symmetry

MDPI

Article

An Efficient VQ Codebook Search Algorithm Applied to AMR-WB Speech Coding

Cheng-Yu Yeh

Department of Electrical Engineering, National Chin-Yi University of Technology, 57, Section 2, Zhongshan Road, Taiping District, Taichung 41170, Taiwan; cy.yeh@ncut.edu.tw; Tel.: +886-4-23924505 (ext. 7236)

Academic Editor: Vladimir Shpilrain
Received: 6 March 2017; Accepted: 10 April 2017; Published: 12 April 2017

Abstract: The adaptive multi-rate wideband (AMR-WB) speech codec is widely used in modern mobile communication systems for high speech quality in handheld devices. Nonetheless, a major disadvantage is that vector quantization (VQ) of immittance spectral frequency (ISF) coefficients takes a considerable computational load in the AMR-WB coding. Accordingly, a binary search space-structured VQ (BSS-VQ) algorithm is adopted to efficiently reduce the complexity of ISF quantization in AMR-WB. This search algorithm is done through a fast locating technique combined with lookup tables, such that an input vector is efficiently assigned to a subspace where relatively few codeword searches are required to be executed. In terms of overall search performance, this work is experimentally validated as a superior search algorithm relative to a multiple triangular inequality elimination (MTIE), a TIE with dynamic and intersection mechanisms (DI-TIE), and an equal-average equal-variance equal-norm nearest neighbor search (EEENNS) approach. With a full search algorithm as a benchmark for overall search load comparison, this work provides an 87% search load reduction at a threshold of quantization accuracy of 0.96, a figure far beyond 55% in the MTIE, 76% in the EEENNS approach, and 83% in the DI-TIE approach.

Keywords: vector quantization (VQ); binary search space-structured VQ (BSS-VQ); immittance spectral frequency (ISF); adaptive multi-rate wideband (AMR-WB); speech codec; VoIP

1. Introduction

With a 16 kHz sampling rate, the adaptive multi-rate wideband (AMR-WB) speech codec [1–4] is one of the speech codecs applied to modern mobile communication systems as a way to remarkably improve the speech quality on handheld devices. The AMR-WB is a speech codec developed on the basis of an algebraic code-excited linear-prediction (ACELP) coding technique [4,5], and provides nine coding modes with bitrates of 23.85, 23.05, 19.85, 18.25, 15.85, 14.25, 12.65, 8.85, and 6.6 kbps. The ACELP-based technique is developed as an excellent speech coding technique, having a double advantage of low bit rates and high speech quality, but a price paid is a high computational complexity required in an AMR-WB codec. Using an AMR-WB speech codec, the speech quality of a smartphone can be improved but at the cost of high battery power consumption.

In an AMR-WB encoder, a vector quantization (VQ) of immittance spectral frequency (ISF) coefficients [6–10] occupies a significant computational load in various coding modes. The VQ structure in AMR-WB adopts a combination of a split VQ (SVQ) and a multi-stage VQ (MSVQ) techniques, referred to as split-multistage VQ (S-MSVQ), to quantize the 16-order ISF coefficient [1]. Conventionally, VQ uses a full search to obtain a codeword best matched with an arbitrary input vector, but the full search requires an enormous computational load. Therefore, many studies [11–19] have made efforts to simplify the search complexity of an encoding process. These approaches are classified into two types in terms of the manner the complexity is reduced: triangular inequality elimination

(TIE)-based approaches [11–14] and equal-average equal-variance equal-norm nearest neighbor search (EEENNS)-based algorithms [15–19].

As in [11], a TIE algorithm is proposed to address the computational load issue in a VQ-based image coding. Improved versions of TIE approaches are presented in [12,13] to reduce the scope of a search space, giving rise to further reduction in the computational load. However, there exists a high correlation between ISF coefficients of neighboring frames in AMR-WB, that is, ISF coefficients evolve smoothly over successive frames. This feature benefits TIE-based VQ encoding, according to which a considerable computational load reduction is demonstrated. Yet, a moving average (MA) filter is employed to smooth the data in advance of VQ encoding of ISF coefficients. It means that the high correlation feature is gone, resulting in a poor performance in computational load reduction. Recently, a TIE algorithm equipped with a dynamic and an intersection mechanism, named DI-TIE, is proposed to effectively simplify the search load, and this algorithm is validated as the best candidate among the TIE-based approaches so far. On the other hand, an EEENNS algorithm was derived from equal-average nearest neighbor search (ENNS) and equal-average equal-variance nearest neighbor search (EENNS) approaches [15–19]. In contrast to TIE-based approaches, the EEENNS algorithm uses three significant features of a vector, i.e., mean, variance, and norm, as a three-level elimination criterion to reject impossible codewords.

Furthermore, a binary search space-structured VQ (BSS-VQ) is presented in [20] as a simple as well as efficient way to quantize line spectral frequency (LSF) coefficients in the ITU-T G.729 speech codec [5]. This algorithm demonstrated that a significant computational load reduction is achieved with a well maintained speech quality. In view of this, this paper will apply the BSS-VQ search algorithm to the ISF coefficients quantization in AMR-WB. This work aims to verify whether the performance superiority of the BSS-VQ algorithm remains, for the reason that the VQ structure in AMR-WB is different from that in G.729. On the other hand, another major motivation behind this is to meet the energy saving requirement on handheld devices, e.g. smartphones, for an extended operation time period.

The rest of this paper is outlined as follows. Section 2 gives the description of ISF coefficients quantization in AMR-WB. The BSS-VQ algorithm for ISF quantization is presented in Section 3. Section 4 demonstrates experimental results and discussions. This work is summarized at the end of this paper.

2. ISF Coefficients Quantization in AMR-WB

In a quantization process of AMR-WB [1], a speech frame of 20 ms is firstly applied to evaluate linear predictive coefficients (LPCs), which are then converted into ISF coefficients. Subsequently, quantized ISF coefficients are obtained following a VQ encoding process, which is detailed below.

2.1. Linear Prediction Analysis

In a linear prediction, a Levinson–Durbin algorithm is used to compute a 16th order LPC, a_i, of a linear prediction filter [1], defined as:

$$\frac{1}{A(z)} = \frac{1}{1 + \sum_{i=1}^{16} a_i z^{-i}}. \tag{1}$$

Subsequently, the LPC parameters are converted into the immittance spectral pair (ISP) coefficients for the purposes of parametric quantization and interpolation. The ISP coefficients are defined as the roots of the following two polynomials:

$$F_1'(z) = A(z) + z^{-16}A(z^{-1}) \tag{2}$$

$$F_2'(z) = A(z) - z^{-16}A(z^{-1}). \tag{3}$$

$F'_1(z)$ and $F'_2(z)$ are symmetric and antisymmetric polynomials, respectively. It can be proven that all the roots of such two polynomials lie and alternate successively on a unit circle in the z-domain. Additionally, $F'_2(z)$ has two roots at $z = 1$ ($\omega = 0$) and $z = -1$ ($\omega = \pi$). Such two roots are eliminated by introducing the following polynomials, with eight and seven conjugate roots respectively on the unit circle, expressed as:

$$F_1(z) = F'_1(z) = (1 + a[16]) \prod_{i=0,2,\ldots,14} (1 - 2q_i z^{-1} + z^{-2}) \tag{4}$$

$$F_2(z) = \frac{F'_2(z)}{(1 - z^{-2})} = (1 - a[16]) \prod_{i=1,3,\ldots,13} (1 - 2q_i z^{-1} + z^{-2}) \tag{5}$$

where the coefficients q_i are referred to as the ISPs in the cosine domain, and $a[16]$ is the last predictor coefficient. A Chebyshev polynomial is used to solve Equations (4) and (5). Finally, 16th order ISF coefficients ω_i can be obtained by taking the transformation $\omega_i = \arccos(q_i)$.

2.2. *Quantization of ISF Coefficients*

Before a quantization process, a mean-removed and first order MA filtering are performed on the ISF coefficients to obtain a residual ISF vector [1], that is:

$$\mathbf{r}(n) = \mathbf{z}(n) - \mathbf{p}(n) \tag{6}$$

where $\mathbf{z}(n)$ and $\mathbf{p}(n)$ respectively denote the mean-removed ISF vector and the predicted ISF vector at frame n by a first order MA prediction, defined as:

$$\mathbf{p}(n) = \frac{1}{3}\hat{\mathbf{r}}(n-1) \tag{7}$$

where $\hat{\mathbf{r}}(n-1)$ is the quantized residual vector at the previous frame.

Subsequently, S-MSVQ is performed on $\mathbf{r}(n)$. As presented in Tables 1 and 2, S-MSVQ is categorized into two types in terms of the bit rate of the coding modes. In Stage 1, $\mathbf{r}(n)$ is split into two subvectors, namely, a 9-dimensional subvector $\mathbf{r}_1(n)$ associated with codebook CB1 and a 7-dimensional subvector $\mathbf{r}_2(n)$ associated with codebook CB2, for VQ encoding. As a preliminary step of Stage 2, the quantization error vectors are split into three subvectors for the 6.60 kbps mode or five for the modes with bitrates between 8.85 and 23.85 kbps, symbolized as $\mathbf{r}_i^{(2)} = \mathbf{r}_i - \hat{\mathbf{r}}_i, i = 1, 2$, respectively. For instance, $\mathbf{r}^{(2)}{}_{1,1\text{–}3}$ in Table 1 represents the subvector split from the 1st to the 3rd components of $\mathbf{r}_1$, and then VQ encoding is performed thereon over codebook CB11 in Stage 2. Likewise, $\mathbf{r}^{(2)}{}_{2,4\text{–}7}$ stands for the subvector split from the 4th to the 7th components of $\mathbf{r}_2$, after which VQ encoding is performed over codebook CB22 in Stage 2. Finally, a squared error ISF distortion measure, that is, Euclidean distance, is used in all quantization processes.

Table 1. Structure of S-MSVQ in AMR-WB in the 8.85–23.85 kbps coding modes.

Structure of S-MSVQ					
Stage 1	CB1: $\mathbf{r}_1$ (1–9 order of $\mathbf{r}$) (8 bits)			CB2: $\mathbf{r}_2$ (10–16 order of $\mathbf{r}$) (8 bits)	
Stage 2	CB11: $\mathbf{r}^{(2)}{}_{1,1\text{–}3}$ (6 bits)	CB12: $\mathbf{r}^{(2)}{}_{1,4\text{–}6}$ (7 bits)	CB13: $\mathbf{r}^{(2)}{}_{1,7\text{–}9}$ (7 bits)	CB21: $\mathbf{r}^{(2)}{}_{2,1\text{–}3}$ (5 bits)	CB22: $\mathbf{r}^{(2)}{}_{2,4\text{–}7}$ (5 bits)

Table 2. Structure of S-MSVQ in AMR-WB in the 6.60 kbps coding mode.

Structure of S-MSVQ			
Stage 1	CB1: $\mathbf{r}_1$ (1–9 order of $\mathbf{r}$) (8 bits)		CB2: $\mathbf{r}_2$ (10–16 order of $\mathbf{r}$) (8 bits)
Stage 2	CB11: $\mathbf{r}^{(2)}{}_{1,1\text{–}5}$ (7 bits)	CB12: $\mathbf{r}^{(2)}{}_{1,6\text{–}9}$ (7 bits)	CB21: $\mathbf{r}^{(2)}{}_{2,1\text{–}7}$ (6 bits)

3. BSS-VQ Algorithm for ISF Quantization

The basis of the BSS-VQ algorithm is that an input vector is efficiently assigned to a subspace where a small number of codeword searches is carried out using a combination of a fast locating technique and lookup tables, as a prerequisite of a VQ codebook search. In this manner, a significant computational load reduction can be achieved.

At the start of this algorithm, each dimension is dichotomized into two subspaces, and an input vector is then assigned to a corresponding subspace according to the entries of the input vector. This idea is illustrated in the following example. There are 2^9 = 512 subspaces for a 9-dimensional subvector $\mathbf{r}_1(n)$ associated with codebook CB1, and an input vector can then be assigned to one of the 512 subspaces by means of a dichotomy according to each entry of the input vector. Finally, VQ encoding is performed using a prebuilt lookup table containing the statistical information on sought codewords.

In this proposal, the lookup table in each subspace is pre-built in a way that requires lots of data for training purposes. A training as well as an encoding procedure in BSS-VQ is illustrated with the example of a 9-dimensional codebook CB1 with 256 entries in AMR-WB.

3.1. BSS Generation with Dichotomy Splitting

As a preliminary step of a training procedure, each dimension is dichotomized into two subspaces, and a dichotomy position is defined as the mean of all the codewords contained in a codebook, formulated as:

$$dp(j) = \frac{1}{CSize} \sum_{i=1}^{CSize} \mathbf{c}_i(j),\ 0 \le j < Dim \tag{8}$$

where $\mathbf{c}_i(j)$ represents the *j*th component of the *i*th codeword $\mathbf{c}_i$, *dp*(*j*) the mean value of all the *j*th components. Taking the codebook CB1 as an instance, *CSize* = 256, *Dim* = 9. As listed in Table 3, all the *dp*(*j*) values are saved and then presented in a tabular form.

Table 3. Dichotomy position for each dimension in the codebook CB1.

*j*th-Order	Mean
0	15.3816
1	19.0062
2	15.4689
3	21.3921
4	26.8766
5	28.1561
6	28.0969
7	21.6403
8	16.3302

Subsequently, for vector quantization on the *n*th input vector $\mathbf{x}_n$, a quantity $v_n(j)$ is defined as:

$$v_n(j) = \begin{cases} 2^j,\ \mathbf{x}_n(j) \ge dp(j) \\ 0,\ \mathbf{x}_n(j) < dp(j) \end{cases},\ 0 \le j < Dim \tag{9}$$

where $\mathbf{x}_n(j)$ denotes the *j*th component of $\mathbf{x}_n$. Then $\mathbf{x}_n$ is assigned to subspace *k* (bss_k), with *k* given as the sum of $v_n(j)$ over the entire dimensions, formulated as:

$$\text{Assigning} : \mathbf{x}_n \in bss_k | k = \sum_{j=0}^{Dim-1} v_n(j). \tag{10}$$

In this study, $0 \leq k < BSize$ and $BSize = 2^9 = 512$ represents the total number of subspaces. Taking an input vector $\mathbf{x}_n = \{20.0, 20.1, 20.2, 20.3, 20.4, 20.5, 20.6, 20.7, 20.8\}$ as an instance, $\nu_n(j) = \{2^0, 2^1, 2^2, 0, 0, 0, 0, 0, 2^8\}$ for each j, $0 \leq j \leq 8$, and $k = 263$ can be obtained by Equations (9) and (10) respectively. Thus, the input vector $\mathbf{x}_n$ is assigned to the subspace bss_k with $k = 263$.

By means of Equations (9) and (10), it is noted that this algorithm requires a small number of basic operations, i.e., comparison, shift and addition, such that an input vector is assigned to a subspace in a highly efficient manner.

3.2. Training Procedure of BSS-VQ

Following the determination of the dichotomy position for each dimension, a training procedure is performed to build a lookup table in each subspace. The lookup tables give the probability that each codeword serves as the best-matched codeword in each subspace, referred to as the hit probability of a codeword in a subspace for short.

A training procedure is stated below as Algorithm 1. With more than 1.56 GB of memory, a duration longer than 876 min and a total of 2,630,045 speech frames, a large speech database, covering a diversity of contents and multiple speakers, is employed as the training data in a training procedure.

Algorithm 1: Training procedure of BSS-VQ

Step 1. Initial setting: assign each codeword to all the subspaces, and then set the probability that the codeword $\mathbf{c}_i$ corresponds to the best-matched codeword in bss_k $P_{hit}(\mathbf{c}_i|bss_k) = 0$, $1 \leq i \leq CSize$, $0 \leq k < BSize$.

Step 2. Referencing Table 3 and through Equations (9) and (10), an input vector can be efficiently assigned to bss_k.

Step 3. A full search is conducted on all the codewords according to the Euclidean distance, given as:

$$dist(\mathbf{x}_n, \mathbf{c}_i) = \sum_{j=0}^{Dim-1} (\mathbf{x}_n(j) - \mathbf{c}_i(j))^2 \tag{11}$$

and an optimal codeword $\mathbf{c}_{opt}$ satisfies:

$$\mathbf{c}_{opt} = \underset{\mathbf{c}_i}{\operatorname{argmin}}\{dist(\mathbf{x}_n, \mathbf{c}_i)\},\ 1 \leq i \leq CSize. \tag{12}$$

Step 4. Update the statistics on the optimal codeword, that is, $P_{hit}(\mathbf{c}_{opt}|bss_k)$.

Step 5. Repeat Steps 2–4, until the training is performed on all the input vectors.

A lookup table is built for each subspace, following the completion of a training procedure. The lookup table gives the hit probability of each codeword in a subspace. For sorting purposes, a quantity $P_{hit}(m|bss_k)$, $1 \leq m \leq CSize$, is defined as the m ranked probability that a codeword hits the best-matched codeword in subspace bss_k. Taking $m = 1$ as an instance, $P_{hit}(m|bss_k)|_{m=1} = \max_{\mathbf{c}_i}\{P_{hit}(\mathbf{c}_i|bss_k)\}$, namely, the highest hit probability in bss_k. As it turns out, the lookup table in each subspace gives the ranked hit probability in descending order and the corresponding codeword.

3.3. Encoding Procedure of BSS-VQ

In the encoding procedure of BSS-VQ, the cumulative probability $P_{cum}(M|bss_k)$ is firstly defined as the sum of the top M $P_{hit}(m|bss_k)$ in bss_k, that is:

$$P_{cum}(M|bss_k) = \sum_{m=1}^{M} P_{hit}(m|bss_k),\ 1 \leq M \leq CSize. \tag{13}$$

Subsequently, given a threshold of quantization accuracy (TQA), a quantity $M_k(TQA)$ represents the minimum value of M that satisfies the condition $P_{cum}(M|bss_k) \geq TQA$ in bss_k, that is:

$$M_k(TQA) = \underset{M}{\operatorname{argmin}}\{M : P_{cum}(M|bss_k) \geq TQA\},\ 1 \leq M \leq CSize,\ 0 \leq k < BSize. \tag{14}$$

For a given TQA, a total of 512 $M_k(TQA)$s are evaluated by Equation (14) for all the subspaces, and the mean value is then given as:

$$\overline{M}(TQA) = \frac{1}{BSize}\sum_{k=0}^{BSize-1} M_k(TQA). \tag{15}$$

Illustrated in Figure 1 is a plot of the average number of searches $\overline{M}(TQA)$ corresponding to the values of TQA ranging between 0.90 and 0.99. Given a TQA = 0.95 as an instance, a mere average of 14.58 codeword searches is required to reach a search accuracy as high as 95%. In simple terms, the search performance can be significantly improved at the cost of a small drop in search accuracy. Furthermore, a trade-off can be made instantly between the quantization accuracy and the search load according to Figure 1. Hence, a BSS-VQ encoding procedure is described below as Algorithm 2.

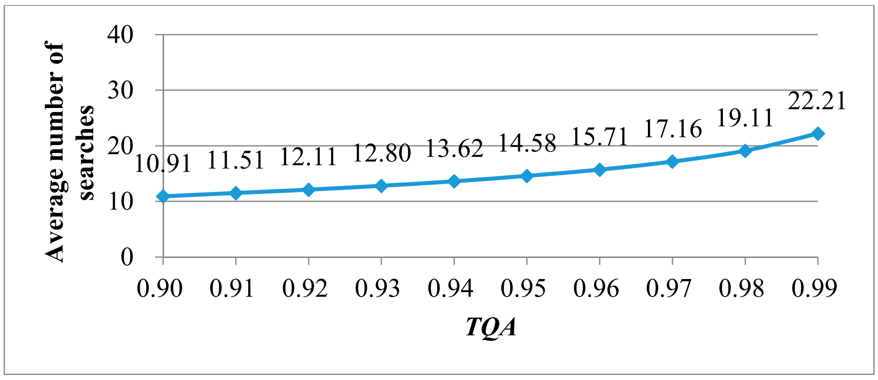

Figure 1. A plot of the average number of searches versus TQA.

Algorithm 2: Encoding procedure of BSS-VQ

Step 1. Given a TQA, $M_k(TQA)$ satisfying Equation (14) is found directly in the lookup table in bss_k.
Step 2. Referencing Table 3 and by means of Equations (9) and (10), an input vector is assigned to a subspace bss_k in an efficient manner.
Step 3. A full search for the best-matched codeword is performed on the top $M_k(TQA)$ sorted codewords in bss_k, and then the output is the index of the found codeword.
Step 4. Repeat Steps 2 and 3 until all the input vectors are encoded.

The BSS-VQ algorithm is briefly summarized as follows. Table 3 is the outcome by performing Equation (8) and is saved as the first lookup table. Subsequently, the second lookup table concerning $P_{hit}(m|bss_k)$ and the corresponding codeword is built for each subspace according to the training procedure. Accordingly, the VQ encoding can be performed using Algorithm 2.

4. Experimental Results

There are three experiments conducted in this work. The first is a search load comparison among various search approaches. The second is a quantization accuracy (QA) comparison among a full search and other search approaches. The third is a performance comparison among various approaches in terms of ITU-T P.862 perceptual evaluation of speech quality (PESQ) [21] as an objective measure of speech quality. A speech database, completely different from all the training data, is employed for outside testing purposes. With one male and one female speaker, the speech database in total takes up more than 221 MB of memory, occupies more than 120 min, and covers 363,281 speech frames.

Firstly, Table 4 lists a comparison on the average number of searches among full search, multiple TIE (MTIE) [13], DI-TIE, and EEENNS, while Table 5 gives the search load corresponding to *TQA* values of the BSS-VQ algorithm. Moreover, with the search load required in the full search algorithm as a benchmark, Tables 6 and 7 present comparisons on the load reduction (LR) with respect to Tables 4 and 5. A high value of LR reflects a high search load reduction. Table 6 indicates that DI-TIE provides a higher value of LR than MTIE and EEENNS search approaches among all the codebooks. It is also found that most LR values of BSS-VQ are higher than the DI-TIE approach by an observation in Tables 6 and 7. For example, the LR values of BSS-VQ are indeed higher than DI-TIE in case the *TQA* is equal to or smaller than 0.99, 0.98, 0.96, and 0.99 in codebooks CB1, CB2, CB21, and CB22, respectively. Accordingly, a remarkable search load reduction is reached by the BSS-VQ search algorithm.

Table 4. Average number of searches among various algorithms in the 8.85–23.85 kbps modes.

Codebooks		**Full Search**	**MTIE**	**DI-TIE**	**EEENNS**
Stage 1	CB1	256	91.57	42.46	58.82
	CB2	256	112.93	42.79	63.87
Stage 2	CB11	64	38.94	12.31	14.17
	CB12	128	82.59	14.40	22.91
	CB13	128	79.33	13.50	21.01
	CB21	32	23.55	8.95	11.08
	CB22	32	27.55	12.42	17.44

Table 5. Search load of the BSS-VQ algorithm versus *TQA* values in the 8.85–23.85 kbps modes.

TQA	**Average Number of Searches in Various Codebooks**						
	CB1	**CB2**	**CB11**	**CB12**	**CB13**	**CB21**	**CB22**
0.90	15.40	26.45	12.47	19.96	19.93	7.11	6.66
0.91	16.10	27.52	12.86	20.84	20.62	7.11	6.72
0.92	16.80	28.85	12.99	21.50	21.19	7.64	6.91
0.93	17.79	30.23	13.52	21.84	22.02	7.64	7.15
0.94	18.87	31.71	14.04	22.85	22.72	8.00	7.57
0.95	20.03	33.58	14.61	23.85	23.72	8.26	7.84
0.96	21.36	35.81	15.04	24.76	24.95	8.87	8.21
0.97	23.18	38.37	15.86	25.93	26.24	9.27	8.51
0.98	25.71	41.82	16.73	27.60	27.86	10.00	9.33
0.99	29.71	47.12	18.21	29.61	29.99	10.49	10.15

Table 6. LR percentage representation of Table 4.

Codebooks		**Full Search (Benchmark)**	**MTIE LR (%)**	**DI-TIE LR (%)**	**EEENNS LR (%)**
Stage 1	CB1	0	64.23	83.42	77.03
	CB2	0	55.89	83.29	75.05
Stage 2	CB11	0	39.16	80.77	77.86
	CB12	0	35.47	88.75	82.10
	CB13	0	38.02	89.46	83.58
	CB21	0	26.40	72.04	65.37
	CB22	0	13.89	61.18	45.50

Table 7. LR percentage representation of Table 5.

TQA	LR (%) in Various Codebooks						
	CB1	**CB2**	**CB11**	**CB12**	**CB13**	**CB21**	**CB22**
0.90	93.98	89.67	80.51	84.41	84.43	77.78	79.20
0.91	93.71	89.25	79.90	83.72	83.89	77.78	79.00
0.92	93.44	88.73	79.71	83.20	83.45	76.11	78.41
0.93	93.05	88.19	78.88	82.94	82.80	76.11	77.67
0.94	92.63	87.61	78.07	82.15	82.25	75.01	76.34
0.95	92.18	86.88	77.18	81.37	81.47	74.19	75.50
0.96	91.66	86.01	76.50	80.65	80.51	72.27	74.34
0.97	90.95	85.01	75.22	79.74	79.50	71.02	73.40
0.98	89.96	83.66	73.86	78.44	78.23	68.76	70.85
0.99	88.39	81.60	71.55	76.87	76.57	67.23	68.29

In the QA aspect, a 100% QA is obtained by the MTIE, DI-TIE, and EEENNS algorithms as compared with a full search approach. Thus, only the QA experiment of BSS-VQ is conducted. The QA corresponding to *TQA* values of the BSS-VQ algorithm is given in Table 8. It reveals that QA is an approximation of *TQA* in either inside or outside testing cases. Moreover, this algorithm provides an LR between 77.78% and 93.98% at *TQA* = 0.90 as well as an LR between 67.23% and 88.39% at *TQA* = 0.99, depending on the codebooks. In other words, a trade-off can be made between the quantization accuracy and the search load.

Table 8. Comparison of QA percentage of the BSS-VQ algorithm versus *TQA* values in the 8.85–23.85 kbps modes among codebooks.

TQA	QA (%) in Various Codebooks						
	CB1	**CB2**	**CB11**	**CB12**	**CB13**	**CB21**	**CB22**
0.90	90.42	90.63	90.84	90.89	91.06	93.16	93.10
0.91	91.52	91.46	91.96	92.18	92.12	93.16	93.27
0.92	92.32	92.58	92.44	93.18	93.05	94.86	93.92
0.93	93.25	93.60	93.54	93.67	94.15	94.86	94.59
0.94	94.20	94.39	94.68	94.65	95.00	95.66	95.93
0.95	95.24	95.35	95.88	95.71	95.82	96.40	96.46
0.96	96.03	96.30	96.35	96.43	96.85	97.33	97.11
0.97	97.01	97.16	97.34	97.36	97.47	97.90	97.57
0.98	97.91	98.08	98.17	98.34	98.32	99.00	98.75
0.99	98.82	99.01	99.23	99.17	99.09	99.39	99.33

Furthermore, an overall LR is evaluated to observe the total search load of an entire VQ encoding procedure of an input vector. The overall LR refers to the total search load, defined as the sum of the average number of searches multiplied by the vector dimension in each codebook. Thus, an overall LR comparison with the full search as a benchmark is presented as a bar graph in Figure 2. As clearly indicated in Figure 2, the overall LR of BSS-VQ is higher than MTIE, DI-TIE, and EEENNS approaches, but at the same time the QA is as high as 0.98. Moreover, Table 9 gives a PESQ comparison, including the mean and the STD, among various approaches. Since MTIE, DI-TIE, and EEENNS provide a 100% QA, they both share the same PESQ with a full search, meaning that there is no deterioration in the speech quality. A close observation reveals little difference between PESQs obtained in a full search and in this search algorithm, that is, the speech quality is well maintained in BSS-VQ at *TQA* not less than 0.90. This BSS-VQ search algorithm is experimentally validated as a superior candidate relative to its counterparts.

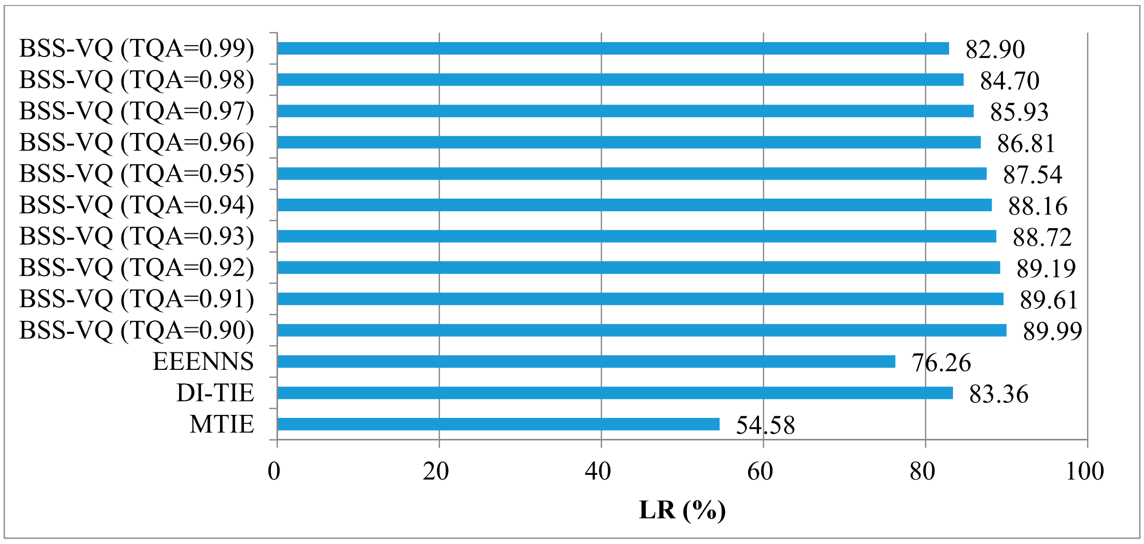

Figure 2. Comparison of overall search load reduction among various approaches.

Table 9. Comparison on mean opinion score (MOS) values using the PESQ algorithm among various methods.

Methods		**MOS**	
		Mean	**STD**
Full search		3.931	0.329
BSS-VQ (*TQA*)	0.90	3.930	0.328
	0.91	3.931	0.328
	0.92	3.931	0.330
	0.93	3.931	0.328
	0.94	3.930	0.329
	0.95	3.933	0.329
	0.96	3.931	0.330
	0.97	3.930	0.330
	0.98	3.932	0.328
	0.99	3.933	0.328

5. Conclusions

This paper presents a BSS-VQ codebook search algorithm for ISF vector quantization in the AMR-WB speech codec. Using a combination of a fast locating technique and lookup tables, an input vector is efficiently assigned to a search subspace where a small number of codeword searches is carried out and the aim of remarkable search load reduction is reached consequently. Particularly, a trade-off can be made between the quantization accuracy and the search load to meet a user's need when performing a VQ encoding. This BSS-VQ search algorithm, providing a considerable search load reduction as well as nearly lossless speech quality, is experimentally validated as superior to MTIE, DI-TIE, and EEENNS approaches. Furthermore, this improved AMR-WB speech codec can be adopted to upgrade the VoIP performance on a smartphone. As a consequence, the energy efficiency requirement is achieved for an extended operation time period due to computational load reduction.

Acknowledgments: This research was financially supported by the Ministry of Science and Technology under grant number MOST 105-2221-E-167-031, Taiwan. The author felt deeply indebted as well to the co-authors of two cited papers, including Tzu-Hung Lin, Shaw-Hwa Hwang, Shun-Chieh Chang and Bing-Jhih Yao, for their contribution as the basis of this work.

Conflicts of Interest: The author declares no conflict of interest.

References

1. 3rd Generation Partnership Project (3GPP). *Adaptive Multi-Rate—Wideband (AMR-WB) Speech Codec; Transcoding Functions*; TS 26.190; 3GPP: Valbonne, France, 2012.
2. Ojala, P.; Lakaniemi, A.; Lepanaho, H.; Jokimies, M. The adaptive multirate wideband speech codec: System characteristics, quality advances, and deployment strategies. *IEEE Commun. Mag.* **2006**, *44*, 59–65. [CrossRef]
3. Varga, I.; De Lacovo, R.D.; Usai, P. Standardization of the AMR wideband speech codec in 3GPP and ITU-T. *IEEE Commun. Mag.* **2006**, *44*, 66–73. [CrossRef]
4. Bessette, B.; Salami, R.; Lefebvre, R.; Jelínek, M.; Rotola-Pukkila, J.; Vainio, J.; Mikkola, H.; Järvinen, K. The adaptive multirate wideband speech codec (AMR-WB). *IEEE Trans. Speech Audio Process.* **2002**, *10*, 620–636. [CrossRef]
5. Salami, R.; Laflamme, C.; Adoul, J.P.; Kataoka, A.; Hayashi, S.; Moriya, T.; Lamblin, C.; Massaloux, D.; Proust, S.; Kroon, P.; et al. Design and description of CS-ACELP: A toll quality 8 kb/s speech coder. *IEEE Trans. Speech Audio Process.* **1998**, *6*, 116–130. [CrossRef]
6. Linde, Y.; Buzo, A.; Gray, R.M. An algorithm for vector quantizer design. *IEEE Trans. Commun.* **1980**, *28*, 84–95. [CrossRef]
7. Wang, L.; Chen, Z.; Yin, F. A novel hierarchical decomposition vector quantization method for high-order LPC parameters. *IEEE Trans. Audio Speech Lang. Process.* **2015**, *23*, 212–221. [CrossRef]
8. Ramirez, M.A. Intra-predictive switched split vector quantization of speech spectra. *IEEE Signal Process. Lett.* **2013**, *20*, 791–794. [CrossRef]
9. Han, W.J.; Kim, E.K.; Oh, Y.H. Multicodebook split vector quantization of LSF parameters. *IEEE Signal Process. Lett.* **2002**, *9*, 418–421.
10. Chatterjee, S.; Sreenivas, T.V. Optimum switched split vector quantization of LSF parameters. *Signal Process.* **2008**, *88*, 1528–1538. [CrossRef]
11. Hwang, S.H.; Chen, S.H. Fast encoding algorithm for VQ-based image coding. *Electron. Lett.* **1990**, *26*, 1618–1619.
12. Choi, S.Y.; Chae, S.I. Incremental-search fast vector quantiser using triangular inequalities for multiple anchors. *Electron. Lett.* **1998**, *34*, 1192–1193. [CrossRef]
13. Hsieh, C.H.; Liu, Y.J. Fast search algorithms for vector quantization of images using multiple triangle inequalities and wavelet transform. *IEEE Trans. Image Process.* **2000**, *9*, 321–328. [CrossRef] [PubMed]
14. Yao, B.J.; Yeh, C.Y.; Hwang, S.H. A search complexity improvement of vector quantization to immittance spectral frequency coefficients in AMR-WB speech codec. *Symmetry* **2016**, *8*, 1–8. [CrossRef]
15. Lu, Z.M.; Sun, S.H. Equal-average equal-variance equal-norm nearest neighbor search algorithm for vector quantization. *IEICE Trans. Inf. Syst.* **2003**, *86*, 660–663.
16. Chu, S.C.; Lu, Z.M.; Pan, J.S. Hadamard transform based fast codeword search algorithm for high-dimensional VQ encoding. *Inf. Sci.* **2007**, *177*, 734–746. [CrossRef]
17. Chen, S.X.; Li, F.W.; Zhu, W.L. Fast searching algorithm for vector quantisation based on features of vector and subvector. *IET Image Process.* **2008**, *2*, 275–285. [CrossRef]
18. Chen, S.X.; Li, F.W. Fast encoding method for vector quantisation of images using subvector characteristics and Hadamard transform. *IET Image Process.* **2011**, *5*, 18–24. [CrossRef]
19. Xia, S.; Xiong, Z.; Luo, Y.; Dong, L.; Zhang, G. Location difference of multiple distances based k-nearest neighbors algorithm. *Knowl.-Based Syst.* **2015**, *90*, 99–110. [CrossRef]
20. Lin, T.H.; Yeh, C.Y.; Hwang, S.H.; Chang, S.C. Efficient binary search space-structured VQ encoder applied to a line spectral frequency quantisation in G.729 standard. *IET Commun.* **2016**, *10*, 1183–1188. [CrossRef]
21. ITU-T Recommendation P.862. *Perceptual Evaluation of Speech Quality (PESQ): An Objective Method for End-to-End Speech Quality Assessment of Narrow-Band Telephone Networks and Speech Codecs*; ITU (International Telecommunication Union): Geneva, Switzerland, 2001.