information

Article

A Quaternion Gated Recurrent Unit Neural Network for Sensor Fusion

Uche Onyekpe [1,*], Vasile Palade [1], Stratis Kanarachos [2] and Stavros-Richard G. Christopoulos [2]

[1] Research Centre for Data Science, Coventry University, Coventry CV1 5FB, UK; ab5839@coventry.ac.uk
[2] Faculty of Engineering and Computing, Coventry University, Coventry CV1 5FB, UK; ab8522@coventry.ac.uk (S.K.); ac0966@coventry.ac.uk (S.-R.G.C.)
* Correspondence: onyekpeu@uni.coventry.ac.uk

Abstract: Recurrent Neural Networks (RNNs) are known for their ability to learn relationships within temporal sequences. Gated Recurrent Unit (GRU) networks have found use in challenging time-dependent applications such as Natural Language Processing (NLP), financial analysis and sensor fusion due to their capability to cope with the vanishing gradient problem. GRUs are also known to be more computationally efficient than their variant, the Long Short-Term Memory neural network (LSTM), due to their less complex structure and as such, are more suitable for applications requiring more efficient management of computational resources. Many of such applications require a stronger mapping of their features to further enhance the prediction accuracy. A novel Quaternion Gated Recurrent Unit (QGRU) is proposed in this paper, which leverages the internal and external dependencies within the quaternion algebra to map correlations within and across multidimensional features. The QGRU can be used to efficiently capture the inter- and intra-dependencies within multidimensional features unlike the GRU, which only captures the dependencies within the sequence. Furthermore, the performance of the proposed method is evaluated on a sensor fusion problem involving navigation in Global Navigation Satellite System (GNSS) deprived environments as well as a human activity recognition problem. The results obtained show that the QGRU produces competitive results with almost 3.7 times fewer parameters compared to the GRU.

Keywords: gated recurrent unit; quaternion neural network; quaternion gated recurrent unit; human activity recognition; INS; GPS outage; autonomous vehicle navigation; inertial navigation; neural networks

Citation: Onyekpe, U.; Palade, V.; Kanarachos, S.; Christopoulos, S.-R.G. A Quaternion Gated Recurrent Unit Neural Network for Sensor Fusion. *Information* **2021**, *12*, 117. https://doi.org/10.3390/info12030117

Received: 31 January 2021
Accepted: 1 March 2021
Published: 9 March 2021

1. Introduction

The success of Recurrent Neural Networks (RNNs) on sequentially-based problems has been emphasized in applications such as natural language processing, financial analysis and signal processing problems [1–5]. Other researchers have demonstrated the excellent performance of RNNs on various time series problems such as on electronic health records [6], classifications of acoustic scenes [7], cyber-security [8], human activity recognition [9,10], and vehicular localisation [11–15]. Although RNNs were formulated to model time-dependent relationships within basic sequential problems [16], real-world problems are often multi-dimensional and thus require a dedicated approach towards modelling the relations inherent in the data [17]. Matsui et al. in [18], showed the existence of local relations within the elements of multi-dimensional data. Real-valued methods such as the RNNs, however, approach the multidimensional elements as independent entities within the input vector, where local relations are considered in the same way as global dependencies [17].

Another challenge commonly faced in machine learning is the efficient computation of the representations of large data within the hidden dimensions. It is important for a good model to encode local relations efficiently within the input features, such as the relations between the red, green and blue channels of a pixel as explored in [18,19], and structural

relations across pixels such as edges or shapes. Such efficient representations lead to a significant reduction in the number of neural parameters needed to facilitate the learning process, with also naturally minimised occurrences of overfitting within the model [17].

Quaternions are a number system characterised by one real and three imaginary components that form their hypercomplex structure. Their composition lends them the ability to represent and manipulate features uniquely, thus enabling efficient learning within and across multidimensional input features through the exploitation of the Hamilton product during quaternion algebraic operations [20–22]. Several quaternion-based learning algorithms have been proposed by researchers. Parcollet et al. [23] studied the success of quaternion Convolutional Neural Network (CNN) by investigating the influence of the Hamilton product on colour image reconstruction from gray-scale images. Moya-Sanchez et al. proposed a bio-inspired quaternion local phase CNN layer, offering the possibility of capturing rotational linear response and contrast invariance in image classification as well as faster learning image rotations than a regular convolution layer [24]. Chen et al. [25] studied the use of quaternion-embedded capsule network model for knowledge graph completion. Ozcan et al. proposed a quaternion capsule network in [26], Grassucci et al. proposed a quaternion-valued variational autoencoder in [27], and Nguyen et al. proposed a quaternion graph neural network in [28]. Parcollet et al. used a quaternion-based RNN and LSTM (Long Short-Term Memory) on a challenging natural language processing task [20]. A bidirectional quaternion LSTM recurrent neural network was explored by Parcollet et al. for speech recognition in [29]. However, the Gated Recurrent Unit (GRU) network, a variant of the LSTM is characterised by a less complex structure, making it computationally more efficient compared to the LSTM and justifying its suitability for computationally demanding applications.

A novel Quaternion Gated Recurrent Unit (QGRU) is thus proposed in this paper to leverage the internal and external dependencies within the quaternion algebra in order to map correlations within and across multidimensional features using fewer parameters within the hidden dimensional space. The QGRU is proposed as an improvement on the GRU to better address sensor fusion applications, as it can be used to efficiently capture the inter and intra dependencies within multidimensional features unlike the Gated Recurrent Unit (GRU). The performance of the quaternion formulation of the GRU is investigated comparatively to the GRU on a complex task involving the navigation of autonomous vehicles in challenging environments problems, as addressed in [16,30], and a human activity recognition classification task, as addressed in [31], with the use of time-based signals rather than the frequency transformed signals as used in [31].

The rest of the paper is structured as follows: Section 2 presents a brief literature review on Quaternion Neural Networks, then, in Section 3, we discuss the formulation of the proposed QGRU network, Section 4 presents some experimentation of the QGRU on a challenging vehicular localisation problem as well as a Human Activity Recognition (HAR) task, and it also details the employed datasets. The results obtained on the performance analysis evaluation of the QGRU and GRU are discussed in Section 5, and finally, the paper is concluded in Section 6.

2. Previous Work on Quaternion Neural Networks

In the past decade, the field of complex-valued neural networks has been actively researched, but with limited influence until its recent application to RNNs. Studies show that complex-valued neural networks have better generalisation capabilities [32] and are easier to optimise [33]. Quaternion neural networks were proposed where the inputs and bias vectors, as well as the weight matrices, are quaternion-based. The quaternion-valued vanilla RNN and LSTM were shown to provide improved accuracy with a significantly reduced number of parameters on speech recognition tasks compared to their real-valued counterparts [20]. Several researchers have proposed several quaternion-based learning algorithms with applications to various challenging problems [19–22]. Cui et al. [34] applied the quaternion neural network to the inverse kinematics of a robot manipulator.

Luo et al. [35] compressed colour images using quaternion neural network principal component analysis. Greenblatt et al. in [36] applied quaternion neural networks to prostate cancer Gleason grading. Shang and Hiros [37], proposed a quaternion neural-network-based PolSAR for land classification in Pointcare-sphere-parameter space. Parcollet et al. studied the applications of a deep quaternion neural network to speech recognition [38,39]. Gaudet and Maidat [39], and Parcollet et al. [40] investigated the use of quaternion convolution networks for image processing on the CIFAR and KITTI datasets and an end-to-end automatic speech recognition problem respectively. Pavllo et al. modelled human motion using quarternion-based neural networks [40]. A quaternion convolutional neural network was used by Comminiello et al. to detect and localise 3D sound events in [41]. Zhu et al. proposed a quaternion convolutional neural network for colour image classification and denoising tasks [42]. Tay et al. explored the use of quaternion networks for lightweight and efficient neural natural language processing in [43]. Parcollet et al. investigated the use of quaternion-valued convolutional and recurrent neural networks on speech recognition in [44]. Parcollet et al. studied the use of quaternion neural networks for theme identification of telephone conversations in [45]. Tran et al. proposed a quaternion-based self-attentive long short-term user preference encoding for recommendation in [46]. The localisation of colour image splicing by using a full quaternion convolutional network was explored by Chen et al. in [47]. A deformable quarternion Gabor convolutional neural network for recognition of colour facial expression was proposed by Jin et al. in [48]. Qiu et al. studied the use of quaternion neural networks for multi-channel distant speech recognition in [49]. A hate speech classification model using multi-modal fusion architecture was proposed by Kumar et al. in [50]. However, the quaternion formulation is yet to be extended to the GRU and could find use in computationally constrained sensor fusion applications.

3. Proposed Quaternion Gated Recurrent Unit

This section presents a novel quaternion formulation of the GRU, which formulates the input and bias vectors as well as the weight matrices as quaternions and replaces some of the multiplicative product operators of the GRU with the Hadamard product. The weight initialisation, gated operations and backward propagation mechanism of the QGRU are discussed in this section.

3.1. Real-Valued GRU

The GRU, which was introduced by Cho et al. in 2014 [51], addresses the vanishing gradient problem of the RNN giving it the opportunity to learn long-term dependencies. The cellular operation is characterised by the combination of the input gate and the update gate into a single "update gate". The hidden state and the cell state are also merged to provide a more computationally efficient model compared to the LSTM. The update and reset gate in the GRU operate to tackle the vanishing gradient problem by deciding what information should be passed to the output, thus removing information that is not relevant to the prediction.

The update gate functions to determine the amount of the previous information to be passed along to the future, while the reset gate controls how much of the previous information to forget. Memory content is introduced to store relevant information from the past using the reset gate. The operation of the gates of the GRU are governed by Equations (1)–(4).

$$update\ gate: \ z_t = \sigma(W_z x_t + U_z h_{t-1}) + b_z \tag{1}$$

$$reset\ gate: \ r_t = \sigma(W_r x_t + U_r h_{t-1}) + b_r \tag{2}$$

$$current\ memory\ state: \ \acute{h}_t = tanh(W_h x_t + r_t * U_h h_{t-1}) + b_h \tag{3}$$

$$final\ memory: \ h_t = z_t * h_{t-1} + (1 - z_t) * \acute{h}_t \tag{4}$$

where $*$ is the Hadamard product, h_{t-1} is the previous state, W_z, W_r and W_h are the weight matrices of the update gate, reset gate and current memory state, respectively, U_z, U_r and U_h are the hidden weight matrices of the update gate, reset gate and current memory state respectively, b_z, b_r and b_h are the bias vectors of the update gate, reset gate and current memory state, respectively, x_t is the input feature vector and σ is the sigmoid activation (non-linear) function. Figure 1 shows the GRU's cell structure.

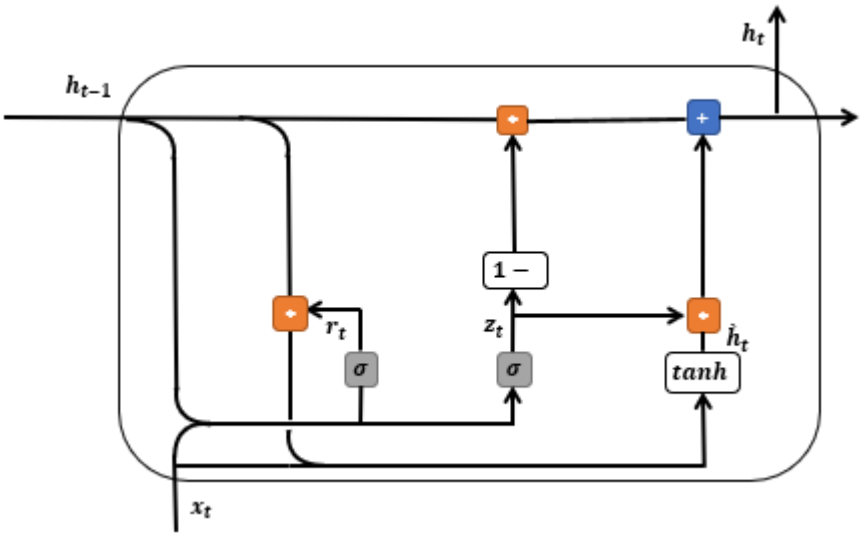

Figure 1. Cell structure of the Gated Recurrent Unit (GRU).

3.2. Quaternion Algebraic Representation and Operations

A quaternion is a four-element vector in the class of hypercomplex numbers composed of a real part and three imaginary parts defined in a four-dimensional space, as expressed in Equation (5).

$$x_Q = x^{(r)} + x^{(i)}i + x^{(j)}j + x^{(k)}k \tag{5}$$

where $x^{(r)}, x^{(i)}, x^{(j)}$ and $x^{(k)}$ are explicit real numbers, x_Q is the quaternion-valued input and i, j and k are the quaternion bases.

Quaternions are further characterised by their ability to satisfy the identities (Hamilton rules) expressed in Equations (6) and (7), establishing their non-commutativity:

$$i^2 = j^2 = k^2 = ijk = 1 \tag{6}$$

$$ij = -ji = k,\ jk = -kj = i,\ ki = -ik = j \tag{7}$$

The conjugate of the quaternion is expressed as:

$$\overline{x}_Q = x^{(r)} - x^{(i)}i - x^{(j)}j - x^{(k)}k \tag{8}$$

The normalised quaternion is expressed as:

$$\dot{x}_Q = \sqrt{x_Q \cdot \overline{x}_Q} = \sqrt{{x^{(r)}}^2 + {x^{(i)}}^2 + {x^{(j)}}^2 + {x^{(k)}}^2} \tag{9}$$

The Hamilton product of two quaternions can be expressed as:

$$\begin{aligned} x_{Q1} \otimes x_{Q2} = &\left(x_1^{(r)}x_2^{(r)} - x_1^{(i)}x_2^{(i)} - x_1^{(j)}x_2^{(j)} - x_1^{(k)}x_2^{(k)}\right) \\ &+\left(x_1^{(r)}x_2^{(i)} + x_1^{(i)}x_2^{(r)} + x_1^{(j)}x_2^{(k)} - x_1^{(k)}x_2^{(j)}\right)i \\ &+\left(x_1^{(r)}x_2^{(j)} - x_1^{(i)}x_2^{(k)} + x_1^{(j)}x_2^{(r)} + x_1^{(k)}x_2^{(i)}\right)j \\ &+\left(x_1^{(r)}x_2^{(k)} + x_1^{(i)}x_2^{(j)} - x_1^{(j)}x_2^{(i)} + x_1^{(k)}x_2^{(r)}\right)k \end{aligned} \tag{10}$$

3.3. Quaternion-Valued Gated Recurrent Unit

A fully connected QGRU has its input, weights, bias and output parameters represented as quaternions. Each variable is broken down into four dimensions representing the four elements of a quaternion $x_Q = x^{(r)} + x^{(i)}i + x^{(j)}j + x^{(k)}k$. Furthermore, the multiplication operator governing the product of the input vector and the weight matrix composed of real-valued elements is replaced by the Hadamard product, as principled by Equation (10). Just like in real-valued layers computations, the fully connected quaternion layers are formulated as matrix multiplications. A sample multiplication is shown in Equation (11).

$$\begin{bmatrix} w^{(r)} & -w^{(i)} & -w^{(j)} & -w^{(k)} \\ w^{(i)} & w^{(r)} & -w^{(k)} & w^{(j)} \\ w^{(j)} & w^{(k)} & w^{(r)} & -w^{(i)} \\ w^{(k)} & -w^{(j)} & w^{(i)} & w^{(r)} \end{bmatrix} \begin{bmatrix} x^{(r)} \\ x^{(i)} \\ x^{(j)} \\ x^{(k)} \end{bmatrix} = \begin{bmatrix} \gamma^{(r)} \\ \gamma^{(i)} \\ \gamma^{(j)} \\ \gamma^{(k)} \end{bmatrix} \tag{11}$$

3.3.1. Weight Initialisation

A successfully trained neural network is dependent on a properly designed weight initialization method. Proper initialisation of the weight parameters is key to the performance of the network, leading to a reduced risk of the vanishing and explosion gradient and an improved convergence. Due to the unique interactions between the weight parameters of a quaternion neural network, a quaternion-valued weight initialisation algorithm used in [20] is used as shown in Equation (12) where w_r, w_i, w_j and w_k are the real and imaginary components of the initialised weights.

$$w_r = \varphi\cos(\theta),\ w_i = \varphi\dot{\mathrm{w}}^{(i)}\sin(\theta),\ w_j = \varphi\dot{\mathrm{w}}^{(j)}\cos(\theta),\ w_k = \varphi\dot{\mathrm{w}}^{(k)}\cos(\theta) \tag{12}$$

where φ is sampled between $-\sigma$ and σ, and σ is established according to the Glorot criterion [35] such that $\sigma = \frac{1}{\sqrt{2(n_{in}+n_{out})}}$, with n_{in} and n_{out} as the number of neurons at the input and output layers; $\dot{\mathrm{w}}^{(i)}$, $\dot{\mathrm{w}}^{(j)}$ and $\dot{\mathrm{w}}^{(k)}$ are the imaginary elements of a normalised imaginary quaternion $\dot{\mathrm{w}}_Q$ as shown in Equations (13)–(16), with the imaginary elements of the base quaternion randomly chosen from a real number between 0 and 1; θ is generated as a random value within $-\pi$ and π.

$$w_Q = 0 + w^{(i)}i + w^{(j)}j + w^{(k)}k \tag{13}$$

$$\overline{w}_Q = 0 - w^{(i)}i - w^{(j)}j - w^{(k)}k \tag{14}$$

$$\dot{\mathrm{w}}_Q = \sqrt{w_Q \cdot \overline{w}_Q} = 0 + \dot{\mathrm{w}}^{(i)}i + \dot{\mathrm{w}}^{(i)}j + \dot{\mathrm{w}}^{(i)}k \tag{15}$$

$$w^{(i)},\ w^{(j)},\ w^{(k)} \leftarrow rand(0,1) \tag{16}$$

3.3.2. Gated Operations

The operations of the gates of the QGRU are governed by Equations (17)–(20). The structure of the QGRU cell is illustrated in Figure 2. The structure of the QGRU remains similar to the GRU, however, the input and output to each cell gate are quaternion-based.

$$update\ gate:\ z_{q,t} = \sigma\left(W_{q,z} \otimes x_{q,t} + U_{q,z} \otimes h_{q,t-1}\right) + b_{q,z} \tag{17}$$

$$reset\ gate:\ r_{q,t} = \sigma\left(W_{q,r} \otimes x_{q,t} + U_{q,r} \otimes h_{q,t-1}\right) + b_{q,r} \tag{18}$$

$$current\ memory\ state:\ \acute{h}_{q,t} = tanh\Big(W_{q,h} \otimes\ x_{q,t} + r_{q,t} * U_{q,h} \otimes h_{q,t-1}\Big) + b_{q,h} \tag{19}$$

$$final\ memory:\ h_{q,t} = z_{q,t} * h_{q,t-1} + \left(1 - z_{q,t}\right) * \acute{h}_{q,t} \tag{20}$$

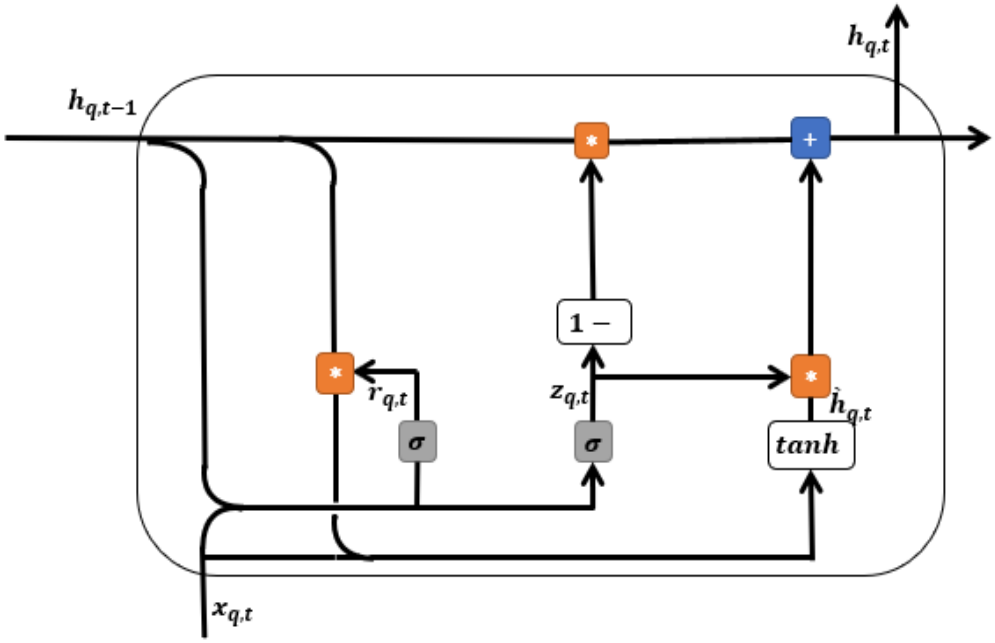

Figure 2. Cell structure of the Quaternion Gated Recurrent Unit (QGRU).

In the above equations, $*$ is the Hadamard product; $h_{q,t-1}$ is the previous quaternionic state; $W_{q,z}$, $W_{q,r}$ and $W_{q,h}$ are the quaternion weight matrices of the update gate, reset gate and current memory state, respectively; $U_{q,z}$, $U_{q,r}$ and $U_{q,h}$ are the hidden weight matrices of the update gate, reset gate and current memory state, respectively;$b_{q,z}$, $b_{q,r}$ and $b_{q,h}$ are the bias vectors of the update gate, reset gate and current memory state, respectively; and $x_{q,t}$ is the quaternionised input features vector.

3.3.3. Quaternion Backward Propagation through Time

The quaternion back-propagation mechanism is adapted from [21]. For each weight matrix, the gradient of the loss e_t with respect to each weight matrix is expressed as shown in Equations (21)–(24), where $\Delta^t_{w_{qy}}$ is the quaternionic representation of the output weight update.

Hidden weights:

$$\Delta^t_{U_{q,z}} = \frac{\partial e_t}{\partial U_{q,z}}, \ \Delta^t_{U_{q,r}} = \frac{\partial e_t}{\partial U_{q,r}}, \ \Delta^t_{U_{q,h}} = \frac{\partial e_t}{\partial \mathrm{U_{q,\,h}}} \tag{21}$$

Input weights:

$$\Delta^t_{w_{q,\,z}} = \frac{\partial e_t}{\partial W_{q,\,z}}, \ \Delta^t_{w_{q,r}} = \frac{\partial e_t}{\partial W_{q,r}}, \ \Delta^t_{w_{q,\,h}} = \frac{\partial e_t}{\partial W_{q,\,h}} \tag{22}$$

Output weights:

$$\Delta^t_{w_{qy}} = \frac{\partial e_t}{\partial W_{qy}} \tag{23}$$

Bias:

$$\Delta^t_{b_{q,z}} = \frac{\partial e_t}{\partial b_{q,z}}, \ \Delta^t_{b_{q,r}} = \frac{\partial e_t}{\partial b_{q,r}}, \ \Delta^t_{b_{q,h}} = \frac{\partial e_t}{\partial b_{q,h}} \tag{24}$$

The gradients can thus be generalised to $\Delta^t = \frac{\partial e_t}{\partial w_q}$ where: $\frac{\partial e_t}{\partial w_q} = \frac{\partial e_t}{\partial w^r} + \frac{\partial e_t}{\partial w^i} i + \frac{\partial e_t}{\partial w^j} j + \frac{\partial e_t}{\partial w^k} k$.

The computation of the loss with respect to each element of the quaternion parameters of the network is done through the application of the chain rule and updated as shown below in Equations (25)–(28).

Hidden weights:

$$U_q = U_q - \lambda \Delta^t_{U,\,q} \tag{25}$$

Input weights:

$$w_q = w_q - \lambda \Delta^t_{w_q} \tag{26}$$

Output weights:

$$w_{qy} = w_{qy} - \lambda \Delta^t_{w_{qy}} \otimes \overline{h}_{q,t} \tag{27}$$

Bias:

$$b_q = b_q - \lambda \Delta^t_{b,\,q} \tag{28}$$

where $\Delta^t_{U,\,q}$, $\Delta^t_{w_q}$ and $\Delta^t_{b,\,q}$ are the generalised forms of the quaternion representations of the hidden weight, input weight and bias update, λ is the learning rate and U_q, w_q, w_{qy} and b_q are the generalised forms of the quaternionic hidden weight matrices, input weight matrices, output weight matrices and bias vectors.

4. QGRU Experiments on Sensor Fusion Applications

This section presents some experiments on evaluating the performance of the QGRU on two sensor fusion applications: the Vehicular Localisation problem in Section 4.1, and the HAR problem in Section 4.2.

4.1. Vehicular Localisation Using Wheel Encoders

The continuous and accurate positioning of autonomous vehicles, road-wise and lane-wise, is critical to their safe performance [52]. In urban canyons, under bridges, tunnels, etc., the visibility of Global Navigation Satellite System (GNSS) is obstructed. Inertial Navigations Systems (INS) and wheel odometers are amongst systems that can be integrated with the GNSS to improve road localisation during GNSS outages. In [30], the wheel encoder was investigated as a replacement to the accelerometer of the INS in tracking the vehicle displacement in challenging GNSS environments, such as Hard-Brake (HB), Wet Road (WR), Successive Left and Right turns and sharp cornering (SLR) [15]. However, the accuracy of the position estimation from the wheel encoder's measurement is affected by factors such as changes in tyre size and wheel slippage. A smaller tyre diameter leads to an under estimation of the vehicle's displacement and vice versa [32]. These uncertainties lead to poor positioning of the vehicles over time as they are cascaded unboundedly during navigation.

Due to the safety-critical nature of this problem, there is however the need to minimize the error drift, thus offering a reliable positioning solution. As such, a localisation solution capable of strongly mapping the features of the motion dynamics to enhance the prediction accuracy of positioning algorithms is needed. The mathematical model of the wheel encoder-based localisation problem is presented in Equations (29)–(36).

The rear left and right wheel's angular velocity (wheel speed) measurements from the wheel encoders are represented as $\hat{\omega}^b_{whrl}$ and $\hat{\omega}^b_{whrr}$, respectively. The errors (uncertainties) corresponding to the left and right rear-wheel speed measurements are defined as ε^b_{whrl} and ε^b_{whrr}. ω^b_{whrr} and ω^b_{whrl} are the wheel speed measurements without errors.

$$\hat{\omega}^b_{whrl} = \omega^b_{whrl} + \varepsilon^b_{whrl} \tag{29}$$

$$\hat{\omega}^b_{whrr} = \omega^b_{whrr} + \varepsilon^b_{whrr} \tag{30}$$

The calculation of the angular velocity of the rear axle is shown in Equations (30)–(31) obtained from the average of the rear left and right wheel measurements.

$$\hat{\omega}^b_{whr} = \frac{\omega^b_{whrr} + \omega^b_{whrl}}{2} + \frac{\varepsilon^b_{whrr} + \varepsilon^b_{whrl}}{2} \tag{31}$$

Taking $\frac{\varepsilon^b_{whrr}+\varepsilon^b_{whrl}}{2}$ as ε^b_{whr} and $\frac{\omega^b_{whrr}+\omega^b_{whrl}}{2}$ as ω^b_{whr}

$$\hat{\omega}^b_{whr} = \omega^b_{whr} + \varepsilon^b_{whr} \tag{32}$$

Using v = wr, the vehicle's linear velocity can be found, with r defined as a constant mapping the speed of the wheel to the vehicle's displacement:

$$v^b_{wh} = \omega^b_{whr} r + \varepsilon^b_{whr} r \tag{33}$$

Taking $\varepsilon^b_{whr} r$ *as* $\varepsilon^b_{whr,v}$

$$v^b_{whr} = \omega^b_{whr} r + \varepsilon^b_{whr,v} \tag{34}$$

The vehicle's displacement can thus be found through the integration of the vehicle's velocity from Equation (34); Where $\varepsilon^b_{whr,x}$ in Equation (35) represents the integral of $\varepsilon^b_{whr,v}$ from Equation (34).

$$x^b_{whr} = \int_{t-1}^{t} (\omega^b_{whr} r) + \varepsilon^b_{whr,x} \tag{35}$$

Here

$$\varepsilon^b_{whr,x} \approx x^b_{whr} - x^b_{GNSS} \tag{36}$$

The vehicle's true displacement is represented as x^b_{GNSS} and calculated according to [53] using the Vincenty's formula for geodesics on an ellipsoid based on the latitudinal and longitudinal information of the vehicle position [53,54].

The focus is on learning to estimate $\varepsilon^b_{whr,x}$ to correct x^b_{whr}. All analysis are done in the body frame as described in [15].

4.1.1. Dataset

The Inertial Odometry Vehicle Navigation Benchmark Dataset (IO-VNBD) [55] is used in the experimentation. The dataset consists of about 98 h of driving data collected over about 5700 km of travel on different driving scenarios. The dataset describes a variant of vehicle motion dynamics using information from sensors such as accelerometers, wheel encoders, gyroscopes, GPS receivers, etc. Although the dataset is collected with a sampling interval of 10 Hz, we down-sampled to a frequency of 1 Hz, as in [30]. The dataset is publicly available at https://github.com/onyekpeu/IO-VNBD (accessed on 30 December 2020) and described in [55]. The training datasets used from the IO-VNBD are V-Vta1a, V-Vta2, V-Vta8, V-Vta10, V-Vta16, V-Vta17, V-Vta20, V-Vta21, V-Vta22, V-Vta27, V-Vta28, V-Vta29, V-Vta30, V-Vtb1, V-Vtb2, V-Vtb3, V-Vtb5, V-Vw4, V-Vw5, V-Vw14b, V-Vw14c, V-Vfa01, V-Vfa02, V-Vfb01a, V-Vfb01b and V-Vfb02b. The test datasets used are as shown in Table 1.

Table 1. Inertial Odometry Vehicle Navigation Benchmark Dataset (IO-VNB) datasets used in the performance evaluation on the localisation task.

Challenging Scenarios	IO-VNB Data Subset
Hard Brake (HB)	V-Vw16b
	V-Vw17
	V-Vta9
Sharp Cornering and Successive Left and Right Turns (SLR)	V-Vw6
	V-Vw7
	V-Vw8
Wet Road (WR)	V-Vtb8
	V-Vtb11
	V-Vtb13

The performance of the QGRU in comparison to the GRU on the localisation problem is evaluated using the maximum CRSE (Cumulative Root Squared Error) metric adopted in [16]. The CRSE is defined as the cumulative root squared of the error estimation of each second for the total duration of the GNSS outage (defined as 10 s). The maximum CRSE from all 10 s length test sequences in each challenging scenario are compared. The CRSE equation is as shown in Equation (37).

$$\text{CRSE} = \sum_{t=1}^{N_t} \sqrt{{e_{pred}}^2} \tag{37}$$

where N_t is GNSS outage length of 10 s, t is the sampling period and e_{pred} is the uncertainty (error) prediction.

4.1.2. Quaternion Features

All input signals are reconstructed by down-sampling the original signals from 10 Hz to 1 Hz and restructured using a sliding window length of 4 per each input signal. The quaternion input feature $x_{Q,t}$ is described in Equation (38).

$$X_{Q,t} = x_{v1} + x_{v2}i + x_{v3}j + x_{v4}k \tag{38}$$

where $v1,\ v2, v3$ and $v4$ refer to the wheel speed information at times $t,\ t-1,\ t-2$ and $t-3$, respectively.

At any time t, the quaternion input feature $X_{Q,t}$ is composed of $X_{Q,1},\ X_{Q,2},\ X_{Q,3}$ and $X_{Q,4}$ as shown in the unrolled architecture of the QGRU in Figure 3. $X_{Q,1},\ X_{Q,2},\ X_{Q,3}$ and $X_{Q,4}$ denote the quaternion inputs at each time step and are defined below such that at time t:

$$X_{Q,1} = x_t + x_{t-1}i + x_{t-2}j + x_{t-3}k \tag{39}$$

$$X_{Q,2} = x_{t-1} + x_{t-2}i + x_{t-3}j + x_{t-4}k \tag{40}$$

$$X_{Q,3} = x_{t-2} + x_{t-3}i + x_{t-4}j + x_{t-5}k \tag{41}$$

$$X_{Q,4} = x_{t-3} + x_{t-4}i + x_{t-5}j + x_{t-6}k \tag{42}$$

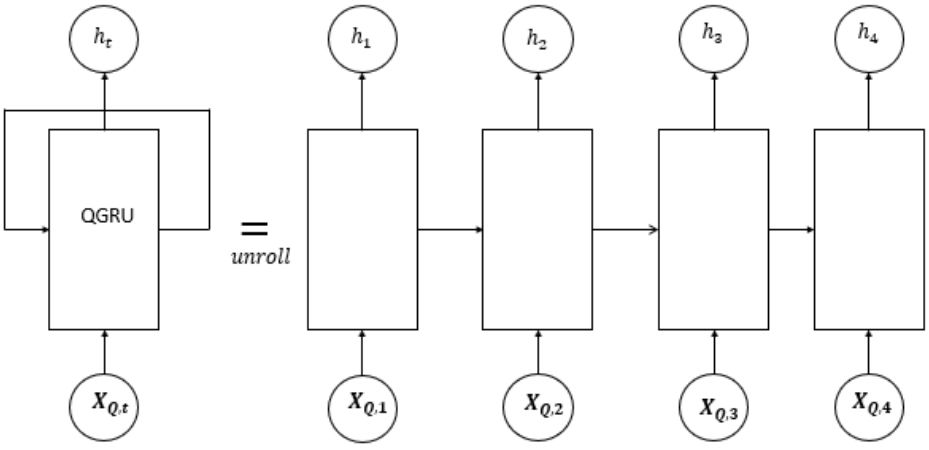

Figure 3. Unrolled QGRU architecture for the vehicular localisation task.

At time $t+1$:

$$X_{Q,1} = x_{t+1} + x_t i + x_{t-1}j + x_{t-2}k \tag{43}$$

$$X_{Q,2} = x_t + x_{t-1}i + x_{t-2}j + x_{t-3}k \tag{44}$$

$$X_{Q,3} = x_{t-1} + x_{t-2}i + x_{t-3}j + x_{t-4}k \tag{45}$$

$$X_{Q,4} = x_{t-2} + x_{t-3}i + x_{t-4}j + x_{t-5}k \tag{46}$$

where x is the wheel speed measurement: ω^b_{whrr} and ω^b_{whrl} that are fed as $X_{Q,t}$ into the neural network to learn the target $\varepsilon^b_{whr,x}$.

As the performance of the QGRU is compared to the GRU in this work, the training process for both the QGRU and GRU are discussed below.

The QGRU training process is done with a single hidden layer with a batch size of 1024 and a recurrent dropout rate of 0.005 applied according to [56]. The model optimization was done using Adamax with an initial learning rate of 0.001. The objective function used is the mean absolute error loss function.

The GRU's training process is also done using a single hidden layer with a batch size of 1024, a recurrent dropout rate of 0.25 and a timestep of 4. The Adamax optimizer is used to optimize the model with an initial learning rate of 0.004. The mean absolute error loss function is also used as the objective function. All input to the QGRU and GRU are normalised to values between 0 and 1.

A varying number of neurons from 4 to 256 are used to compare the performance of the QGRU to the GRU.

4.2. Human Activity Recognition

The identification of different activities performed by humans from sensor data records is an active research topic. Wearable devices, such as smartphones and bracelets, are used to record the actions carried out by humans whilst performing activities such as walking, running, standing, sitting, etc. Information on these activities are used to support domains such as healthcare, home automation and fitness. The challenge, however, lies in the management of the huge amount of information obtained from an array of several sensors as well as their temporal relationships and the lack of knowledge on how to relate the information recorded to the defined activities.

4.2.1. Dataset

The UCI HAR dataset is the second dataset used in our experiments. The dataset, described in [31], is stored in the UCI Machine Learning Repository at http://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones. (accessed on 30 December 2020). The dataset contains information from waist-mounted smartphone sensors, such as the accelerometer and gyroscope at a sampling frequency of 50 Hz. Unlike the IO-VNB Dataset, the signals were pre-processed for noise reduction with a median filter and a 3rd order low-pass Butterworth filter using a cut-off frequency of 20 Hz. The HAR dataset captures static human activities, such as standing, sitting and laying down as well as dynamic human activities, such as walking, walking upstairs and walking downstairs. The training set consists of 70% random samples from the original dataset, while the test set is made up of the remaining 30% of the dataset as used in [31].

4.2.2. Quaternion Features

The shape of the HAR signal is also ordered by time and sampled in sliding windows of 2.56 s (length of 128) and 50% overlap between them. The quaternion input feature at time t denoted as $X_{Q,t}$ is as described in Equation (47).

$$X_{Q,t} = x_{v1} + x_{v2}i + x_{v3}j + x_{v4}k \tag{47}$$

where $v1$, $v2, v3$ and $v4$ refer to each element entry of the quarter divisions of the signal as shown in Equations (48)–(51). As such, $X_{Q,t}$ is made up of $X_{Q,1}$, $X_{Q,2}$, $X_{Q,3}$, ... $X_{Q,32}$ as shown in Figure 4 where $X_{Q,1}$, $X_{Q,2}$, $X_{Q,3}$, ... $X_{Q,32}$ also denote the quaternion input at each time step and are as defined below.

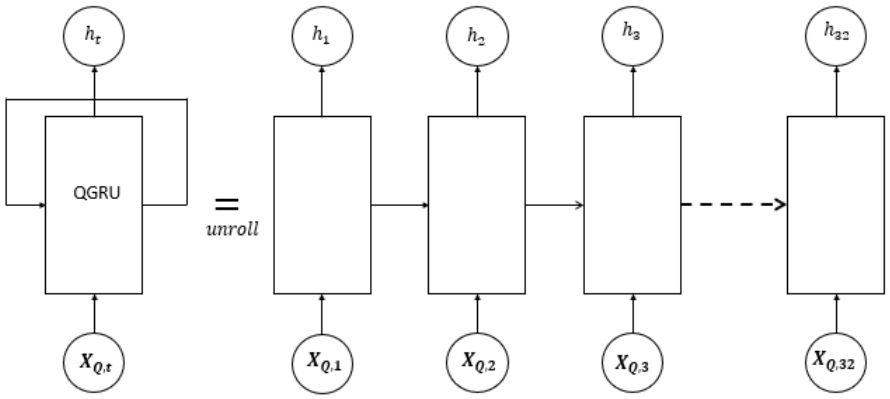

Figure 4. Unrolled QGRU architecture for the Human Activity Recognition (HAR) task.

At every time t:

$$X_{Q,1} = x_{T1} + x_{T33}i + x_{T65}j + x_{T97}k \tag{48}$$

$$X_{Q,2} = x_{T2} + x_{T34}i + x_{T66}j + x_{T98}k \tag{49}$$

$$X_{Q,3} = x_{T3} + x_{T35}i + x_{T67}j + x_{T99}k \tag{50}$$

$$X_{Q,32} = x_{T32} + x_{T64}i + x_{T96}j + x_{T128}k \tag{51}$$

where $T1$, $T2$, $T3 \ldots$.and Tn refer to the first, second, third and nth element entry of the signal and x is an input signal (one of the 9 signals): 3-axis linear acceleration, 3-axis angular velocity and 3-axis jerk information.

The training process of the QGRU is done with a single hidden layer, 300 epochs and a batch size of 1280. The model is optimized using the Adamax optimizer with an initial learning rate of 0.005. The objective function chosen is the mean square error loss function with a dropout rate of 0.005. However, the GRU is trained with a batch size of 4, time step of 128, epoch length of 100, an initial learning rate of 0.002, a categorical cross-entropy loss function, a Stochastic Gradient descent model optimiser and a recurrent dropout rate of 0.25. The neural networks are trained to accurately classify the activity of the human, i.e. standing, walking, laying down, sitting, walking upstairs and walking downstairs. Similarly to the localisation experiment, the performance of the QGRU and the GRU are compared using a varying number of neurons ranging from 4 to 256.

5. Results and Discussion

In this section, the performance of the QGRU and GRU are evaluated on the vehicular localisation problem (regression task) as well as the HAR problem (classification task) described above.

5.1. Challenging Vehicular Localisation Task

The results from the vehicle localisation experiments are presented in Table 2. The performance of the QGRU is compared to the GRU and the physical model (the directly integrated information from the wheel encoder) in estimating the positioning error (uncertainties) $\varepsilon^b_{whr,x}$ needed for the correction of the vehicle's positioning information. The evaluation is done on three challenging scenarios for vehicular positioning in GNSS deprived environments: Hard Brake scenario (HB), sharp cornering and Successive Left and Right turn scenario (SLR), and the Wet Road scenario (WR). With the task of finding the model capable of accurately estimating the positioning uncertainties in each scenario considered, the error in accurately estimating this uncertainty from the QGRU and GRU in comparison to the original uncertainty from the physical model $\varepsilon^b_{whr,x}$ are reported in Table 2. In the hard brake scenario, the QGRU provided the least estimation error of 2.86 m, compared to the GRU's estimation error of 3.15 m and the initial physical model's uncertainty of 9.99 m. The results from the successive left and right turn and sharp cornering

scenario shows that the QGRU also offers the least error in estimating the positioning uncertainty, with an error of 1.24 m compared to the GRU's estimation error of 1.31 m and the original uncertainty of the physical model of 8.19 m. The QGRU performs similarly in the wet road scenario, with the least uncertainty estimation error of 2.09 m compared to 2.36 of the GRU and the physical model's original uncertainty of 5.36 m. The results highlight the QGRU providing an improvement over the GRU of 9.2% in the HB scenario, 5.3% in the SLR scenario and 11.4% in the WR scenario. The results so obtained are in line with those presented in [30]. Remarkably, despite the QGRU providing better estimates compared to the GRU, it does so with fewer of trainable parameters. For instance, in the HB scenario, the QGRU provides better estimates with 3809 parameters compared to 13,121 parameters with the GRU, as shown in Table 3. While in the SLR scenario, the QGRU provided the best estimation with 1137 parameters compared to 3489 parameters of the GRU. Additionally, in the WR scenario, the QGRU estimated the position uncertainty best with 13,761 parameters compared to 50,817 parameters of the GRU.

Table 2. Comparison between the QGRU and GRU on each scenario of the vehicle localisation task.

Number of Neurons	HB (m)			SLR (m)			WR (m)		
	Physical Model	GRU	QGRU	Physical Model	GRU	QGRU	Physical Model	GRU	QGRU
4	9.99	5.16	3.02	8.19	3.46	1.31	5.36	3.3	2.29
8		3.63	2.9		2.16	**1.24**		3.26	2.42
16		3.55	**2.86**		1.8	1.24		3.41	2.24
32		3.52	2.94		**1.31**	1.24		3.38	**2.09**
64		**3.15**	2.94		1.58	1.3		3.42	2.25
128		3.58	3.13		1.32	1.32		**2.36**	2.09
256		3.76	3.14		1.36	1.44		2.48	2.35

Table 3. The number of trainable parameters across various numbers of neurons used in the vehicle localisation experiment.

Number of Neurons	Number of Trainable Parameters	
	GRU	QGRU
4	101	377
8	297	1137
16	977	3809
32	3489	13,761
64	13,121	52,097
128	50,817	202,497
256	199,937	798,209

5.2. Human Activity Recognition (HAR) Task

The performance of the QGRU and GRU on the HAR task across different weighted connections are reported in Table 4. Both neural networks are tasked with accurately classifying the human activities in the HAR dataset, i.e. standing, walking, laying down, sitting, walking upstairs and walking downstairs. The QGRU performs slightly better than the GRU, with a classification accuracy of 95.28% and 95.16%, respectively, which is in line with those presented in [31]. This highlights a 0.08% overall improvement of the QGRU over the GRU. Even so, the QGRU performs better than the GRU in all neuron

numbers experimented with except in the 32 neurons experiment, where the GRU provides a better classification accuracy. Similar to the localisation problem, the QGRU offers a significant parameter reduction in providing the best overall classification accuracy, with 59,015 parameters compared to 206,087 of the GRU, as shown in Table 5.

Table 4. Comparison between the QGRU and GRU performance on the HAR task.

Number of Neurons	Classification Accuracy (%)	
	GRU	QGRU
4	87.51	91.72
8	91.18	92.57
16	92.6	93.62
32	93.62	93.15
64	94.3	**95.28**
128	95.01	95.12
256	**95.16**	95.23

Table 5. The number of trainable parameters across various numbers of neurons used in the HAR task experiment.

Number of Neurons	Number of Trainable Parameters	
	GRU	QGRU
4	203	815
8	495	2007
16	1367	5543
32	4263	17,223
64	14,663	**59,015**
128	53,895	216,327
256	**206,087**	825,063

The performance of the QGRU may be attributed to the quaternion algebra and Hamilton multiplication properties, lending support to a more compact Neural Network formulation. Such reduction in the parametric complexity of the model makes it more suitable for use on low memory embedded devices.

6. Conclusions

This paper proposed a novel Quaternion Gated Recurrent Unit (QGRU) to map multi-dimensional features efficiently using fewer parameters. The QGRU leverages the Hamilton product of quaternions to capture internal and external dependencies efficiently within and across multi-dimensional features. The performance of the QGRU is evaluated over a vehicular localisation problem and a Human Activity Recognition (HAR) task. On the vehicular localisation problem, the QGRU provided the least error in estimating the positioning uncertainty, with a 9.2% improvement over the GRU in the hard brake scenario, a 5.3% improvement the GRU in the sharp cornering and successive left and right turns scenario and an 11.4% improvement over the GRU in the wet road scenario. However, on the HAR task, the QGRU outperforms the GRU with a classification accuracy of 95.28% compared to 95.16% of the GRU. The results obtained from the study show that the QGRU is able to obtain these positioning uncertainty estimates and better classification accuracy compared to the GRU with up to 3.7 times fewer parameters. However, without the use of a carefully designed CUDA kernel, the frequent memory copy operations between the

CPU and GPU during training could cause significant computational delays compared to the GRU.

Our future work will involve an investigation into higher complex-valued neural networks for reduced parametric computations on the sensor fusion problems described in this paper as well as other similar problems.

Author Contributions: Conceptualization, U.O.; methodology, U.O.; validation, V.P., S.K. and S.-R.G.C.; formal analysis, U.O; investigation, U.O; resources, U.O., S.K. and V.P.; data curation, U.O.; writing—original draft preparation, U.O.; writing—review and editing, U.O, V.P., S.K. and S.-R.G.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The IO-VNB dataset is located at https://github.com/onyekpeu/IO-VNBD (accessed on 30 December 2020) and described in [40]. The UCI-HAR dataset is located at http://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones (accessed on 30 December 2020) and described in [33].

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

References

1. Purohit, H.; Tanabe, R.; Ichige, K.; Endo, T.; Nikaido, Y.; Suefusa, K.; Kawaguchi, Y. MIMII dataset: Sound dataset for malfunctioning industrial machine investigation and inspection. *arXiv* **2019**, arXiv:1909.09347.
2. Tsang, G.; Deng, J.; Xie, X. Recurrent neural networks for financial time-series modelling. In *Proceedings of the International Conference on Pattern Recognition, Beijing, China, 20–24 August 2018*; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2018; pp. 892–897. [CrossRef]
3. El-Moneim, S.A.; Nassar, M.A.; Dessouky, M.I.; Ismail, N.A.; El-Fishawy, A.S.; Abd El-Samie, F.E. Text-independent speaker recognition using LSTM-RNN and speech enhancement. *Multimed. Tools Appl.* **2020**, *79*, 24013–24028. [CrossRef]
4. Mao, W.; Wang, M.; Sun, W.; Qiu, L.; Pradhan, S.; Chen, Y.-C. RNN-based room scale hand motion tracking. In *Proceedings of the 25th Annual International Conference on Mobile Computing and Networking, Los Cabos, Mexico, 21–25 October 2019*; Association for Computing Machinery (ACM): New York, NY, USA, 2019; Volume 19, pp. 1–16. [CrossRef]
5. Senturk, U.; Yucedag, I.; Polat, K. Repetitive neural network (RNN) based blood pressure estimation using PPG and ECG signals. In *Proceedings of the ISMSIT 2018—2nd International Symposium on Multidisciplinary Studies and Innovative Technologies, Ankara, Turkey, 19–21 October 2018*; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2018. [CrossRef]
6. Rajkomar, A.; Oren, E.; Chen, K.; Dai, A.M.; Hajaj, N.; Hardt, M.; Liu, P.J.; Liu, X.; Marcus, J.; Sun, M.; et al. Scalable and accurate deep learning with electronic health records. *NPJ Digit. Med.* **2018**, *1*, 18. [CrossRef] [PubMed]
7. Nwe, T.L.; Dat, T.H.; Ma, B. Convolutional neural network with multi-task learning scheme for acoustic scene classification. In *Proceedings of the 9th Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA ASC 2017, Kuala Lumpur, Malaysia, 12–15 December 2017*; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2018; pp. 1347–1350. [CrossRef]
8. Susto, G.A.; Cenedese, A.; Terzi, M. Time-series classification methods: Review and Applications to power systems data. In *Big Data Application in Power Systems*; Elsevier: Amsterdam, The Netherlands, 2018; pp. 179–220. ISBN 9780128119693.
9. Nweke, H.F.; Teh, Y.W.; Al-garadi, M.A.; Alo, U.R. Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges. *Expert Syst. Appl.* **2018**, *105*, 233–261. [CrossRef]
10. Wang, J.; Chen, Y.; Hao, S.; Peng, X.; Hu, L. Deep learning for sensor-based activity recognition: A survey. *Pattern Recognit. Lett.* **2019**, *119*, 3–11. [CrossRef]
11. Chen, C.; Lu, X.; Markham, A.; Trigoni, N. IONet: Learning to cure the curse of drift in inertial odometry. *arXiv* **2018**, arXiv:1802.02209.
12. Dai, H.F.; Bian, H.W.; Wang, R.Y.; Ma, H. An INS/GNSS integrated navigation in GNSS denied environment using recurrent neural network. *Def. Technol.* **2019**. [CrossRef]
13. Fang, W.; Jiang, J.; Lu, S.; Gong, Y.; Tao, Y.; Tang, Y.; Yan, P.; Luo, H.; Liu, J. A LSTM algorithm estimating pseudo measurements for aiding INS during GNSS Signal outages. *Remote Sens.* **2020**, *12*, 256. [CrossRef]
14. Brossard, M.; Barrau, A.; Bonnabel, S. AI-IMU dead-reckoning. *IEEE Trans. Intell. Veh.* **2020**. [CrossRef]
15. Onyekpe, U.; Palade, V.; Kanarachos, S. Learning to localise automated vehicles in challenging environments using Inertial Navigation Systems (INS). *Appl. Sci.* **2021**, *11*, 1270. [CrossRef]
16. Schuster, M.; Paliwal, K.K. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.* **1997**, *45*, 2673–2681. [CrossRef]
17. Parcollet, T.; Morchid, M.; Linarès, G. A survey of quaternion neural networks. *Artif. Intell. Rev.* **2020**, *53*, 2957–2982. [CrossRef]
18. Matsui, N.; Isokawa, T.; Kusamichi, H.; Peper, F.; Nishimura, H. Quaternion neural network with geometrical operators. *J. Intell. Fuzzy Syst.* **2004**, *15*, 149–164.

19. Kusamichi, H.; Kusamichi, H.; Isokawa, T.; Isokawa, T.; Matsui, N.; Matsui, N.; Ogawa, Y.; Ogawa, Y.; Maeda, K.; Maeda, K. A new scheme for color night vision by quaternion neural network. In Proceedings of the 2nd International Conference on Autonomous Robots and Agents (ICARA2004), Palmerston North, New Zealand, 13–15 December 2004; pp. 101–106.
20. Parcollet, T.; Ravanelli, M.; Morchid, M.; Linarès, G.; Trabelsi, C.; De Mori, R.; Bengio, Y. Quaternion Recurrent Neural Networks. 2019. Available online: https://github.com/Orkis-Research/Pytorch-Quaternion-Neural-Networks (accessed on 16 June 2020).
21. Choi, J.; Wang, Z.; Venkataramani, S.; Chuang, P.I.-J.; Srinivasan, V.; Gopalakrishnan, K. PACT: Parameterized Clipping Activation for Quantized Neural Networks. *arXiv* **2018**, arXiv:1805.06085.
22. Isokawa, T.; Kusakabe, T.; Matsui, N.; Peper, F. Quaternion neural network and its application. In *Proceedings of the Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science)*; Springer: Berlin/Heidelberg, Germany, 2003; Voluem 2774, Part 2, pp. 318–324. [CrossRef]
23. Parcollet, T.; Morchid, M.; Linares, G. Quaternion convolutional neural networks for heterogeneous image processing. In *Proceedings of the ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing, Brighton, UK, 12–17 May 2019*; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2019; pp. 8514–8518. [CrossRef]
24. Moya-Sánchez, E.U.; Xambó-Descamps, S.; Sánchez Pérez, A.; Salazar-Colores, S.; Martínez-Ortega, J.; Cortés, U. A bio-inspired quaternion local phase CNN layer with contrast invariance and linear sensitivity to rotation angles. *Pattern Recognit. Lett.* **2020**, *131*, 56–62. [CrossRef]
25. Chen, H.; Wang, W.; Li, G.; Shi, Y. A quaternion-embedded capsule network model for knowledge graph completion. *IEEE Access* **2020**, *8*, 100890–100904. [CrossRef]
26. Özcan, B.; Kınlı, F.; Kıraç, F. Quaternion Capsule Networks. *arXiv* **2020**. Available online: https://github.com/Boazrciasn/Quaternion-Capsule-Networks.git (accessed on 24 February 2021).
27. Grassucci, E.; Comminiello, D.; Uncini, A. QUATERNION-VALUED VARIATIONAL AUTOENCODER. *arXiv* **2020**, arXiv:2010.11647v1.
28. Nguyen, D.Q.; Nguyen, T.D.; Phung, D. Quaternion graph neural networks. *arXiv* **2020**. Available online: https://github.com/daiquocnguyen/QGNN (accessed on 24 February 2021).
29. Parcollet, T.; Morchid, M.; Linares, G.; De Mori, R. Bidirectional quaternion long short-term memory recurrent neural networks for speech recognition. In *Proceedings of the ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing, Brighton, UK, 12–17 May 2019*; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2019; pp. 8519–8523.
30. Onyekpe, U.; Kanarachos, S.; Palade, V.; Christopoulos, S.-R.G. Learning uncertainties in wheel odometry for vehicular localisation in GNSS deprived environments. In Proceedings of the International Conference on Machine Learning Applications (ICMLA), Miami, FL, USA, 14–17 December 2020; pp. 741–746.
31. Anguita, D.; Ghio, A.; Oneto, L.; Parra, X.; Reyes-Ortiz, J.L. A public domain dataset for human activity recognition using smartphones. In Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, Bruges, Belgium, 24–26 April 2013.
32. Hirose, A.; Yoshida, S. Generalization characteristics of complex-valued feedforward neural networks in relation to signal coherence. *IEEE Trans. Neural Networks Learn. Syst.* **2012**, *23*, 541–551. [CrossRef]
33. Nitta, T. On the critical points of the complex-valued neural network. In *Proceedings of the ICONIP 2002 9th International Conference on Neural Information Processing: Computational Intelligence for the E-Age, Singapore, 18–22 November 2002*; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2002; Volume 3, pp. 1099–1103. [CrossRef]
34. Yao, W.; Zhou, D.; Zhan, L.; Liu, Y.; Cui, Y.; You, S.; Liu, Y. GPS signal loss in the wide area monitoring system: Prevalence, impact, and solution. *Electr. Power Syst. Res.* **2017**, *147*, 254–262. [CrossRef]
35. Luo, L.; Feng, H.; Ding, L. Color image compression based on quaternion neural network principal component analysis. In Proceedings of the 2010 International Conference on Multimedia Technology, ICMT 2010, Ningbo, China, 29–31 October 2010. [CrossRef]
36. Greenblatt, A.; Mosquera-Lopez, C.; Agaian, S. Quaternion neural networks applied to prostate cancer gleason grading. In Proceedings of the 2013 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2013, Manchester, UK, 13–16 October 2013; pp. 1144–1149. [CrossRef]
37. Shang, F.; Hirose, A. Quaternion neural-network-based PolSAR land classification in poincare-sphere-parameter space. *IEEE Trans. Geosci. Remote Sens.* **2014**, *52*, 5693–5703. [CrossRef]
38. Parcollet, T.; Morchid, M.; Linares, G. Deep quaternion neural networks for spoken language understanding. In *Proceedings of the 2017 IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2017, Okinawa, Japan, 16–20 December 2017*; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2018; pp. 504–511. [CrossRef]
39. Parcollet, T.; Morchid, M.; Linarès, G. Quaternion Denoising Encoder-Decoder for Theme Identification of Telephone Conversations. 2017. 3325–3328. Available online: https://hal.archives-ouvertes.fr/hal-02107632 (accessed on 30 December 2020). [CrossRef]
40. Pavllo, D.; Feichtenhofer, C.; Auli, M.; Grangier, D. Modeling human motion with quaternion-based neural networks. *Int. J. Comput. Vis.* **2020**, *128*, 855–872. [CrossRef]
41. Comminiello, D.; Lella, M.; Scardapane, S.; Uncini, A. Quaternion convolutional neural networks for detection and localization of 3D sound events. In *Proceedings of the ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing, Brighton, UK, 12–17 May 2019*; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2019; pp. 8533–8537.

42. Zhu, X.; Xu, Y.; Xu, H.; Chen, C. Quaternion Convolutional Neural Networks. 2019. Available online: https://arxiv.org/abs/1903.00658 (accessed on 30 December 2020).
43. Tay, Y.; Zhang, A.; Tuan, L.A.; Rao, J.; Zhang, S.; Wang, S.; Fu, J.; Hui, S.C. Lightweight and efficient neural natural language processing with quaternion networks. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 1494–1503.
44. Parcollet, T.; Ravanelli, M.; Morchid, M.; Linarès, G.; De Mori, R. Speech recognition with quaternion neural networks. *arXiv* **2018**, arXiv:1811.09678.
45. Parcollet, T.; Morchid, M.; Linares, G.; De Mori, R. Quaternion convolutional neural networks for theme identification of telephone conversations. In *Proceedings of the 2018 IEEE Spoken Language Technology Workshop, SLT 2018, Athens, Greece, 18–21 December 2018*; Institute of Electrical and Electronics Engineers Inc.: New York, NY, USA, 2019; pp. 685–691. [CrossRef]
46. Tran, T.; You, D.; Lee, K. Quaternion-based self-attentive long short-term user preference encoding for recommendation. In *Proceedings of the International Conference on Information and Knowledge Management, Galway, Ireland, 19–23 October 2020*; Association for Computing Machinery: New York, NY, USA, 2020; pp. 1455–1464. [CrossRef]
47. Chen, B.; Gao, Y.; Xu, L.; Hong, X.; Zheng, Y.; Shi, Y.-Q. Color image splicing localization algorithm by quaternion fully convolutional networks and superpixel-enhanced pairwise conditional random field. *MBE* **2019**, *16*, 6907–6922. [CrossRef]
48. Jin, L.; Zhou, Y.; Liu, H.; Song, E. Deformable quaternion gabor convolutional neural network for color facial expression recognition. In *Proceedings of the International Conference on Image Processing, ICIP, Abu Dhabi, United Arab Emirates, 25–28 October 2020*; IEEE Computer Society: Washington, DC, USA, 2020; pp. 1696–1700. [CrossRef]
49. Qiu, X.; Parcollet, T.; Ravanelli, M.; Lane, N.; Morchid, M. Quaternion neural networks for multi-channel distant speech recognition. In Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH, Shanghai, China, 14–18 September 2020; International Speech Communication Association: Baixas, France, 2020; pp. 329–333. [CrossRef]
50. Kumar, D.; Kumar, N.; Mishra, S. QUARC: Quaternion multi-modal fusion architecture for hate speech classification. *arXiv* **2020**. Available online: https://github.com/smlab-niser/quaternionFusion (accessed on 24 February 2021).
51. Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In Proceedings of the EMNLP 2014—2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 25–29 October 2014. [CrossRef]
52. Onyekpe, U.; Kanarachos, S.; Palade, V.; Christopoulos, S.-R.G. Vehicular localisation at high and low estimation rates during GNSS outages: A deep learning approach. In *Deep Learning Applications, Volume 2. Advances in Intelligent Systems and Computing*; Wani, M.A., Khoshgoftaar, T.M., Palade, V., Eds.; Springer: Singapore, 2020; Volume 1232, pp. 229–248. ISBN 978-981-15-6758-2.
53. Vincenty, T. Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations. *Surv. Rev.* **1975**, *23*, 88–93. [CrossRef]
54. Pietrzak, M. Vincenty · PyPI. Available online: https://pypi.org/project/vincenty/ (accessed on 12 April 2019).
55. Onyekpe, U.; Palade, V.; Kanarachos, S.; Szkolnik, A. IO-VNBD: Inertial and odometry benchmark dataset for ground vehicle positioning. *Data Br.* **2021**, *35*, 106885. [CrossRef] [PubMed]
56. Gal, Y.; Ghahramani, Z. A theoretically grounded application of dropout in recurrent neural networks. *arXiv* **2016**, arXiv:1512.05287.