectified linear unit (ReLU) and softmax activation functions were used in the hidden and output layers, respectively. The hyperparameters of the MLP neural network were as follows: Regularization rate (L2) = 0.01, learning rate = 0.0005, and batch size = 128.

Figure 14 plots the training and validation accuracy, where the model was evaluated on a hold-out validation dataset after each epoch during the training. We can observe that the neural network did not overfit, and the model converged after 20 epochs, achieving a prediction accuracy of 96.7%.

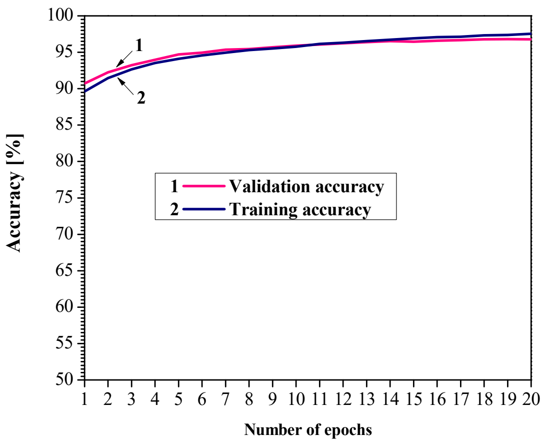

Figure 14. Learning curves for the considered multiplayer perceptron (MLP) neural network.

Binary quantization was then applied to the learned neural network weights, which were given in the matrix form of dimensions 784×128 for the weights between the input and hidden layers and 128×10 for the weights between hidden and output layers. Distribution of the learned weights between the input and the hidden layers is illustrated in Figure 15, showing that the Laplacian PDF can be used as a good model.

To perform effective quantization, one should adapt the quantizer to the statistics of the input matrix. Hence, the mean value was estimated and subtracted from the original network weights. Information about the mean value of the data was stored in full precision format (32-bit floating point). Furthermore, the standard deviation of data was estimated, and the binary quantizer was adapted (scaled) accordingly. This information was also stored in full precision format. Mean-normalized network weights were quantized using the adaptive binary quantizer. In order to reconstruct the original network weights, the mean value needed to be added to the quantized mean-normalized network weights.

Table 3 gives the achieved accuracies for the MLP neural network with (using a binary quantizer with optimal and non-optimal representative levels) and without (denoted as full precision) quantization of the weights. Namely, the non-optimal representative levels of the binary quantizer are specified as $x_{max}/2$ (binary quantizer type 1) and x_{max} (binary quantizer type 2), where x_{max} denotes the maximal data value in the network weight matrix (x_{max} = 0.45). The SQNR measure, indicating the efficiency of the observed binary models on the available real data, is also provided in Table 3.

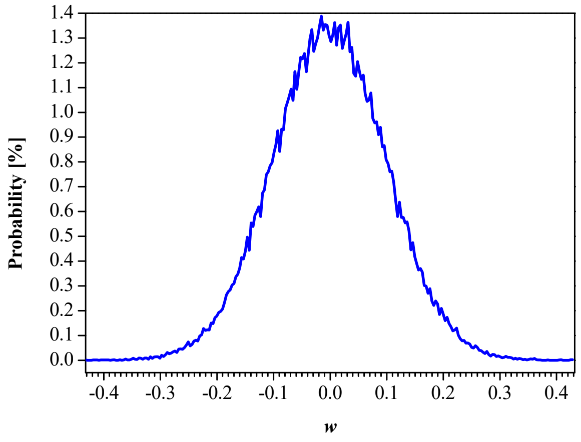

Figure 15. Distribution of learned weights for the considered MLP neural network.

Table 3. Prediction accuracy of the multilayer perceptron neural network for different representation levels of the binary quantizer.

	Y_2			Full Precision
	$1/\sqrt{2}$	x_{max}/2 **(Type 1)**	x_{max} **(Type 2)**	
Accuracy (%)	91.28	81.66	89.96	96.70
SQNR (dB)	4.287	1.636	3.205	-

It can be seen that the highest values for both performance measures, prediction accuracy (91.28%) and SQNR (4.287 dB), were attained in the case of the optimal binary quantizer ($y_2 = 1/\sqrt{2}$). In comparison with the network weights stored in the full precision format, the accuracy dropped by approximately 5%, but the compression ratio was substantial and amounted to 32. Therefore, the decreased performance can be compensated by large compression, which may be crucial for implementation in devices with limited memory and processing power. It is also interesting to note that the non-optimal binary model used so far [27,33], which applies the maximal data value as the representative level (type 2), had a lower accuracy of approximately 1.3%.

Eventually, considering the analysis for the arbitrary chosen level, Table 3 reveals, based on the SQNR values, that quantizer type 2 is more efficient than quantizer type 1. This can be also validated by the theoretical model in Figure 5 (see performance for x_{clip} = 0.45).

6. Conclusions

This paper has addressed a binary scalar quantizer for data with a Laplacian PDF, with applications in speech and image coding, as well as the compression of neural networks. Two types of binary quantizers have been designed, taking into account the effect of clipping and the determination of the optimal clipping factor. Detailed performance analysis has shown that both quantizers provide the same maximal SQNR in an optimal setting. However, in cases where non-optimal design is necessary, binary quantizer type 1 is more efficient than binary quantizer type 2 for image compression, leading to better quality of the reconstructed image and reduced granular noise. Contrary to this finding, quantizer type 2 is more efficient for the compression of speech signal and neural network parameters.

Theoretical analysis of the considered models in a wide dynamic range has also been conducted, showing that their robustness is low, making them inefficient for non-stationary data. Therefore, the forward adaptive model has been introduced for non-stationary data. Several real-world data have been used to test the adaptive model, including speech, images, and the quantization of neural network weights, confirming the applicability of the proposed models and, furthermore, showing excellent matching between the experimental and theoretical results.

Future research will include the application of the introduced adaptive methods to compression of not only neural network weights, but also activations or even gradients and, furthermore, to quantization of state-of-the-art deep neural networks, such as AlexNet, GoogleNet, ResNet, or VGG.

Author Contributions: Conceptualization and supervision, Z.P.; data curation and writing—original draft preparation, B.D.; software and validation, M.S.; writing—review and editing, V.D. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been supported by the Ministry of Education, Science and Technological Development of the Republic of Serbia and by the Science Found of the Republic of Serbia (Grant No. 6527104, AI-Com-in-AI).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Jayant, N.C.; Noll, P. *Digital Coding of Waveforms: Principles and Applications to Speech and Video*; Prentice Hall: Englewood Cliffs, NJ, USA, 1984.
2. Gersho, A.; Gray, R. *Vector Quantization and Signal Compression*; Kluwer Academic Publishers: New York, NY, USA, 1992.
3. Chu, W.C. *Speech Coding Algorithms: Foundation and Evolution of Standardized Coders*; John Wiley & Sons: Hoboken, NJ, USA, 2003.
4. Peric, Z.; Nikolic, J.; Denic, B.; Despotovic, V. Forward adaptive dual-mode quantizer based on the first-degree spline approximation and embedded G.711 codec. *Radioengineering* **2019**, *28*, 729–739. [CrossRef]
5. Nikolic, J.; Peric, Z. Lloyd-Max's algorithm implementation in speech coding algorithm based on forward adaptive technique. *Informatica* **2008**, *19*, 255–270. [CrossRef]
6. Prosalentis, E.A.; Tombras, G.S. 2-bit adaptive delta modulation system with improved performance. *EURASIP J. Adv. Signal Proc.* **2007**, *2006*, 16286. [CrossRef]
7. Peric, Z.; Denic, B.; Despotovic, V. Novel two-bit adaptive delta modulation algorithms. *Informatica* **2019**, *30*, 117–134. [CrossRef]
8. Peric, Z.; Denic, B.; Despotovic, V. An efficient two-digit adaptive delta modulation for Laplacian source coding. *Int. J. Elect.* **2019**, *106*, 1085–1100. [CrossRef]
9. Denic, B.; Peric, Z.; Despotovic, V. Three-level delta modulation for Laplacian source coding. *Adv. Elect. Comp. Eng.* **2017**, *17*, 95–102. [CrossRef]
10. Delp, E.J.; Saenz, M.; Salama, P. Block Truncation Coding (BTC). In *Handbook of Image and Video Processing*; Elsevier Academic Press: San Diego, CA, USA, 2005; pp. 661–670.
11. Jiang, M.; Yang, H. Secure outsourcing algorithm of BTC feature extraction in cloud computing. *IEEE Access* **2020**, *8*, 106958–106967. [CrossRef]
12. Simic, N.; Peric, Z.; Savic, M. Coding algorithm for grayscale images—Design of piecewise uniform quantizer with Golomb-Rice code and novel analytical model for performance analysis. *Informatica* **2017**, *28*, 703–724. [CrossRef]
13. Savic, M.; Peric, Z.; Dincic, M. Coding algorithm for grayscale images based on piecewise uniform quantizers. *Informatica* **2012**, *23*, 125–140. [CrossRef]
14. Huang, K.; Ni, B.; Yang, X. Efficient quantization for neural networks with binary weights and low bitwidth activations. In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19), Honolulu, HI, USA, 27 January–1 February 2019.
15. Nia, V.P.; Belbahr, M. Binary quantizer. *J. Comput. Vis. Imaging Syst.* **2018**, *4*, 3.
16. Qina, H.; Gonga, R.; Liu, X.; Baie, X.; Songc, J.; Sebed, N. Binary neural networks: A survey. *arXiv* **2020**, arXiv:2004.03333. [CrossRef]
17. Pouransari, H.; Tu, Z.; Tuzel, O. Least squares binary quantization of neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 14−19 June 2020.

18. Courbariaux, M.; Hubara, I.; Soudry, D.; El-Yaniv, R.; Bengio, Y. Binarized neural networks: Training neural networks with weights and activations constrained to +1 or −1. *arXiv* **2016**, arXiv:1602.02830.
19. Hubara, I.; Courbariaux, M.; Soudry, D.; El-Yaniv, R.; Bengio, Y. Binarized neural networks. In Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS), Barcelona, Spain, 5−10 December 2016.
20. Simons, T.; Lee, D.J. A review of binarized neural networks. *Electronics* **2019**, *8*, 661. [CrossRef]
21. Darabi, S.; Belbahri, M.; Courbariaux, M.; Nia, V.P. Regularized binary network training. *arXiv* **2018**, arXiv:1812.11800.
22. Gazor, S.; Zhang, W. Speech probability distribution. *IEEE Signal Proc. Lett.* **2003**, *10*, 204–207. [CrossRef]
23. Banner, R.; Nahshan, Y.; Hoffer, E.; Soudry, D. ACIQ: Analytical clipping for integer quantization of neural networks. *arXiv* **2018**, arXiv:1810.05723.
24. Banner, R.; Nahshan, Y.; Soudry, D. Post training 4-bit quantization of convolutional networks for rapid-deployment. In Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS), Vancouver, BC, Canada, 8−10 December 2019.
25. Zrilic, D.G. *Circuits and Systems Based on Delta Modulation*; Springer: Berlin/Heidelberg, Germany, 2005.
26. Gibson, J.D. Speech compression. *Information* **2016**, *7*, 32. [CrossRef]
27. Jacob, B.; Kligys, S.; Chen, B.; Zhu, M.; Tang, M.; Howard, A.; Adam, H.; Kalenichenko, D. Quantization and training of neural networks for efficient integer-arithmetic-onlyinference. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18−23 June 2018.
28. Gong, J.; Shen, H.; Zhang, G.; Liu, X.; Li, S.; Jin, G.; Maheshwari, N.; Fomenko, E.; Segal, E. Highly efficient 8-bit low precision inference of convolutional neural networks with IntelCaffe. *arXiv* **2018**, arXiv:1805.08691.
29. Krishnamoorthi, R. Quantizing deep convolutional networks for efficient inference: Awhitepaper. *arXiv* **2018**, arXiv:1806.08342.
30. McKinstry, J.L.; Esser, S.K.; Appuswamy, R.; Bablani, D.; Arthur, J.V.; Yildiz, I.B.; Modha, D.S. Discovering low-precision networks close to full-precision networks for efficient embedded inference. *arXiv* **2018**, arXiv:1809.04191.
31. Choi, J.; Wang, Z.; Venkataramani, S.; Chuang, P.I.; Srinivasan, V.; Gopalakrishnan., K. Pact: Parameterized clipping activation for quantized neural networks. *arXiv* **2018**, arXiv:1805.06085.
32. Ullah, I.; Manzo, M.; Shah, M.; Madden, M. Graph Convolutional Networks: Analysis, improvements and results. *arXiv* **2019**, arXiv:1912.09592.
33. Hubara, I.; Courbariaux, M.; Soudry, D.; El-Yaniv, R.; Bengio, Y. Quantized neural networks: Training neural networks with low precision weights and activations. *J. Mach. Learn. Res.* **2018**, *18*, 1–30.
34. Tkachenko, R.; Izonin, I.; Kryvinska, N.; Dronyuk, I.; Zub, K. An approach towards increasing prediction accuracy for the recovery of missing IoT data based on the GRNN-SGTM ensemble. *Sensors* **2020**, *20*, 2625. [CrossRef] [PubMed]
35. Tkachenko, R.; Izonin, I. Model and principles for the implementation of neural-like structures based on geometric data transformations. In Proceedings of the International Conference on Computer Science (ICCSEEA 2018) AISC Series; Springer: Cham, Switzerland, 2019; Volume 754, pp. 578–587.
36. Na, S. Asymptotic formulas for mismatched fixed-rate minimum MSE Laplacian quantizers. *IEEE Signal Proc. Lett.* **2008**, *15*, 13–16.
37. Demonte, P.; HARVARD Speech Corpus—Audio Recording 2019. University of Salford Collection. 2019. Available online: https://doi.org/10.17866/rd.salford.c.4437578.v1 (accessed on 1 September 2020).
38. The USC-SIPI Image Database. Available online: http://sipi.usc.edu/database (accessed on 1 September 2020).
39. Lecun, Y.; Cortez, C.; Burges, C. The MNIST Handwritten Digit Database. Available online: yann.lecun.com (accessed on 1 September 2020).