Journal of *Marine Science and Engineering*

MDPI

Article

An Attention-Averaging-Based Compression Algorithm for Real-Time Transmission of Ship Data via Beidou Navigation System

Chunchang Zhang [1] and Ji Zeng [2,*]

[1] Merchant Marine College, Shanghai Maritime University, Shanghai 201306, China; cczhang@shmtu.edu.cn
[2] Zhiyuan College, Shanghai Jiao Tong University, Shanghai 200240, China
* Correspondence: zeng_ji@sjtu.edu.cn

Abstract: The real-time transmission of ship status data from vessels to shore is crucial for live status monitoring and guidance. Traditional reliance on expensive maritime satellite systems for this purpose is being reconsidered with the emergence of the global short message communication service offered by the BeiDou-3 navigation satellite system. While this system presents a more cost-effective solution, its bandwidth is notably insufficient for handling real-time ship status data. This inadequacy necessitates the compression of such data. Therefore, this paper introduces an algorithm tailored for real-time compression of sequential ship status data. The algorithm is engineered to ensure both accuracy and the preservation of valid data range integrity. Our methodology integrates quantization, predictive coding employing an attention-averaging-based predictor, and arithmetic coding. This combined approach facilitates the transmission of succinct messages through the BeiDou Navigation System, enabling the live monitoring of ocean-going vessels. Experimental trials conducted with authentic data obtained from ship monitoring systems validate the efficiency of our approach. The achieved compression rates closely approximate theoretical minimum values. Consequently, this method exhibits substantial promise for the real-time transmission of parameters across various systems.

Keywords: ship data compression; predictive coding; attention averaging; Convolutional Neural Networks (CNNs)

Citation: Zhang, C.; Zeng, J. An Attention-Averaging-Based Compression Algorithm for Real-Time Transmission of Ship Data via Beidou Navigation System. *J. Mar. Sci. Eng.* **2024**, *12*, 300. https://doi.org/10.3390/jmse12020300

Academic Editor: Diego Villa

Received: 3 January 2024
Revised: 1 February 2024
Accepted: 6 February 2024
Published: 8 February 2024

1. Introduction

The advent of concepts such as smart ships and unmanned vessels has highlighted the growing need for remote ship status monitoring. Ships generate a multitude of operational parameters, such as speed, engine torque, and fuel consumption. By analyzing these data, insights into the ship's operational status and its energy-saving impact can be derived. The real-time transmission of these data allows onshore management personnel to analyze the system's status based on these parameters and issue real-time instructions to enhance energy efficiency and safety.

With the help of real-time data, common risk factors, such as equipment failure, human operational errors, and navigation accidents, can be promptly identified and eliminated by the shore-based data analysis system [1,2]. The shore-based system can issue more efficient dispatch instructions, such as Virtual Arrival, to save carbon emissions by utilizing the ship's position and speed data [3].

Traditional monitoring systems typically rely on costly maritime satellite systems to transmit data. To address the cost issue, the global short message communication service offered by the BeiDou-3 navigation satellite system can be leveraged for the nearly cost-free transmission of ship monitoring data [4,5]. However, its bandwidth falls significantly short of the requirements for this scenario. Specifically, within the global short message communication service of the BeiDou-3 navigation satellite system, each message cannot

exceed 560 bits in length, and a single civil communication card can only send one message per minute [4]. The success rate decreases as the message length increases, sometimes dropping as low as 50% [6–8].

In our dataset, for instance, there are 41 fields to transmit, which would require 1312 bits when encoded in float32. The monitoring system records a data point every 10 s, with no tolerance for data loss. Although existing methods, such as utilizing multiple communication cards to increase bandwidth [9] and employing channel coding to eliminate data loss [8,9], can be adopted, the capacity of Beidou falls significantly short of the number of data to be transmitted. Therefore, data compression is necessary to minimize its length. Furthermore, both the compression algorithm's allowable calculation time and the computing power of both the sender on the ship and the receiver in the database are relatively abundant. Hence, optimizing the compression rate becomes the primary objective in the compression problem for the real-time transmission of ship data via Beidou Navigation System.

In this paper, we introduce our QAAPA method, a novel method combining **Q**uantization, **A**ttention-**A**veraging-based **P**redictive coding, and **A**rithmetic coding and facilitating the real-time transmission of ship data via the Beidou Navigation System. Our contributions can be summarized as follows:

- To address the challenge of economically transmitting real-time ship status data via the Beidou system, we have developed the QAAPA Method. This method comprises a series of tailored steps designed to compress data in real-time while maintaining fixed precision.
- Unlike traditional compression methods that transmit entire datasets, our approach transmits only the residuals, resulting in a significant reduction in data size. This transmission of residuals is made possible through a critical step in our method: attention-averaging-based predictive coding. Here, we compute the difference between actual and predicted data, reconstructing it at the receiver's end. Our predictor is built on a foundation of linear regression models and Convolutional Neural Networks (CNNs), incorporating appropriate weights. This framework ensures accurate real-time predictions by leveraging historical data, enabling precise reconstruction of transmitted residuals.
- Experimental Validation: We conducted experiments using actual ship monitoring system data. Results showcase that QAAPA achieves compression rates approaching theoretical minimums estimated by commercial algorithms, surpassing traditional approaches for similar problems.

The remainder of this paper is structured as follows: Section 2 discusses existing compression solutions, highlighting the absence of suitable methods for real-time ship status data transmission via Beidou. Section 3 details the QAAPA method tailored for Beidou-based ship data transmission. Section 4 presents experiments validating QAAPA's performance. Finally, Section 5 concludes the paper and outlines future work.

2. Related Work

The BeiDou-3 navigation satellite system's global short message communication service is integral to numerous monitoring systems operating where ground-based communication systems are unavailable [10–13]. However, prevalent encoding formats within these systems often lack significant compression effects or prove unsuitable for our specific challenge. Traditional methods usually use related techniques of signal processing [11,13]. In recent years, many studies have used machine learning for data compression [14]. Compression algorithms combined with machine learning are mainly used for images, sound and video, and text [15–18], and machine learning is rarely used in the compression of instrument status data. Common compression algorithms applied to numerical data, such as vector quantization [19] and transformation coding [20], fail to ensure our fixed precision goals while maintaining high compression rates.

Predictive coding presents the potential to achieve both objectives. Crucial to ensuring fixed precision is the encoding of residuals in predictive coding. The approach proposed by Zhidan Yan et al. [21] involves reducing the value range of residuals, while another approach proposed by Fout Nathaniel and Ma Kwan-Liu [22] adopts a variable precision floating-point format for encoding residuals, reducing precision for larger absolute values. However, both of them neglected meeting fixed precision goals, while neither approach guarantees fixed precision, especially considering sharp changes in ship states during engine start/stop cycles, rendering these methods unsuitable.

The utilization of machine learning-based prediction in predictive coding holds promise for enhancing the precision of predicted values. Past research in the compression of text or genome data has demonstrated successful applications of predictive coding. For example, Goyal Mohit et al. [23] and Cox David [18] approached data as symbol sequences, employing prediction methodologies across the spectrum of potential data values. However, our scenario involving ship status data predominantly comprises floating-point data spanning an extensive range of potential values. This broad spectrum renders predicting every possible value's probability nearly unattainable, a challenge not present in the more confined scope of letters or genomes. Consequently, the application of their compression methodologies to our scenario is unfeasible. Chen Yong [13] proposed a data compression method specifically tailored for enabling communication through Beidou's short message system. This method combines the Spinning Door Transformation algorithm for initial lossy compression and the Pre-special byte method for subsequent lossless compression. Nevertheless, this compression methodology is optimized for scalar sequences and cannot be readily adapted and optimized for our specific ship-status data structure, which consists of vector sequential data.

The state-of-the-art data compression techniques available are primarily designed for general or specific scenarios that diverge significantly from the unique nature of our situation regarding real-time transmission of ship status data, particularly vector sequential data. Consequently, none of the existing methodologies can be directly applied to our context. This predicament underscores the necessity of devising a bespoke data compression methodology tailored explicitly to our specific scenario. This necessity has served as the driving force behind our research endeavors, which will be comprehensively expounded upon in the subsequent sections.

3. Methodology

3.1. Preliminaries

3.1.1. Formal Definition

The compression problem for the real-time transmission of ship data via the BeiDou Navigation System can be described as follows:

Periodically, the sender generates a set of ship status data, denoted as $\mathbf{x}_t \in \mathbb{R}^n$, representing variables such as the ship's speed, engine torque, and fuel consumption, at the current time t. Both the sender and receiver maintain a record of all historical data sequences, $\mathbf{x}_{<t}$, which have been previously transmitted. These historical sequences are integral to the compression algorithm.

The sender employs an encoding function, denoted as E, to encode $\mathbf{x}_t$ into a binary string $c = E(\mathbf{x}_t|\mathbf{x}_{<t})$, which is then transmitted via the BeiDou short message communication system. Upon receiving this message c, the receiver performs decoding, resulting in an approximation of the original message, $\mathbf{x}'_t = D(c|\mathbf{x}_{<t})$.

Additionally, predetermined upper and lower limits exist, denoted as $\mathbf{L}_u, \mathbf{L}_d \in \mathbb{R}^n$, as well as transmission decimal places, denoted as $\mathbf{a} \in \mathbb{Z}^n$. These limits and decimal places are determined by the monitoring system's administrator according to their specific requirements. Our compression algorithm must encompass all potential values of $\mathbf{x}_t$, ensuring that the decoded approximation, $\mathbf{x}'_t$, meets the required level of accuracy. Moreover, the algorithm must strive to minimize the average coding length of the message c.

Formally, the compression algorithm must guarantee that:

$$\forall \mathbf{x}_t \in \mathbb{R}^n, \mathbf{x}_{t,i} \in [\mathbf{L}_{d,i}, \mathbf{L}_{u,i}], c = E(\mathbf{x}_t|\mathbf{x}_{<t}), \mathbf{x}'_t = D(c|\mathbf{x}_{<t}), |\mathbf{x}'_{t,i} - \mathbf{x}_{t,i}| \leq 10^{-\mathbf{a}_i}, \quad 0 \leq i < n \tag{1}$$

and minimize the average coding length of the message c. When applying this problem in practical scenarios, we make the following key assumptions:

- The interpretation of each dimension within the data vector, denoted as $\mathbf{x}$, remains constant, and each dimension represents a parameter associated with the ship's internal state.
- Both the sender and receiver utilize computers, as opposed to circuit boards, to process the data. They possess sufficient computing power and time (several seconds) to execute intricate compression algorithms.
- With the help of appropriate channel coding [8,9], the communication process is reliable.

3.1.2. Quantization

Quantization is a fundamental process involving the approximation of continuous signal values or a vast array of discrete values into a finite and often reduced set of discrete values [24]. It is used to ensure the fixed accuracy goal required in the problem while eliminating the unnecessary precision of the data.

3.1.3. Predictive Coding

Predictive coding, utilized in data compression, minimizes the data needed to represent information by transmitting or storing only the information that cannot be accurately predicted. Initially, it identifies repetition and similarity within the data, generates predicted values using a model, and compares them with actual values to derive residuals. These residuals, ideally smaller in size than the original data, facilitate more efficient transmission or storage [25]. The accuracy of the chosen predictive model significantly influences the effectiveness of predictive coding. It particularly excels in scenarios with high correlation or predictability between successive data points.

3.1.4. Arithmetic Coding

Arithmetic coding is a technique used in data compression to encode a sequence of symbols or characters with variable-length codes. Unlike more common methods like Huffman coding, which assigns fixed-length codes to individual symbols, arithmetic coding encodes entire sequences of symbols into a single code. Its typical process involves the following steps:

- Symbol Probability Modeling: The process starts with a probability model that estimates the probability of each symbol in the input sequence.
- Defining the Range: Represent the entire range of possible values with a fractional range between 0 and 1. This range is divided proportionally to the probabilities of the symbols.
- Mapping Symbols to Subranges: Each symbol in the input sequence is then mapped to a subrange within the total range based on their probabilities.
- Iterative Encoding: As each symbol is encoded, the subrange corresponding to that symbol is selected and used to narrow down the overall range further.
- Outputting the Code: The final output is a binary fraction within the narrowed range, which represents the compressed form of the entire input sequence.

Arithmetic coding is advantageous in that it can achieve higher compression ratios compared to methods like Huffman coding because it utilizes the probabilities of entire sequences rather than individual symbols. However, it requires more computational resources and may be more complex to implement efficiently.

3.1.5. Linear Regression and Convolutional Neural Networks (CNNs)

In our ship status data compression process, linear regression and CNN models are used. Linear regression is a fundamental statistical method used to model the relationship

between two or more variables. Its simplicity and interpretability make it a widely applied technique in various fields, including economics, social sciences, health sciences, and machine learning.

At its core, linear regression aims to establish a linear relationship between a dependent variable (the outcome of interest) and one or more independent variables (predictors or explanatory variables). The model assumes that the relationship between these variables can be approximated by a straight line.

Mathematically, the linear regression model can be represented as:

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n \quad (2)$$

The predictive values $\hat{y}$ are calculated as the weighted sum of input features, with n representing the number of features, x_n representing the ith feature value, and θ_j representing the jth model parameter (particularly, θ_j is the intercept, indicating the value of y when all independent variables are zero) [26]. However, it should be noted that these linear regression models assume a linear relationship by default [27].

In recent years, it has become popular to use neural networks to simulate the learning process of the human brain to construct models and provide automated solutions within the field of Artificial Intelligence (AI). Neural networks consist of interconnected neurons with numerous parameters, and they exhibit layers connecting input and output [28]. Convolutional Neural Networks (CNNs) [29] represent a class of deep neural networks specifically designed to process visual data. They have revolutionized various fields, ranging from computer vision to natural language processing, due to their capability to automatically extract intricate features from input data. Originally inspired by the visual processing mechanism of the human brain, CNNs have demonstrated remarkable success in tasks such as image classification, object detection, and segmentation. The essence of CNNs lies in their architecture, which comprises convolutional layers, pooling layers, and fully connected layers. Convolutional layers apply learnable filters to input images, extracting features hierarchically by convolving across the image. Pooling layers subsequently downsample these features, reducing computational complexity while retaining essential information. Fully connected layers, situated at the end of the network, utilize these learned features for classification or regression tasks. If the two-dimensional convolution in its convolutional layer is changed to a one-dimensional convolution, it can also be used for sequence data. In our ship data compression scenario, we applied CNNs in the predictive coding step and achieved relatively higher prediction accuracy than other deep neural networks, such as RNNs (Recurrent Neural Networks).

3.2. Overview of Proposed Method

The real-time transmission of ship status data comprises three primary stages (see Figure 1). Initially, at the ship's end, the ship status data undergoes encoding to reduce the message length for transmission. This encoding process encompasses quantization, predictive coding, and arithmetic coding steps. Subsequently, the compressed message is transmitted via the Beidou system to the intended receiver. Finally, upon reception, the receiver decodes (recovers) the message, initiating a reverse arithmetic decoding step followed by another predictive coding step. It is noteworthy that all blocks, except for quantization, maintain a lossless nature.

For a formal definition of the input, steps, and output involved in the encoding and decoding processes, refer to Algorithms 1 and 2 for detailed descriptions. Subsequent sections will elaborate on each step sequentially, providing in-depth insights into the methodologies.

Algorithm 1 Encoding algorithm

Input:
$\mathbf{x}_t \in \mathbb{R}^n$: current data to be compressed
$\{\mathbf{y}_\tau\}_{\tau=t-m}^{t-1} \in \mathbb{Z}^{n \cdot m}$: synced recent historical data
$P : \mathbb{Z}^{n \cdot m} \to \mathbb{Z}^n$ predictor
$Q : \mathbb{R}^n \to \mathbb{Z}^n$ quantizer
$A_e : \mathbb{Z}^n \to$ binary string arithmetic encoder with pre-calculated frequency table
Output: c: encoded message
$\mathbf{y}_t \leftarrow Q(\mathbf{x}_t)$
$\mathbf{y}'_t \leftarrow P(\{\mathbf{y}_\tau\}_{\tau=t-m}^{t-1})$
$\mathbf{e}_{t,i} \leftarrow (\mathbf{y}_t - \mathbf{y}'_t)_i \bmod L_i$
$c \leftarrow A_e(\mathbf{e}_t)$

Algorithm 2 Decoding algorithm

Input:
c: encoded message
$\{\mathbf{y}_\tau\}_{\tau=t-m}^{t-1} \in \mathbb{Z}^{n \cdot m}$: synced recent historical data
$P : \mathbb{Z}^{n \cdot m} \to \mathbb{Z}^n$ pre-trained predictor
$A_d :$ binary string $\to \mathbb{Z}^n$ arithmetic decoder with pre-calculated frequency table
Output: $\mathbf{x}'_t \in \mathbb{R}^n$ approximation of the original data
$\mathbf{e}_t \leftarrow A_d(c)$
$\mathbf{y}'_t \leftarrow P(\{\mathbf{y}_\tau\}_{\tau=t-m}^{t-1})$
$\mathbf{y}_{t,i} \leftarrow (\mathbf{y}'_t + \mathbf{e}_t)_i \bmod L_i$
$\mathbf{x}'_t \leftarrow \mathbf{y}_t$

3.3. Quantization

Quantization, as introduced in Section 3.1.2, serves as a pivotal process to approximate continuous signal values or extensive discrete datasets into a finite and often condensed set of discrete values. In the domain of ship sensor data, this principle holds significant relevance for several reasons. Firstly, sensor measurement accuracy faces inherent limitations owing to environmental noise, technological constraints, or sensor precision, making the transmission of sensor readings with absolute accuracy unnecessary for the monitoring system. Instead, the system can maintain data precision to a specific decimal level. Moreover, ensuring sensor readings adhere to a predefined range is crucial, preventing the inclusion of outliers or erroneous values due to sensor anomalies. Effectively mapping floating-point sensor readings to a finite set of integers capitalizes on these properties. Quantization, as the initial step in the compression process, offers the system the capability to establish accurate thresholds consistently for all $\mathbf{x}_t \in STH$—a critical factor in aligning the algorithm with precise ship monitoring requirements. In essence, quantization strikes a balance between data precision and efficiency. It reduces data volume while maintaining a level of precision that precisely aligns with monitoring needs. Hence, we employed quantization to reduce ship sensor data volume while preserving the necessary precision. Fulfilling the demands of data precision and efficiency, the quantization step necessitates a configuration table. This table elucidates the practical significance of ship sensor data and defines the effective range required by the Beidou system. Practically, we prepared and maintained the configuration table in JSON format. Formally, the quantization step is described as follows:

$$Q(\mathbf{x})_i = \text{round}\left(\frac{\mathbf{x}_i - \mathbf{L}_{d,i}}{10^{\mathbf{a}_i}}\right) \tag{3}$$

where round stands for the round function, which finds the closest integer of any real number.

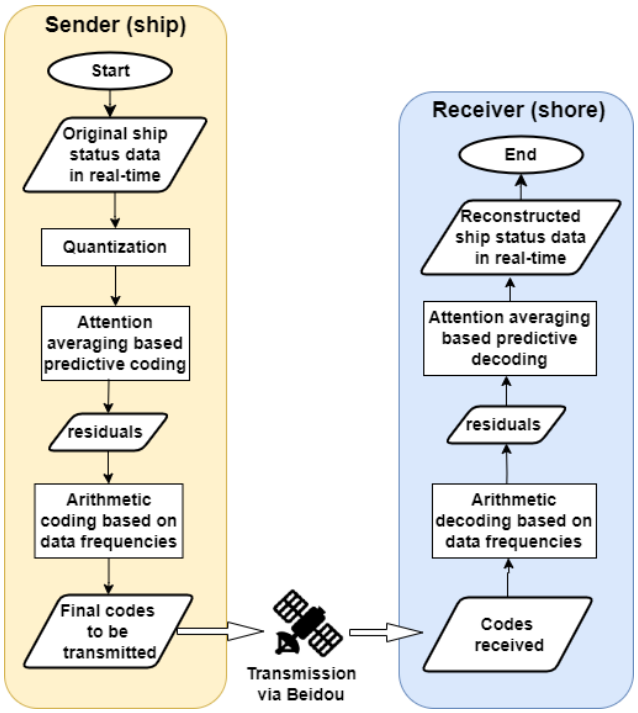


Figure 1. Flow chart of our proposed method. This figure is only used to show the global process of our data compression approach; therefore, it does not contain the technique details of each step. The corresponding specific structures or implementation details can be found in the subsequent sections.

The post-quantization data $\mathbf{y}_t \in \mathbb{Z}^n$ are defined as $\mathbf{y}_t = Q(\mathbf{x}_t)$. Each dimension of $\mathbf{y}_t$ is limited to L_i possible values, a constraint arising from finite accuracy goals and specific upper and lower limits. Notably, L_i exhibits considerable variability across our dataset, ranging from 100 to 100,000. This substantial variation poses a challenge during subsequent encoding procedures.

3.4. Attention-Averaging-Based Predictive Coding

Following the quantization step, a pivotal aspect of our proposed method to reduce data size involves the attention-averaging-machine-learning-based predictive coding step. In this process, a predictive model anticipates data values, comparing them with the actual values to derive errors or residuals. These residuals, representing the difference between predicted and actual data, are transmitted or stored in a reduced data size instead of the entire dataset.

The effectiveness of predictive coding hinges on the precision of the applied predictive model. Machine-learning-based prediction offers potential for generating more accurate predicted values, which are notably successful in compressing textual and genomic data [18,23]. However, text and genomic data's limited symbol variations enable the generation of probability values for each input symbol, facilitating their application in arithmetic encoding. Yet, directly applying existing predictive coding methods to ship status data presents a challenge. The extensive range and volume of ship status data values make generating probability values for each integer point impractical.

To address this, we designed a tailored predictive coding process for ship status data, enabling the transmission of residual errors between predicted and true values instead of transmitting the entire sensor data for arithmetic coding. This approach proves feasible for ship status data due to its periodic generation and transmission, maintaining continuous time correlations. Leveraging transmitted historical data at the receiver end, we can pre-train a predictor based on this historical ship sensor data. Calculating the residual error between predicted and true values of current real-time ship data enables us to transmit only these residuals to the next arithmetic coding step. These residuals can be

reconstructed at the receiver end using the same predictor, ensuring the retrieval of the complete required data.

Figure 2 illustrates the specific process of our custom-designed predictive coding method. This method's key sub-step involves pre-training a predictor based on historical ship sensor data, utilized to generate predicted values for both ship and shore-side data. This pre-trained predictor enables the transmission of significantly smaller residual errors rather than the entire dataset to the subsequent arithmetic coding step. Formally, the predictor P forecasts the quantized current data $\mathbf{y}_t$ using the most recent m quantized historical data $\{\mathbf{y}_\tau\}_{\tau=t-m}^{t-1}$[1].

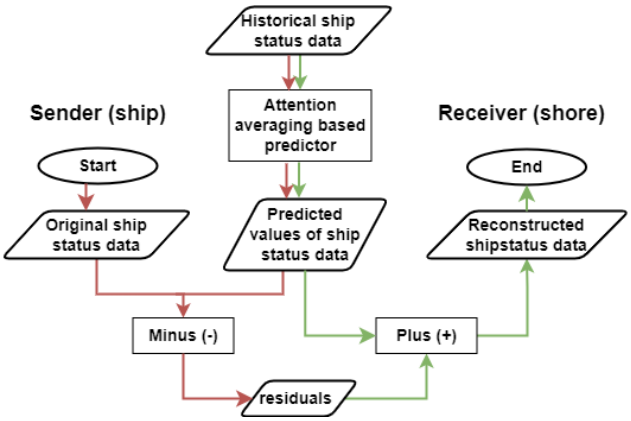


Figure 2. Flowchart of attention averaging based predictive coding.

Subsequently, only the predicted error $\mathbf{e}_{t,i} = (\mathbf{y}_t - \mathbf{y}'_t)_i \bmod L_i$ is transmitted, facilitating the reconstruction of $\mathbf{y}_t$ as $\mathbf{y}_{t,i} = (\mathbf{e}_t + \mathbf{y}'_t)_i \bmod L_i$. Both sender and receiver independently calculate $\mathbf{y}'_t$ using synchronized historical data $\{\mathbf{y}\tau\}_{\tau=t-m}^{t-1}$, ensuring identical results. Given that $\mathbf{e}_t$ tends to be close to zero most of the time, predictive coding substantially reduces message length, aided by arithmetic coding in Section 3.5.

3.4.1. Attention-Averaging-Based Predictor

In predictive coding, the accuracy of the predictor directly influences the length of transmitted information. Optimal predictor selection significantly impacts the efficacy of predictive coding. In our approach, we employed a predictor utilizing a blending concept akin to the attention-based averaging method proposed in [30]. This approach enables the blending of a linear model with a CNN, enhancing prediction accuracy. Our blending strategy stems from empirical observations of solely employing the linear or CNN models to predict ship status data. These observations revealed differing strengths of both models across different data fields. Consequently, to leverage their respective advantages, we designed an attention-averaging-based method, assigning different weights to the linear model and CNN based on distinct data properties.

Figure 3 illustrates the structure of our attention-averaging-based predictor. Initially, the predictor takes historical data as input for both the linear regression and CNN models. The linear regression model, unique in this paper, operates without normalization, differing from conventional linear prediction methods common in data compression. It processes 2-D historical data as input and produces a 1-D vector as a predicted value for all fields. The weights in the linear model are calculated using the SVD algorithm, optimizing their performance based on historical training data rather than being theoretically derived.

Concerning the CNN model, we adopted the WaveNet architecture [31], renowned for its success in audio processing. Within a WaveNet block, multi-layer convolutional kernels of size 2 and gradually increased diffusion rates are employed. The global CNN model structure includes normalization in the first layer, followed by a WaveNet block, a flattening layer, and two densely connected layers (as shown in Figure 4b). To optimize the model size, we applied valid padding.

Subsequently, the outputs from both the linear regression and CNN models are fed into our attention averaging network alongside the original historical data. This network, comprising three densely connected layers, determines the suitable weights for the linear regression and CNN models based on current data characteristics. Notably, the weights generated by the attention averaging network are vectors rather than scalars. This distinction allows field-specific variation in the weights of the linear model and CNN in the weighted average, enabling the model to adapt predictors to diverse field characteristics.

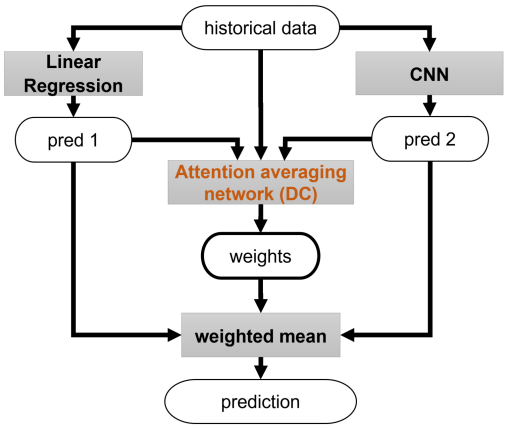
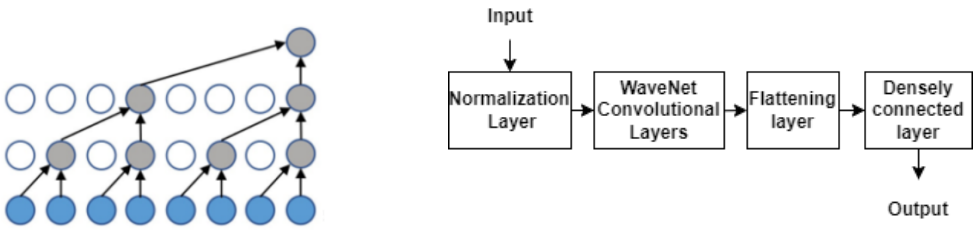


Figure 3. Structure of predictor.



(**a**) (**b**)

Figure 4. CNN model used in the attention averaging based predictor. (**a**) WaveNet Convolutional Layers. (**b**) The global CNN architecture.

Moreover, the attention averaging model, by considering both the original inputs and outputs of the prediction models, dynamically adjusts weights based on different input conditions. Consequently, this approach amalgamates the strengths of linear models and neural networks. Additionally, to ensure the sum of output weights in each dimension equals 1, the Softmax function serves as the output layer activation function.

Once the attention averaging-based predictor determines suitable weights for the two models, it computes the final predicted values using these generated weights. The aim is to optimize prediction accuracy specifically tailored to ship data.

3.4.2. Residual Generation

Following the prediction by our attention averaging-based predictor, the values are quantized into integers. This quantization enables the calculation of residuals—derived by subtracting the quantized predicted values from the quantized actual values of the current ship data—to be fed into the arithmetic encoder, generating the final codes.

The process of computing residuals involves only integer operations, eliminating potential precision issues. Assuming the data to be sent is $\mathbf{x}$, the most recent historical data (typically the latest 50 data points in this context) is fed into the predictor to compute a predicted value $\mathbf{x}'$. Given that the receiver (typically the shore side) possesses an identical predictor and historical data, transmitting solely the predicted residual $\mathbf{x} - \mathbf{x}'$ facilitates complete reconstruction of the original data $\mathbf{x}$ at the receiver's end.

3.5. Data-Frequency-Based Arithmetic Coding

Our observations on ship status data reveal varying frequency distributions among values. For instance, in the "Boiler Heavy Oil Density" field, although values range from 825 to 845, a significant portion tends to hover around 835. Leveraging this insight, arithmetic coding algorithms, with the typical ideas assigning shorter encoding for frequently occurring values and longer encoding for occasional ones, are quite suitable to be applied for our ship status data. This approach compresses data further while minimizing information loss, functioning as an almost optimal entropy encoding method [32].

The challenge of applying arithmetic coding algorithm to our scenario lies in the fact that our maritime vessel state data can be viewed as a fixed-dimensional multidimensional vector, whereas arithmetic coding is typically utilized in symbol sequences of variable lengths. Given the diversity of fields in ship status data, each field necessitates an independent frequency table. Subsequently, the code vectors generated for all fields are combined into a single binary string—the final form transmitted to the receiver (typically the shore side). Notably, both sender (typically the ship side) and receiver share estimated data frequencies. During the encoding process, the frequency statistic tables remain static and are not dynamically updated or transmitted. Estimating frequencies for each value within a field is challenging directly, thus necessitating estimation from historical data. For every field in the dataset, the estimated frequency p_i of the i-th value is computed using the formula:

$$p_i = \frac{f_i + 1}{m + l} \tag{4}$$

Here, m represents the total historical data entries, l denotes the possible values in the field, and f_i signifies the occurrence count of the i-th value in the data.

Subsequently, the residuals obtained from the preceding predictive coding step are encoded using these frequency tables and transmitted to the receiver.

4. Experiments and Results

In this section, we introduce our experimental settings and discuss the experiment results we obtained.

4.1. Experimental Setting

4.1.1. Our Used Datasets

Our dataset comprises real sensor monitoring data collected from a ship's energy efficiency monitoring system over a two-month period. We have quantized the dataset according to the precision requirements and numerical limits of actual ship monitoring system, using quantization techniques discussed in Section 3.3. Our ship status data include 41 floating-point fields (Figure 5 shows a part of the data fields), which mainly describe the real-time fuel consumption related data fields, the rotate speed, the torque and the power of the main engine, the ship speed data fields, ship-drought-related data fields, and the real-time weather-related data fields. With data logged approximately every 10 s, this compilation spans 479,742 rows.

DGMGODensity	BlrMGODensity	DG1Power	DG2Power	DG3Power	MERpm	METorque	ShipSpdToWater	ShipSpd	ShipHeel	ShipTrim
279	84	899	727	89	136	586	174	1212	650	693
279	78	905	724	89	136	580	174	1212	667	716
279	82	885	706	89	136	578	173	1204	684	643
279	84	886	711	89	135	580	173	1204	689	641
279	82	871	688	89	136	585	173	1204	717	606
279	82	883	710	89	136	589	174	1190	667	631
279	85	879	698	89	136	585	173	1190	663	697
279	83	872	690	89	136	585	172	1190	646	712
279	84	881	707	89	135	592	171	1190	645	670
279	78	873	686	89	136	606	172	1190	657	691
279	82	871	687	89	136	594	172	1190	688	748
279	83	865	675	89	135	584	172	1190	682	610
279	83	873	695	89	135	592	173	1190	632	646
279	81	881	705	89	136	602	173	1190	668	752
279	82	894	709	89	136	597	173	1190	679	673

Figure 5. The screenshot of our quantized ship status dataset.

In applying our compression algorithm to the ship monitoring system, the number of fields for transmission in the practical application may not precisely match those in the experimental dataset. However, we assume similarity in the nature of the data between the application scenario and our experimental datasets.

To avoid overfitting, we chronologically divided the dataset into training, validation, and test sets, adhering to a 7:2:1 ratio. The training set serves for predictor training in our predictive coding step and for estimating data frequencies applied in arithmetic coding. The validation set aids in adjusting hyperparameters and method comparison. Lastly, the test set is reserved to validate the efficacy of our QAAPA approach, mitigating potential overfitting concerns.

4.1.2. Experimental Environments

Throughout our research, we conducted experiments using various compression methods for ship status data under the following configurations: The primary remote machine[2] boasted robust specifications with an Intel Core i9-10900 CPU[3], Nvidia GeForce RTX 3090 GPU[4], and 128GB of RAM; Operating on Ubuntu 20.04, we utilized Python 3.9, TensorFlow 2.5.0, tmux for prolonged program execution, and a Jupyter server hosted at a research institute under the Chinese Academy of Sciences. We accessed the remoted machine via SSH. After SSH connection, the Jupyter server on the remote machine linked to the local machine's 8888 port. We executed and debugged code using Microsoft Visual Studio Code[5]'s Jupyter Notebook and Python. The high specifications of the remote machine expedited training and model exploration, typically requiring mere minutes for neural network model training. For practical implementation, less resource-intensive servers or cloud services could be employed to curtail costs during model training.

4.1.3. Experimental Design

Our primary objective in the ship status data transmission scenario is to compress real-time data approaching its theoretical minimal length, enabling real-time transmission via the Beidou navigation system. Our optimization focus lies in achieving a long-term average compression rate for compression algorithms, reflected in the average code length post-compression. Given the steps involved in these algorithms—initially estimating symbol occurrence probabilities followed by arithmetic coding—the average code length closely aligns with the information entropy value calculated from these probabilities.

In our approach, we estimate the summed code length of encoding methods using the information entropy of estimated probabilities. This estimation enables us to gauge the compression rate of different algorithms. Sequential information entropy is estimated using commercial compression software, calculated as the compressed file size divided by the number of data points. Notably, the minimum achieved result across various configurations of compression software stands at 141 bit.

We devised a series of experiments to assess the effectiveness and reliability of our designed compression algorithm for real-time ship status data:

- **Experiments to estimate data frequency estimation effectiveness:** We evaluated the validity of the frequency estimation method employed in our study for future data. This analysis involved learning curves and performance comparison on training and validation sets. Estimation used the early part of the training data to gauge frequency and calculate information entropy.
- **Ablation Experiments on our compression algorithm (QAAPA):** We conducted experiments to analyze the key steps in our QAAPA method—Quantization, Attention-Averaging-based Predictive Coding, and Arithmetic Coding. We trained the attention averaging-based predictor by initially training the linear and CNN models separately. The subsequent training of the attention averaging model involved freezing linear model parameters for 20 epochs and using a constant learning rate of 5e-4. We implemented models in Keras with TensorFlow as the backend, trained on an Nvidia Quadro K620 GPU. We compared the summed coding length before and after applying

our QAAPA method across training, validation, and test datasets. Additionally, we assessed individual steps within QAAPA by comparing it with alternative methods:

- Original: only encoding data in *float64* format without data compression
- Q Method: Solely employing Quantization for data compression.
- QA method: Employing Quantization and Arithmetic Coding for data compression.

All of the above methods remove one or more steps from our QAAPA method. The following two methods replace the averaging-attention-based predictor of our QAAPA method:

- QLRPA: Replacing the attention-averaging-based predictor in QAAPA with linear-regression-based predictive coding and including Quantization, linear-regression-based predictive coding, and Arithmetic coding for data compression.
- QCNNPA: Replacing the attention-averaging-based predictor in QAAPA with CNN-based predictive coding and including Quantization, CNN-based Predictive coding, and Arithmetic coding for data compression.

- **Experiments on the reliability of our QAAPA method:** This experiment involved comparing compression results using our QAAPA method across training, validation, and test sets. The compression results are evaluated by the following descriptive statistics, including the mean value, standard deviation, minimum value, 25th percentile, median (50th percentile), 75th percentile, and maximum value. In addition, we analyzed the transmitted code length via Beidou over time to evaluate the reliability of our method.

4.2. Results and Discussion

4.2.1. Results on the Effectiveness of Data Frequency Estimation

Figure 6 shows our frequency estimation evaluation results. The information entropy calculated from the estimated frequency remains stable when the data constitute more than 15% of the total training dataset. This stability across varying data indicates that the frequency estimation method exhibits no significant overfitting tendencies. Conclusively, the frequency estimation derived from historical data demonstrates consistency, validating its suitability for future data and ensuring the reliability of the subsequent arithmetic coding step.

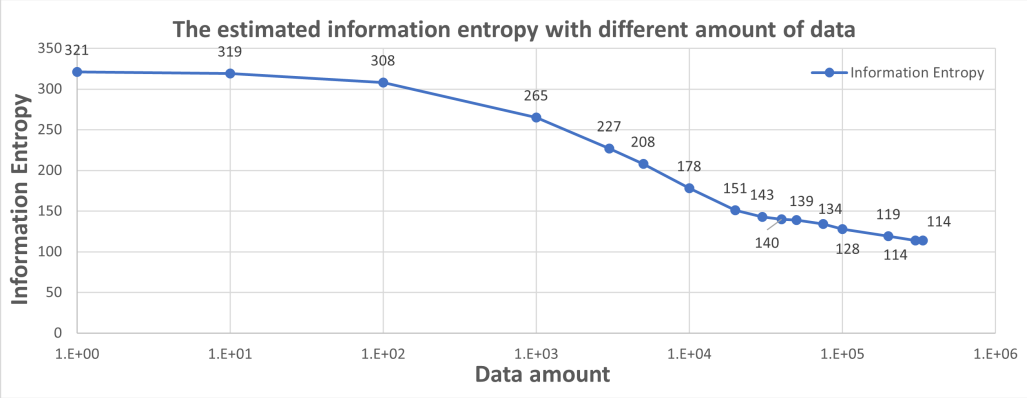


Figure 6. The estimated information entropy with different number of data.

4.2.2. Results on Ablation Experiments on our QAAPA Method

As introduced in Section 4.1.3, the coding lengths are obtained by summing the information entropy of all data fields. Figure 7 shows the separate information entropy of examples data fields. As we can see from it, with our tested dataset, the linear regression predictor is able to provide shorter coding lengths than the CNN predictor for most of the data fields, which indicates that a simple algorithm is sufficient to provide satisfactory compression effectiveness for most cases. However, there are still data fields that can achieve shorter coding lengths using more complicated CNN models, which proves the

necessity to combine different algorithms. As expected, our attention-averaging-based algorithm can achieve the shortest coding lengths by assigning suitable weights for both algorithms for all data fields.

Based on the above-introduced summation method, we can obtain the average coding lengths of using different data compression methods shown in Table 1[6]. As we can see, Quantization alone reduced the average coding length to approximately 40% of the original data size. Incorporating arithmetic coding further halved the average coding length. Introducing predictive coding enhanced compression, with linear-regression-based prediction outperforming CNN-based prediction for most ship status data fields. The complete QAAPA method achieved the most substantial compression, reducing the original data size to less than 10%. These results collectively demonstrate the effectiveness of each step within the QAAPA approach and highlight the superior performance of the complete method in compressing real-time ship status data.

	Coding Lengths		
Data Fields	LR	CNN	Attention Averaging
DG1Power	5.01	8.56	5.01
DG2Power	4.4	8.69	4.4
DG3Power	4.7	8.52	4.7
MERpm	1.52	3.98	1.48
METorque	5.04	7.22	4.9
ShipSpdToWater	1.4	5.42	1.4
ShipSpd	3.39	9.29	3.39
ShipHeel	6.57	6.99	6.57
ShipTrim	6.79	8.39	6.79
WindSpd	4.99	8.26	4.99
_MEActCons	9.33	11.1	9.14
_DGActCons_1	7.77	9.91	7.76
_DGActCons_2	5.48	7.13	3.88
_ShipDraft_m2	6.81	9.56	6.79
_ShipDraught_L_R	6.69	9.54	6.68
_dlt_MEAccFOCons	5.94	3.81	2.36
_dlt_DGAccFOCons	5.34	3.68	1.42
_dlt_BlrAccFOCons	0.33	0.93	0.33
_dlt_DGAccMGOCons	2.69	0.46	0.33
_dlt_BlrAccMGOCons	0.00	0.00	0.00
_dlt_WindDir	4.11	4.05	4.10

Figure 7. The separate information entropy of data fields of the training dataset obtained using linear regression predictor (LR), CNN predictor (CNN), and our proposed attention-averaging-based predictor (Attention Averaging).

Table 1. Ablation study: average coding length of different data compression methods.

Methods	Average Coding Lengths (bits)	
	Training Dataset	Validation Dataset
Original	1312	1312
Q	504	504
QA	263	274
QLRA	122	120
QCNNPA	216	240
QAAPA	**110**	**108**

4.2.3. Experiments on the Reliability of Our QAAPA Method

We evaluated our proposed method in the test set and compared the performance on the test set with those on the train and valid set. The results of the test are shown in Table 2. As we can see from it, the compression effectiveness observed in the test set

remained consistent with the training and validation sets, demonstrating stability and absence of overfitting. The graphical representation of generated code length versus time (Figure 8) illustrates the stability of compression effectiveness over time. The average coding length achieved by the proposed QAAPA method closely aligns with the sequential information entropy estimated earlier, indicating compression results nearing the theoretical minimum. These findings confirm the robustness and reliability of the QAAPA method for compressing real-time ship status data without significant overfitting issues.

Table 2. Average coding length of our QAAPA method.

Dataset	Statistics in Bits						
	Mean	**Std**	**Min**	**25%**	**50%**	**75%**	**Max**
train (70%)	109.96	18	68	96	108	120	360
valid (20%)	108.38	18	68	96	104	120	360
test (10%)	104.10	20	68	88	100	112	348

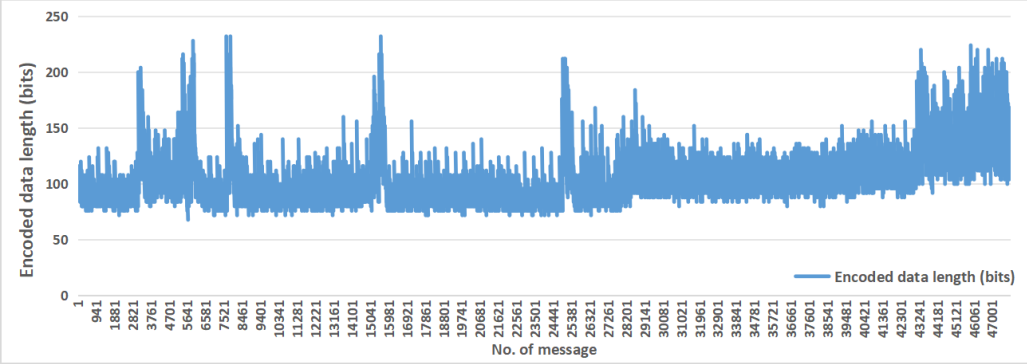


Figure 8. The generated code length versus time.

5. Conclusions and Future Work

This study presents QAAPA, a novel methodology facilitating real-time transmission of ship data via the Beidou Navigation System. The approach involves the real-time compression of sequential ship status data, achieving precision while adhering to predefined data ranges. QAAPA integrates quantization, attention-averaging-based predictive coding, and arithmetic coding algorithms to compress the data stream effectively. Experimental evaluation using real ship monitoring data showcases the efficacy of QAAPA. The results demonstrate its proximity to the theoretical minimum estimated through commercial compression algorithms, surpassing traditional approaches in similar contexts.

Future research avenues include expanding the method's utility in analogous scenarios, such as dynamically adjusting coding accuracy to limit message lengths and integrating incremental learning to eliminate the need for pre-training. Additionally, optimizing the structure of the predictive coding predictor stands as a promising area for enhanced efficiency and effectiveness of the QAAPA approach.

Author Contributions: Conceptualization, C.Z.; methodology, C.Z.; validation, J.Z.; formal analysis, J.Z.; investigation, J.Z.; resources, C.Z. and J.Z.; data curation, C.Z.; writing—original draft preparation, C.Z. and J.Z.; writing—review and editing, J.Z.; visualization, J.Z.; supervision, C.Z.; project administration, C.Z.; and funding acquisition, C.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Key R&D Program of China (2022YFB4301403).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data available on request due to restrictions: The data presented in this study are available on request from the corresponding author.

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

Notes

[1] *m* empirically set to 50.
[2] sourced from Shanghai, China
[3] Manufactured by Intel Corp., Santa Clara, CA, USA
[4] Manufactured by Nvidia Corp., Santa Clara, CA, USA
[5] Version 1.86.0
[6] The bold numbers indicate that our method has the best performance compared to other methods.

References

1. Caban, J.; Brumerčík, F.; Vrábel, J.; Ignaciuk, P.; Misztal, W.; Marczuk, A. Safety of Maritime Transport in the Baltic Sea. *MATEC Web Conf.* **2017**, *134*, 00003. [CrossRef]
2. Yang, X.; Zhi, J.; Zhang, W.; Xu, S.; Meng, X. A Novel Data-Driven Prediction Framework for Ship Navigation Accidents in the Arctic Region. *J. Mar. Sci. Eng.* **2023**, *11*, 2300. [CrossRef]
3. Shao, T.; Du, W.; Ye, Y.; Li, H.; Dong, J.; Liu, G.; Zheng, P. A Novel Virtual Arrival Optimization Method for Traffic Organization Scenarios. *Sustainability* **2024**, *16*, 403. [CrossRef]
4. China Satellite Navigation Office. *Development of the BeiDou Navigation Satellite System (Version 4.0)*; China Satellite Navigation Office: Beijing, China, 2019.
5. China Satellite Navigation Office. *10 Application Scenarios of BDS in Africa*; China Satellite Navigation Office: Beijing, China 2021.
6. Li, G.; Guo, S.; Lv, J.; Zhao, K.; He, Z. Introduction to global short message communication service of BeiDou-3 navigation satellite system. *Adv. Space Res.* **2021**, *67*, 1701–1708. [CrossRef]
7. Nie, X.; Xie, J.; Liu, T.; Liu, C.; Zhao, K. Research on Key Performance of BeiDou Global Short Message Communication Service. In *Proceedings of the China Satellite Navigation Conference*; Springer: Singapore, 2020; pp. 435–442.
8. Xiaoxing, Y.; Ming, C.; Xiaojuan, G. Design of reliable Remote Communication system Based on Beidou Satellite. *Comput. Eng.* **2017**, *34*, 62–68.
9. Kai-wen, X.; Jing, F.; Qian-ru, W. Improvement Mechanism of Beidou Short Message Meteorological and Hydrological Data Transmission Efficiency. *Commun. Inf. Process.* **2021**, *40*, 5.
10. Lanhong, J.; Jinpeng, C.; Zengfeng, B. Application analysis of Beidou Satellite communication system flood exceeding designed level monitoring. *Water Resour. Informatiz.* **2021**, 68–71. [CrossRef]
11. Zhonghui, O.; Huijin, F.; Qinghua, C.; Daochang, H. Research on Compression Transmission Method of Special Vehicle Status Information Based on Beidou Short Message. *J. Ordnance Equip. Eng.* **2020**, *41*, 6.
12. Yue, Y.; Li-juan, Y.; Shou-qi, C.; Cheng-ming, C.; Bing-yi, Z. Design of River Water Quality Monitoring System Based on Beidou Communication. *Acta Metrol. Sin.* **2020**, *41*, 6.
13. Yong, C. Process Data Compression Method Based On Beidou's Short Message Communication. Ph.D. Thesis, Yan Shan University, Qinhuangdao, China, 2017.
14. Yang, Y.; Mandt, S.; Theis, L. An Introduction to Neural Data Compression. *Found. Trends Comput. Graph. Vis.* **2023**, *15*, 113–200. [CrossRef]
15. Li, M.; Zuo, W.; Gu, S.; Zhao, D.; Zhang, D. Learning convolutional networks for content-weighted image compression. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 3214–3223.
16. Possemiers, A.; Lee, I. Evaluating deep learned voice compression for use in video games. *Expert Syst. Appl.* **2021**, *181*, 115180. [CrossRef]
17. Lu, G.; Ouyang, W.; Xu, D.; Zhang, X.; Cai, C.; Gao, Z. Dvc: An end-to-end deep video compression framework. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 11006–11015.
18. Cox, D. Syntactically informed text compression with recurrent neural networks. *arXiv* **2016**, arXiv:1608.02893.
19. Makhoul, J.; Roucos, S.; Gish, H. Vector quantization in speech coding. *Proc. IEEE* **1985**, *73*, 1551–1588. [CrossRef]
20. Goyal, V.K. Theoretical foundations of transform coding. *IEEE Signal Process. Mag.* **2001**, *18*, 9–21. [CrossRef]
21. Yan, Z.; Wang, J.; Sheng, L.; Yang, Z. An effective compression algorithm for real-time transmission data using predictive coding with mixed models of LSTM and XGBoost. *Neurocomputing* **2021**, *462*, 247–259. [CrossRef]
22. Fout, N.; Ma, K.L. An adaptive prediction-based approach to lossless compression of floating-point volume data. *IEEE Trans. Vis. Comput. Graph.* **2012**, *18*, 2295–2304. [CrossRef] [PubMed]
23. Goyal, M.; Tatwawadi, K.; Chandak, S.; Ochoa, I. Deepzip: Lossless data compression using recurrent neural networks. *arXiv* **2018**, arXiv:1811.08162.
24. Gray, R.M.; Neuhoff, D.L. Quantization. *IEEE Trans. Inf. Theory* **1998**, *44*, 2325–2383. [CrossRef]
25. Sprevak, M. Predictive coding I: Introduction. *Philos Compass* **2023**, *19*, e12950. [CrossRef]

26. Géron, A. *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow*; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2022.
27. Hackeling, G. *Mastering Machine Learning with Scikit-Learn*; Packt Publishing Ltd.: Birmingham, UK, 2017.
28. Taye, M.M. Understanding of Machine Learning with Deep Learning: Architectures, Workflow, Applications and Future Directions. *Computers* **2023**, *12*, 91. [CrossRef]
29. Chua, L.O.; Roska, T. The CNN paradigm. *IEEE Trans. Circuits Syst. I Fundam. Theory Appl.* **1993**, *40*, 147–156. [CrossRef]
30. Tulyakov, S.; Gehrig, D.; Georgoulis, S.; Erbach, J.; Gehrig, M.; Li, Y.; Scaramuzza, D. Time Lens: Event-based Video Frame Interpolation. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 16150–16159. [CrossRef]
31. Oord, A.v.d.; Dieleman, S.; Zen, H.; Simonyan, K.; Vinyals, O.; Graves, A.; Kalchbrenner, N.; Senior, A.; Kavukcuoglu, K. Wavenet: A generative model for raw audio. *arXiv* **2016**, arXiv:1609.03499.
32. Witten, I.H.; Neal, R.M.; Cleary, J.G. Arithmetic coding for data compression. *Commun. ACM* **1987**, *30*, 520–540. [CrossRef]