# DEEP QUATERNION NEURAL NETWORKS FOR SPOKEN LANGUAGE UNDERSTANDING

*Titouan Parcollet*[1,2]*, Mohamed Morchid*[1]*, Georges Linarès*[1]

[1]LIA, University of Avignon (France)
[2]Orkis (France)
{firstname.lastname}@univ-avignon.fr

## ABSTRACT

Deep Neural Networks (DNN) received a great interest from researchers due to their capability to construct robust abstract representations of heterogeneous documents in a latent subspace. Nonetheless, mere real-valued deep neural networks require an appropriate adaptation, such as the convolution process, to capture latent relations between input features. Moreover, real-valued deep neural networks reveal little in way of document internal dependencies, by only considering words or topics contained in the document as an isolate basic element. Quaternion-valued multi-layer perceptrons (QMLP), and autoencoders (QAE) have been introduced to capture such latent dependencies, alongside to represent multidimensional data. Nonetheless, a three-layered neural network does not benefit from the high abstraction capability of DNNs. The paper proposes first to extend the hyper-complex algebra to deep neural networks (QDNN) and, then, introduces pre-trained deep quaternion neural networks (QDNN-AE) with dedicated quaternion encoder-decoders (QAE). The experiments conduced on a theme identification task of spoken dialogues from the DECODA data set show, inter alia, that the QDNN-AE reaches a promising gain of $2.2\%$ compared to the standard real-valued DNN-AE.

***Index Terms*—** Quaternions, deep neural networks, spoken language understanding, autoencoders, machine learning.

## 1. INTRODUCTION

Deep neural networks (DNN) have become ubiquitous in a broad spectrum of domains specific applications, such as image processing [1, 2], speech recognition [3], or spoken language understanding (SLU) [4]. State-of-the art approaches involve different neural-based structures to construct abstract representations of documents in a low dimensional subspace, such as deep neural networks [5], recurrent neural networks (RNN)[6, 7, 8, 9], convolutional neural networks (CNN)[1], and, more recently, generative adversarial neural networks (GAN)[10]. However, in a standard real-valued neural structure, the latent relations between input features are difficult to represent. Indeed, multidimensional features require to be reduced to a one dimensional vector before the learning process, while an appropriate solution is to process a multidimensional input as a single homogeneous entity. In other words, real-valued representations reveal little in way of document internal structure by only considering words or topics contained in the document as an isolate basic element. Therefore, quaternion multi-layer perceptrons (QMLP) [11, 12, 13] and quaternion autoencoders (QAE) [14] have been introduced to capture such latent dependencies, thanks to the fourth dimensionality of hyper-complex numbers alongside to the Hamilton product [15]. Nonetheless, previous quaternion-based studies focused on three-layered neural networks, while the efficiency and the effectiveness of DNN have already been demonstrated [16, 5].

Therefore, this paper proposes first to extend QMLPs to deep quaternion neural networks (QDNN) for theme identification of telephone conversations. Indeed the high abstraction capability of DNNs added to the quaternion latent relation representations, fully expose the potential of hyper-complex based neural structures. Nevertheless, in [17], the authors highlighted the non-local optimum convergence, and the high overfitting probability of training deep neural networks. To alleviate these weaknesses, different methods have been proposed, such as adding noises during learning to prevent the overfitting phenomenon[18], or a pre-training process to easily converge to a non-local optimum [19], with a Restricted Boltzman Machine (RBM) [20] or an encoder-decoder neural network (AE) [21].

The paper proposes then to compare the randomly initialized QDNN with a greedy layer-wise pre-trained QDNN using QAE, called "QDNN-AE", to fully expose the quaternion deep neural structure capabilities during a SLU task. The experiments are conduced on the DECODA telephone conversations framework and show a promising gain of the QDNN compared to the QMLP. Moreover, the experiments underline the impact of pre-training with a dedicated autoencoder for a QDNN. Finally the proposed quaternion based models are compared to the real-valued ones.

The rest of the paper is organized as follows: Section 2 presents the quaternion deep neural networks and quater-

nion encoder-decoders and Section 3 details the experimental protocol. The results are discussed in Section 4 before concluding on Section 5

## 2. DEEP QUATERNION NEURAL NETWORKS (QDNN) AND QUATERNION AUTOENCODERS (QAE)

The proposed QDNN combines the well-known real-valued deep neural network [1] with the Quaternion algebra. Section 2.1 details the quaternion fundamental properties required to define and understand the QDNN algorithms presented in Section 2.2.

### 2.1. Quaternion algebra

The quaternion algebra $\mathbb{Q}$ is an extension of the complex numbers defined in a four dimensional space as a linear combination of four basis elements denoted as $1, \mathbf{i}, \mathbf{j}, \mathbf{k}$ to represent a rotation. A quaternion $Q$ is written as:

$$Q = r1 + x\mathbf{i} + y\mathbf{j} + z\mathbf{k} \tag{1}$$

In a quaternion, $r$ is its real part while $x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$ is the imaginary part ($I$) or the vector part. There is a set of basic Quaternion properties needed for the QDNN definition:

- all products of $\mathbf{i}, \mathbf{j}, \mathbf{k}$

$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1$$

- quaternion conjugate $Q^*$ of $Q$ is: $Q^* = r1 - x\mathbf{i} - y\mathbf{j} - z\mathbf{k}$
- dot product between two quaternions $Q_1$ and $Q_2$ is $\langle Q_1, Q_2 \rangle = r_1 r_2 + x_1 x_2 + y_1 y_2 + z_1 z_2$
- quaternion norm: $|Q| = \sqrt{r^2 + x^2 + y^2 + z^2}$
- normalized quaternion $Q^\triangleleft = \frac{Q}{|Q|}$
- Hamilton product $\otimes$ between $Q_1$ and $Q_2$ encodes latent dependencies and is defined as follows:

$$\begin{aligned} Q_1 \otimes Q_2 =& (r_1 r_2 - x_1 x_2 - y_1 y_2 - z_1 z_2) + \\ & (r_1 x_2 + x_1 r_2 + y_1 z_2 - z_1 y_2)\mathbf{i} + \\ & (r_1 y_2 - x_1 z_2 + y_1 r_2 + z_1 x_2)\mathbf{j} + \\ & (r_1 z_2 + x_1 y_2 - y_1 x_2 + z_1 r_2)\mathbf{k} \end{aligned} \tag{2}$$

This encoding capability has been confirmed by [22]. Indeed, the authors have demonstrated the rotation, transformation and scaling capabilities of a single quaternion due to the Hamilton product. Moreover, it performs an interpolation between two rotations following a geodesic over a sphere in the $\mathbb{R}^3$ space.

[1] A multilayer perceptron (MLP) with more than one hidden layer

Given a segmentation $S = \{s_1, s_2, s_3, s_4\}$ of a document $p \in P$ depending on the document segmentation detailed in [12] and a set of topics from a latent Dirichlet allocation (LDA) [23] $z = \{z_1, \ldots, z_i, \ldots, z_T\}$, each topic $z_i$ in a document $d$ is represented by the quaternion:

$$Q_p(z_i) = x_d - p^1(z_i)1 + x_p^2(z_i)i + x_p^3(z_i)j + x_p^4(z_i)k \tag{3}$$

where $x_p^m(z_i)$ is the prior of the topic $z_i$ in segment $s_m$ of a document $p$ as described in [12]. This quaternion is then normalized to obtain the input $Q_p^\triangleleft(z_i)$ of QMLPs.

More about hyper-complex numbers can be found in [24, 25, 26] and more precisely about quaternion algebra in [27].

### 2.2. Deep Quaternion Neural Networks (QDNN)

This section details the QDNN algorithms and structure (Figure 1). QDNN differs from the real-valued DNN in each learning subprocess, and all elements of the QDNN (inputs $Q_p$, labels $t$, weights $w$, biases $b$, outputs $\gamma$, ...) are quaternions:

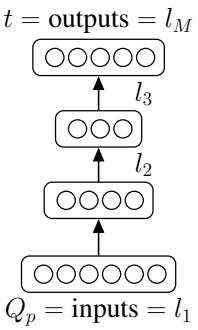


**Fig. 1**. Illustration of a Quaternion Deep Neural Network with 2 hidden layers ($M = 4$).

**Activation function**
The activation function $\beta$ is the split [11] ReLU function ($\alpha$), applied to each element of the quaternion $Q = r1 + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$ as follows:

$$\beta(x) = \alpha(r)1 + \alpha(x)\mathbf{i} + \alpha(y)\mathbf{j} + \alpha(z)\mathbf{k} \tag{4}$$

Where $\alpha$ is

$$\alpha(x) = Max(0, x) \tag{5}$$

**Forward phase**
Let $N_l$ be the number of neurons contained in the layer $l$ ($1 \leq l \leq L$) and $L$ be the number of layers of the QDNN including the input and the output layers. $\theta_n^l$ is the bias of the neuron $n$ ($1 \leq n \leq N_l$) from the layer $l$. Given a set of $P$ normalized quaternion input patterns $Q_p^\triangleleft$ ($1 \leq p \leq P$, denoted as $Q_p$ for convenience in the rest of the paper) and a set of labels

$t_p$ associated to each input $Q_p$, the output $\gamma_n^l$ ($\gamma_n^0 = Q_p^n$ and $\gamma_n^M = t_n$) of the neuron $n$ from the layer $l$ is given by:

$$\gamma_n^l = \beta(S_n^l)$$
$$\text{with } S_n^l = \sum_{m=0}^{N_{l-1}} w_{nm}^l \otimes \gamma_m^{l-1} + \theta_n^l \tag{6}$$

**Learning phase**

The error $e$ observed between the expected outcome $y$ and the result of the forward phase $\gamma_M$ is then evaluated for the output layer ($l = M$) as follows:

$$e_n^l = t_n - \gamma_n^l \tag{7}$$

and for the hidden layer ($2 \leq l < M - 1$)

$$e_n^l = \sum_{h=1}^{N_{l+1}} w_{h,n}^{*l+1} \otimes \delta_h^{l+1} \ , \tag{8}$$

The gradient $\delta$ is computed with

$$\delta_n^l = e_n^l \times \frac{\partial \beta(S_n^l)}{\partial S_n^l} \text{ where } \frac{\partial \beta(S_n^l)}{\partial S_n^l} = \begin{cases} 1, & \text{if } S_n^l > 0 \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

**Update phase**

The weights $w_{n,m}^l$ and the bias values $\theta_n^l$ have to be respectively updated to $\widehat{w}_{n,m}^l$ and $\widehat{\theta}_n^l$:

$$\widehat{w}_{n,m}^l = w_{n,m}^l + \epsilon \delta_n^l \otimes \beta^\star(S_n^l) \tag{10}$$
$$\widehat{\theta}_n^l = \theta_n^l + \epsilon \delta_n^l \ . \tag{11}$$

### 2.3. Quaternion Autoencoder (QAE)

The QAE [14] is a three-layered ($M = 3$) neural network made of an encoder and a decoder where $N_1 = N_3$ as depicted in Figure 2. The well-known autoencoder (AE) is obtained with the same algorithm than the QAE, but with real numbers, and the Hamilton product is replaced with the mere dot product.

Given a set of $P$ normalized inputs $Q_p$ ($1 \leq p \leq P$), the **encoder** computes an hidden representation $l_2$ of $Q_p$, while the **decoder** attempts to reconstruct the input vector $Q_p$ from this hidden vector from $l_2$ to obtain the output vector $\tilde{Q}_p$. The learning phase follows the algorithm previously described in Section 2.2. Indeed, the QAE attempts to reduce the reconstruction error $e_{\text{MSE}}$ between $\tilde{Q}_p$ and $Q_p$ by using the traditional Mean Square Error (MSE) [19] between all $m$ ($1 \leq m \leq N_1$) quaternions $Q_m$ and estimated $\tilde{Q}_m$ composing the pattern $Q_p$:

$$e_{\text{MSE}}(\tilde{Q}_m, Q_m) = ||\tilde{Q}_m - Q_m||^2 \tag{12}$$

to minimize the total reconstruction error $L_{\text{MSE}}$

$$L_{\text{MSE}} = \frac{1}{P} \sum_{p \in P} \sum_{m \in M} e_{\text{MSE}}(\tilde{Q}_m, Q_m). \tag{13}$$

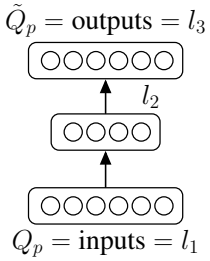


**Fig. 2**. Illustration of a quaternion autoencoder.

### 2.4. QDNN initialized with dedicated QAEs

Deep neural networks learning process is impacted by a wide variety of issues related to the large number of parameters [3], such as the vanishing or exploding gradient, and the over-fitting phenomenon. Different techniques have been proposed to address thes drawbacks [18, 28, 29, 30]: additive noises, normalization preprocessing, adaptive learning rates, and pre-training. The pre-training process allows the neural network structure to converge faster using a pre-learning phase in an unsupervised task, to a non-local optimum. Indeed, an autoencoder is employed for learning the weight matrices composing the QDNN, except the last one that is randomly initialized, as illustrated in Figure 3-(b)-(c). Therefore, the auto-encoded neural networks (DNN-AE, QDNN-AE) are able to map effectively the initial input features in an homogeneous subspace, learned during an unsupervised training process with dedicated encoder-decoder neural networks.

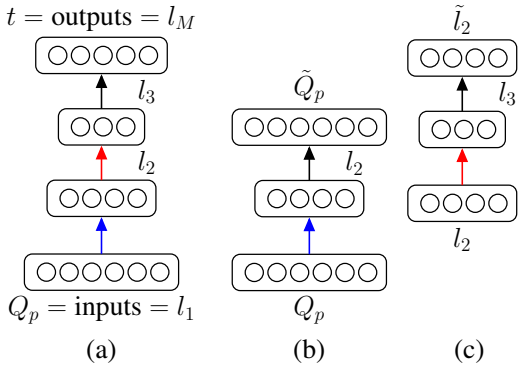


**Fig. 3**. Illustration of a pre-trained Quaternion Deep Neural Network (a) based on 2 dedicated Quaternion encoder-decoders (b-c).

## 3. EXPERIMENTAL PROTOCOL

The efficiency and the effectiveness of the proposed QDNN and QDNN-AE are evaluated during a spoken language un-

derstanding task of theme identification of telephone conversations described in Section 3.1. The conversations data set is from, the DECODA framework detailed in Section 3.2. Section 3.3 expresses the dialogue features employed as inputs of autoencoders as well as the configurations of each neural network.

### 3.1. Spoken Language Understanding task

The application considered in this paper, and depicted in Figure 4, concerns the automatic analysis of telephone conversations [31] between an agent and a customer in the call center of the Paris public transport authority (RATP) [32]. The most important speech analytics for the application are the conversation themes. Relying on the ontology provided by the RATP, we have identified 8 themes related to the main reason of the customer call, such as *time schedules, traffic states, special offers, lost and found*,...

A conversation involves a customer, which is calling from an unconstrained environment (typically from train station or street, by using a mobile phone) and an agent which is supposed to follow a conversation protocol to address customers requests or complains. The conversation tends to vary according to the model of the agent protocol. This paper describes a theme identification method that relies on features related to this underlying structure of agent-customer conversation.

Here, the identification of conversation theme encounters two main problems. First, speech signals may contain very noisy segments that are decoded by an Automatic Speech Recognition (ASR) system. On such difficult environments, ASR systems frequently fail and the theme identification component has to deal with high Word Error Rates (WER $\simeq 49\%$).

Second, themes can be quite ambiguous, many speech acts being theme-independent (and sometimes confusing) due to the specificities of the applicative context: most of conversations evoke traffic details or issues, station names, time schedules, etc... Moreover, some of the dialogues contain secondary topics, augmenting the difficulty of dominant theme identification. On the other hand, dialogues are redundant and driven by the RATP agents which try to follow, as much as possible, standard dialogue schemes.

### 3.2. Spoken dialogue data set

The DECODA corpus [32] contains human-human telephone real-life conversations collected in the Customer Care Service System of the Paris transportation system (RATP). It is composed of $1,242$ telephone conversations, corresponding to about 74 hours of signal, split into a train (train - 739 dialogues), a development (dev - 175 dialogues) and a test set (test - 327 dialogues). Each conversation is annotated with one of the 8 themes. Themes correspond to customer problems or inquiries about itinerary, lost and found, time schedules, transportation cards, state of the traffic, fares, fines and special offers. The LIA-Speeral Automatic Speech Recognition (ASR) system [33] is used for automatically transcribing each conversation. Acoustic model parameters are estimated from 150 hours of telephone speech. The vocabulary contains $5,782$ words. A 3-gram language model (LM) is obtained by adapting a basic LM with the training set transcriptions. Automatic transcriptions are obtained with word error rates (WERs) of $33.8\%$, $45.2\%$ and $49.\%$ on the train, development and test sets respectively. These high rates are mainly due to speech disfluencies in casual users and to adverse acoustic environments in metro stations and streets.

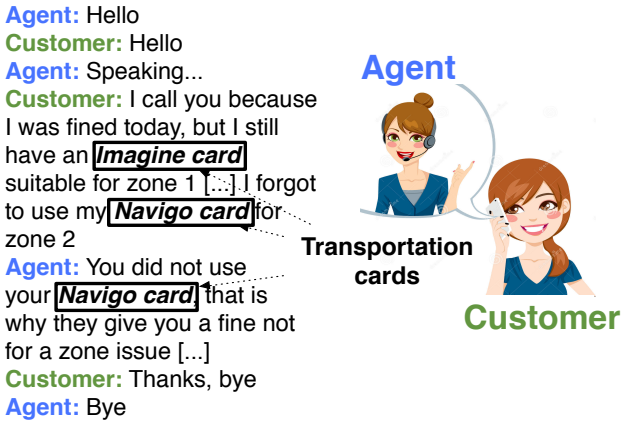


**Fig. 4**. Exemple of a dialogue from the DECODA corpus for the SLU task of theme identification. This dialogue has been labeled by the agent as "OBJECTS" (Lost & founds objects).

### 3.3. Input features and Neural Networks configurations

The experiments compare our proposed QDNN, QDNN-AE with DNN, DNN-AE based on real-numbers and to the QMLP[12], MLP made of a single hidden layer.

**Input features:** [12] show that a LDA [23] space with 25 topics and a specific user-agent document segmentation involving the quaternion $Q = r1 + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$ to be build with the user part of the dialogue in the first complex value $x$, the agent in $y$ and the topic prior of the whole dialogue on $z$, achieve the best results on 10 folds with the QMLP. Therefore, we keep this segmentation and concatenate the 10 representations of size 25 in a single input vector of size $Q_p = 250$. Indeed, the compression of 10 folds in a single input vector gives to the QDNNs more features to generalize patterns. For fair comparison, a QMLP with the same input vector is tested.

**Neural Networks configurations:** First of all, the appropriate size of a single layer for both DNN (MLP) and QDNN (QMLP) have to be investigated by varying the number of neurons $N$ before extending to multiple layers. Different QMLP, MLP have thus been learned by fluctuating the hid-

den layer size from 8 to 1024. Finally we trained multiple DNN and QDNN by varying the number of layers from 1 to 5. Indeed, it is not straightforward to investigate all the possible topologies using 8 to 1024 neurons in 1 to 5 layers. Therefore, each layer contains the same fixed number of neurons. During the experiments, a dropout rate of 50% is used for each layer to prevent overfitting.

## 4. EXPERIMENTAL RESULTS

Section 4.1 details the experiments to find out the "optimal" number of hidden neurons $N_l$ with a real-valued (MLP) and a quaternion-valued (QMLP) neural networks. Then, DNN/QDNN and their pre-trained equivalents DNN-AE/QDNN-AE are compared in Section 4.2. Finally, performances of all neural networks models (real- and quaternion-valued) are depicted in Section 4.3.

### 4.1. QMLP *vs.* MLP

Figure 5 shows the different accuracies obtained on the development and test data sets, with a real-valued and a quaternion based neural networks, composed with a single hidden layer ($M = 3$). To stick with a realistic case, the optimal neurons number in the hidden layer is chosen with respect to the results obtained on the development data set, by varying the number of neurons in the hidden layer. The best accuracies on the development data set for both MLP and QMLP are observed with an hidden layer composed with 512 neurons. Indeed, the QMLP and MLP reach an accuracy of 90.38% and 91.38% respectively. Moreover, both MLP and QMLP performances go down since the hidden layer contains more than 512 neurons.

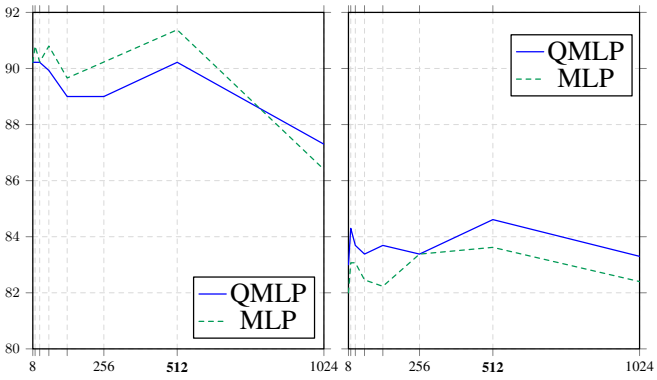


**Fig. 5**. **Accuracies** in % obtained on the development (left) and test (right) data sets by varying the number of neurons in the hidden layer of the **QMLP** and **MLP** respectively.

### 4.2. Quaternion- and real-valued Deep Neural Networks

This Section details the performances obtained for both DNN, QDNN, DNN-AE and QDNN-AE for $1, 2, 3, 4$ and 5 layers composed of 512 neurons.

**DNN *vs.* QDNN randomly initialized.** Table 1 and 2 show the performances obtained for the straightforward real-valued and the proposed quaternion deep neural networks trained without autoencoders and learned with a dropout noise [18] to prevent overfitting.

The "Real Test" accuracies observed for the Test data set are obtained depending on the best accuracy reached on the Development data set.

The "Best Test" accuracy is obtained with the best configuration (number of hidden neurons for the MLP/QMLP or number of hidden layers for the DNN/QDNN) on the Test data set.

| Topology | Dev | Best Test. | Real Test | Epochs |
|---|---|---|---|---|
| **2-Layers** | **91.38** | **84.92** | **84.30** | **609** |
| 3-Layers | 90.80 | 84 | 84 | 649 |
| 4-Layers | 86.76 | 85.23 | 82.39 | 413 |
| 5-Layers | 87.36 | 80.02 | 77.36 | 728 |

**Table 1**. **Summary** of accuracies in % obtained by the **DNN**

It is worth emphasizing that, as depicted in Table 1, the results observed for the DNN on the development and test data sets drastically decrease while the number of layer increases. This is due to the small size of the training data set (739 documents). Indeed, there is not enough patterns for the DNN to construct a high abstract representation of the documents. Conversely, Table 2 shows that the proposed QDNN achieves stable performances with a standard deviation of barely 0.6 on the development set while the DNN gives more than 2.0. Indeed, the DNN accuracies move down from 85% with 2/3/4 hidden layers, to 80% with 5 hidden layers. This can be easily explained by the random initialization of the large number of neural parameters, that makes difficult DNNs to converge to a non-local optimum. Indeed, the Hamilton product of the QDNN constraint the model to learn the latent relations between each component. Therefore the best DNN results are observed with only 2 hidden layers with 91.38% and 84.30%, while the QDNN obtains 92.52% and 85.23% with 4 layers for the development and test data sets respectively. Finally, the QDNN converged about 6 times faster than the DNN with the same topology (148 epochs for the QDNN and 728 for the DNN composed with 5 hidden layers for example).

**DNN *vs.* QDNN initialized with dedicated encoder-decoders.** Table 3 and Table 4 expose the obtained pre-trained QDNN-AE results with dedicated autoencoders (QAE for the QDNN and AE for the DNN). It is worth underlying that the numbers of epochs required to converge for the DNN-

| Topology | Dev | Best Test. | Real Test | Epochs |
|---|---|---|---|---|
| 2-Layers | 91.95 | 86.46 | 84 | 140 |
| 3-Layers | 91.95 | 85.53 | 85.23 | 113 |
| **4-Layers** | **92.52** | **86.46** | **85.23** | **135** |
| 5-Layers | 90.80 | 85.84 | 84 | 148 |

**Table 2**. **Summary** of accuracies in % obtained by the **QDNN**

AE is lower than those for the DNN for all the topologies, as depicted in Table 1. Moreover, the accuracies reported for the DNN-AE are more stable and move up with regard to the number of layers.

| Topology | Dev | Best Test. | Real Test | Epochs |
|---|---|---|---|---|
| 2-Layers | 90.23 | 84 | 82.46 | 326 |
| 3-Layers | 90.80 | 84.92 | 83.69 | 415 |
| 4-Layers | 91.52 | 85.23 | 84.64 | 364 |
| **5-Layers** | **91.95** | **85.23** | **84.30** | **411** |

**Table 3**. **Summary** of accuracies in % obtained by the **DNN-AE**

The same phenomenon is observed with the QDNN-AE, but with a smaller gain for the number of epochs alongside with better reported accuracies. Indeed, the DNN-AE gives an accuracy of $84.30\%$ on the test data set while the QDNN-AE obtains an accuracy of $86.46\%$ in real conditions with a gain of $2.16$ points.

| Topology | Dev | Best Test. | Real Test | Epochs |
|---|---|---|---|---|
| 2-Layers | 92.52 | 86.46 | 84.61 | 100 |
| 3-Layers | 93.57 | 86.23 | 85.83 | 95 |
| 4-Layers | 92.52 | 86.46 | 86.46 | 88 |
| **5-Layers** | **93.57** | **86.76** | **86.46** | **132** |

**Table 4**. **Summary** of accuracies in % obtained by the **QDNN-AE**

Overall, the pre-training process allows each model (DNN-AE/QDNN-AE) to better perform on a theme identification task of telephone conversations. Indeed, both DNN-AE and QDNN-AE need less epochs (and thus less processing time) and reach better accuracies, due to their pre-training process based on dedicated encoder-decoders to converge quickly to an optimal configuration (weight matrices $w$) during the fine tuning phase.

### 4.3. QDNN-AE *vs.* other neural networks

Table 5 sums up the results obtained on the theme identification task of telephone conversations from the DECODA corpus, with different real-valued and quaternion neural networks. The first remark is that the proposed QDNN-AE obtains the best accuracy ($86.46\%$) for both development and test data sets compared to the real-valued neural networks (Deep stacked autoencoder DSAE ($82\%$), MLP ($83.38\%$), DNN ($84\%$) and DNN-AE ($84.3\%$)). Moreover, the QDNN randomly initialized outperforms also all real-valued neural networks with an accuracy of $85.23\%$. We can point out that each quaternion-based neural networks performs better than his real-valued equivalent thanks to the Hamilton product ($+2.61\%$ for the QMLP for example). Finally, the QDNN presents a gain of roughly $1.25\%$ compared to the real-valued DNN, and the pre-trained QDNN-AE shows an improvement of $2.10\%$ compared to the DNN-AE.

| Models | Type | Dev. | Real Test | Epochs | Impr. |
|---|---|---|---|---|---|
| DSAE[34] | $\mathbb{R}$ | 88.0 | 82.0 | - | - |
| MLP | $\mathbb{R}$ | 91.38 | 83.38 | 499 | +1.38 |
| QMLP | $\mathbb{Q}$ | 90.38 | 84.61 | 381 | +2.61 |
| DNN | $\mathbb{R}$ | 91.38 | 84 | 609 | - |
| QDNN | $\mathbb{Q}$ | 92.52 | 85.23 | 135 | +1.23 |
| DNN-AE | $\mathbb{R}$ | 91.95 | 84.30 | 411 | - |
| **QDNN-AE** | $\mathbb{Q}$ | **93.57** | **86.46** | **132** | **+2.16** |

**Table 5**. **Summary** of accuracies in % obtained by different neural networks on the DECODA famework.

## 5. CONCLUSION

**Summary.** This paper proposes a promising deep neural network framework, based on the quaternion algebra, coupled with a well-adapted pre-training process made of quaternion encoder-decoders. The initial intuition that the QDNN-AE better captures latent abstract relations between input features, and can generalize from small corpus due to the high dimensionality added by multiple layers, has been demonstrated. It has been shown that a well-suited pre-training process alongside to an increased number of neural parameters, allow the QDNN-AE to outperform all the previously investigated models on the DECODA SLU task. Moreover, this paper shows that quaternion-valued neural networks always perform better and faster than real-valued ones, achieving impressive accuracies on the small DECODA corpus with a small number of input features and, therefore, few neural parameters.

**Limitations and Future Work.** The document segmentation process is a crucial issue when it comes to better capture latent, temporal and spacial informations, and thus needs more investigation to expose the potential of quaternion-based models. Moreover, such DNN algorithms are adapted from real-based ones and do not take into account the entire set of specificities of the quaternion algebra. Therefore, a future work will consist in investigate different structures of neural networks such as recurrent and convolutional, and propose well-tailored learning algorithms adapted to hyper-complex numbers (rotations).

## 6. REFERENCES


[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[2] Dan Ciregan, Ueli Meier, and Jürgen Schmidhuber, "Multi-column deep neural networks for image classification," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 3642–3649.

[3] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al., "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.

[4] George E Dahl, Dong Yu, Li Deng, and Alex Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 30–42, 2012.

[5] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton, "Speech recognition with deep recurrent neural networks," in *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*. IEEE, 2013, pp. 6645–6649.

[6] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur, "Recurrent neural network based language model.," in *Interspeech*, 2010, vol. 2, p. 3.

[7] Ken-ichi Funahashi and Yuichi Nakamura, "Approximation of dynamical systems by continuous time recurrent neural networks," *Neural networks*, vol. 6, no. 6, pp. 801–806, 1993.

[8] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins, "Learning to forget: Continual prediction with lstm," 1999.

[9] Alex Graves and Jürgen Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural Networks*, vol. 18, no. 5, pp. 602–610, 2005.

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[11] Paolo Arena, Luigi Fortuna, Giovanni Muscato, and Maria Gabriella Xibilia, "Multilayer perceptrons to approximate quaternion valued functions," *Neural Networks*, vol. 10, no. 2, pp. 335–342, 1997.

[12] Titouan Parcollet, Mohamed Morchid, Pierre-Michel Bousquet, Richard Dufour, Georges Linarès, and Renato De Mori, "Quaternion neural networks for spoken language understanding," in *Spoken Language Technology Workshop (SLT), 2016 IEEE*. IEEE, 2016, pp. 362–368.

[13] Mohamed Morchid, Georges Linarès, Marc El-Beze, and Renato De Mori, "Theme identification in telephone service conversations using quaternions of speech features," in *Interspeech*. ISCA, 2013.

[14] Teijiro Isokawa, Tomoaki Kusakabe, Nobuyuki Matsui, and Ferdinand Peper, "Quaternion neural network and its application," in *Knowledge-based intelligent information and engineering systems*. Springer, 2003, pp. 318–324.

[15] William Rowan Hamilton, *Elements of quaternions*, Longmans, Green, & Company, 1866.

[16] Ronan Collobert and Jason Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 160–167.

[17] Xavier Glorot and Yoshua Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *International conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[18] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[19] Yoshua Bengio, "Learning deep architectures for ai," *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[20] Ruslan Salakhutdinov and Geoffrey Hinton, "Deep boltzmann machines," in *Artificial Intelligence and Statistics*, 2009, pp. 448–455.

[21] Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle, et al., "Greedy layer-wise training of deep networks," *Advances in neural information processing systems*, vol. 19, pp. 153, 2007.

[22] Toshifumi Minemoto, Teijiro Isokawa, Haruhiko Nishimura, and Nobuyuki Matsui, "Feed forward neural network with random quaternionic neurons," *Signal Processing*, vol. 136, pp. 59–68, 2017.

[23] David M Blei, Andrew Y Ng, and Michael I Jordan, "Latent dirichlet allocation," *the Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003.

[24] I.L. Kantor, A.S. Solodovnikov, and A. Shenitzer, *Hypercomplex numbers: an elementary introduction to algebras*, Springer-Verlag, 1989.

[25] Jack B Kuipers, *Quaternions and rotation sequences*, Princeton university press Princeton, NJ, USA:, 1999.

[26] Fuzhen Zhang, "Quaternions and matrices of quaternions," *Linear algebra and its applications*, vol. 251, pp. 21–57, 1997.

[27] J.P. Ward, *Quaternions and Cayley numbers: Algebra and applications*, vol. 403, Springer, 1997.

[28] Diederik Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[29] Matthew D Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[30] Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio, "Why does unsupervised pre-training help deep learning?," *Journal of Machine Learning Research*, vol. 11, no. Feb, pp. 625–660, 2010.

[31] John J Godfrey, Edward C Holliman, and Jane McDaniel, "Switchboard: Telephone speech corpus for research and development," in *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*. IEEE, 1992, vol. 1, pp. 517–520.

[32] Frederic Bechet, Benjamin Maza, Nicolas Bigouroux, Thierry Bazillon, Marc El-Beze, Renato De Mori, and Eric Arbillot, "Decoda: a call-centre human-human spoken conversation corpus.," in *LREC*, 2012, pp. 1343–1347.

[33] Georges Linares, Pascal Nocéra, Dominique Massonie, and Driss Matrouf, "The lia speech recognition system: from 10xrt to 1xrt," in *Text, Speech and Dialogue*. Springer, 2007, pp. 302–308.

[34] Killian Janod, Mohamed Morchid, Richard Dufour, Georges Linares, and Renato De Mori, "Deep stacked autoencoders for spoken language understanding," *ISCA INTERSPEECH*, vol. 1, pp. 2, 2016.