SER ⇒ Sp. Emo - Reg.

Learning Speech Emotion Representations in the Quaternion Domain

Eric Guizzo, Tillman Weyde, Simone Scardapane, and Danilo Comminiello *Senior Member, IEEE*

***Abstract*—The modeling of human emotion expression in speech signals is an important, yet challenging task. The high resource demand of speech emotion recognition models, combined with the the general scarcity of emotion-labelled data are obstacles to the development and application of effective solutions in this field. In this paper, we present an approach to jointly circumvent these difficulties. Our method, named RH-emo, is a novel semi-supervised architecture aimed at extracting quaternion embeddings from real-valued monoaural spectrograms, enabling the use of quaternion-valued networks for speech emotion recognition tasks. RH-emo is a hybrid real/quaternion autoencoder network that consists of a real-valued encoder in parallel to a real-valued emotion classifier and a quaternion-valued decoder. On the one hand, the classifier permits to optimize each latent axis of the embeddings for the classification of a specific emotion-related characteristic: valence, arousal, dominance and overall emotion. On the other hand, the quaternion reconstruction enables the latent dimension to develop intra-channel correlations that are required for an effective representation as a quaternion entity. We test our approach on speech emotion recognition tasks using four popular datasets: Iemocap, Ravdess, EmoDb and Tess, comparing the performance of three well-established real-valued CNN architectures (AlexNet, ResNet-50, VGG) and their quaternion-valued equivalent fed with the embeddings created with RH-emo. We obtain a consistent improvement in the test accuracy for all datasets, while drastically reducing the resources' demand of models. Moreover, we performed additional experiments and ablation studies that confirm the effectiveness of our approach. The RH-emo repository is available at: https://github.com/ispamm/rhemo.**

***Index Terms*—Speech Emotion Recognition, Quaternion Neural Networks, Quaternion Algebra, Transferable Embeddings**

I. Introduction

HUMAN-MACHINE interaction is becoming increasingly important in our everyday life. Research on speech recognition reached near-human performance in the recent years. Nevertheless, besides the mere sequence of words there is additional information that the speech can carry, in particular about emotion. Speech Emotion Recognition (SER) is therefore acquiring a growing role in research on human-machine interaction, since it helps provide a more complete account of the information conveyed by speech signals. Despite the impressive success that neural networks have achieved in this task, SER is still challenging due to the variablilty of emotional expression, especially in real world scenarios where generalization to unseen speakers and contexts is required [1], [2]. The difficulty of this task is partly related to the general scarcity of emotionally-labelled audio data, which is due to the high cost of the recording and labelling such data. Another well-known difficulty of SER is that emotional information in speech involves long-term temporal dependencies that are in the order of seconds [3]–[5]. This forces models to analyze large temporal windows and, consequently, to use a large amount of resources.

This study proposes a joint solution for two main issues in SER research. Broadly speaking, we propose to map speech signals into a compact multi-channel latent representation that permits having different "emotional viewpoints" of the signal, that are signal representations individually related to different components of human emotion, namely: valence arousal and dominance. To this end, we make use of quaternion information processing, which is a well-established strategy to minimize models' resources demand without reducing their performance, as we discuss in detail in Sections II and III. The resulting proposed model, named Real to Emotional H-Space (RH-emo), is a semi-supervised autoencoder architecture that maps input speech signals to an embedded emotional-quaternion space. The axes of the embedded space are individually related to different emotion characteristics, i.e., valence, arousal and dominance, which are represented as quaternion components. As we will be further explored from Section V onward, when used as a feature extractor that feeds into quaternion neural networks (QNNs), RH-emo improves the performance in SER tasks while considearbly reducing the number of trainable parameters and computing resources, compared to equivalent real-valued models processing plain spectrograms. This behavior is consistent also in situations where data is very scarce.

The specific contributions of our work are the following:

- We define a novel method, RH-emo, that draws quaternion-valued embeddings from speech signals, where each quaternion component is tailored on a specific emotional characteristic.
- We leverage the capabilities of quaternion emotion embeddings and the effectiveness of quaternion convolutional neural networks (QCNNs) to jointly solve two of the most significant issues related to speech emotion recognition: data scarcity and high resources demand of models.
- We extensively evaluate our approach using 4 popular SER datasets and 3 widely-used CNN-based architec-

E. Guizzo and T. Weyde are with the Department of Computer Science, City, University of London, Northampton Square, London EC1V 0HB, United Kingdom. S. Scardapane and D. Comminiello are with the Department of Information Engineering, Electronics and Telecommunications (DIET), Sapienza University of Rome, Via Eudossiana 18, 00184 Rome, Italy. Corresponding author's email: ericguizzo@city.ac.uk.

This work has been performed while the first author was a PhD Visiting Student at Sapienza University of Rome, Italy.

This work has been partly supported by "Progetti di Ricerca Grandi" of Sapienza University of Rome under grant number RG11916B88E1942F.

tures.

- We provide open-source code[1] and pretrained models[2] that can be exploited to improve the performance and efficiency of existing SER models.

The reminder of the paper is organized as follows: Section II reviews the relevant literature, Section III is a brief overview of quaternion neural networks, Section IV describes our proposed method in detail, Section V presents our experimental setup and results, Section VI presents the ablation studies we conduct, Section VII discusses our outcomes and the properties of our approach and Section VIII draws the conclusions of this paper.

II. Background

In the literature, two main approaches to label expressed human emotions exist. On the one hand, discrete models provide a set of fixed emotion categories, such as *happy, sad, angry, fearful, surprised, disgusted, neutral*. On the other hand, continuous models map emotions into a multidimensional space. The most common model is a 2D *valence-arousal* space, where valence describes the degree of emotional pleasantness and arousal (or activation) the intensity of the emotion. Dominance can be added as a third dimension describing the amount of control of a person expressing an emotion. Discrete emotions can be mapped in this continuous space although the exact mapping is not standardized and different studies can use slightly different mappings.

A traditional approach to SER is based on two consecutive stages: hard-coded extraction of affect-salient features followed by a learning-based classification or regression. Various combinations of features and classifier types have been proposed. The most commonly used features are: base pitch, formant features, energy/spectral features, and prosody. A wide variety of classifiers has been proposed: artificial neural networks [6]–[8], Bayesian networks, [9], Hidden Markov Models [10], [11], support vector machines [12], [13], and Gaussian mixture models [14]. Nevertheless, in state-of-the-art methods there is no default choice of features and classifier type [15]. With the advent of deep learning, end-to-end learning mostly replaced hard-coded feature extraction and selection, with models automatically extracting features from low-level representations of the input data (usually Fourier-based transforms, wavelet transforms or raw audio data). This enables a model to fine-tune the feature extraction for a specific task and, consequently, often obtain a higher accuracy compared to engineered feature extraction. A range of deep learning architectures have been adopted for SER. The most commonly used are convolutional neural networks [16]–[18], recurrent neural networks [19], [20] and combinations of the two [21]–[23] Various studies directly compare the performance of approaches using end-to-end learning and hard-coded feature extraction, showing that the former generally provides a higher classification accuracy on the same data [24]–[27]. Nevertheless, as a drawback, deep learning models generally requires a higher computational cost and longer training times than traditional machine learning techniques and the end-to-end learning usually requires a large amount of labelled data [28], [29].

A well-established solution to overcome the data scarcity in SER is transfer learning by weight initialization: network weights are initialized with values from a network that was pretrained with a different task, possibly on a different (usually large) dataset. Many variants of this method have been shown to improve the performance of SER models in limited-data scenarios and even when the task is rather distant from speech emotion [30]–[32]. Also, various data augmentation strategies have been successfully adopted for the same purpose, e.g. [33], [34]. On the other hand, the application of dimensionality reduction transformations to the model's input data are established strategies for reducing the resource demands while limiting the loss of useful information carried by the input data. Among others, autoencoders, PCA-based approaches and transformer networks have been used in the field of SER [31], [35], [36], obtaining improvement both in the model's efficiency and classification accuracy.

A recent and increasingly popular strategy to improve the efficiency and the performance of deep learning models is the use of quaternion information processing [37]–[43]. Performing operations in the quaternion domain permits to bootstrap intra-channel correlations in multidimensional signals [44], [45], i.e., among the color channels of RGB-encoded images. Moreover, due to the fewer degrees of freedom of the Hamilton product compared to the regular dot product, quaternion networks have a significantly lower number of parameters compared to the real counterparts [37]. Quaternion-valued neural networks have also been successfully adopted in the audio domain [46] and specifically for speech recognition [42] and speech emotion recognition [43]. Nevertheless, an intrinsic limitation of quaternion information processing is that it requires three or four-dimensional data as input, where intra-channel correlations exist [38]–[41]. This is necessary to enable the benefits derived from the use of the Hamilton product instead of the regular dot product, as further discussed in Section III. In the audio domain, first order Ambisonics [47] signals are naturally suited for a quaternion representation, being four-dimensional and presenting strong correlations among the spatial channels, and the application of quaternion networks to problems related to this audio format has already been extensively investigated [48], [49]. Nevertheless, in the vast majority of cases, audio-related machine-learning tasks deal with monaural signals, which are usually treated as vectors of scalars (time-domain signals), matrices of scalars (magnitude spectrograms) or 3D tensors (complex spectrograms). Hence they can not be naturally represented as a quaternion entity and additional processing is required to produce a suitable quaternion representation of these signals.

A number of different approaches have been proposed to overcome the necessity of having three or four-dimensional input data with intra-channel correlations. Among others, [42] use mel spectrograms, cepstral coefficients and first and second order derivatives as the four axes of the encoded quaternion. In contrast, [43] convert mel spectrograms to color-scaled images and use the RGB channels as axes of the encoded quaternion,

[1] https://github.com/ispamm/rhemo.

[2] Pretrained models: rhemo/weights.

Complex

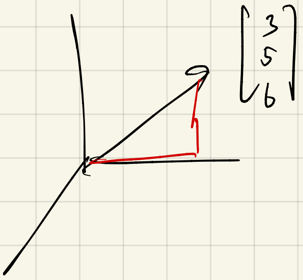
$z = \cancel{x}a + bi$

$i^2 = -1$

$Q = a + bi + cj + dk$

$i^2 = j^2 = k^2 = ijk = -1$

$\}$ R



$z_1 \cdot z_2 = (a_1 + b_1 i)\cdot(a_2 + b_2 i)$

$= (a_1 a_2 - b_1 b_2) + ($

$\hat{\imath}^2 = \hat{\jmath}^2 = \hat{k}^2 = ijk = -1$

following a computer vision-oriented approach. Parcollet *et al.* [50] presented two learning-based approaches to map real-valued vectors into the quaternion domain, by producing through a network four-channel representations of the input data that present meaningful intra-channel correlations. On the one hand, the Real to H-space encoder [50], applied to speech recognition tasks, consists of a simple real-valued dense layer applied at the beginning of a quaternion classifier network, which is trained jointly with the classifier. On the other hand, the Real to H-space Autoencoder, tested in the natural language processing field (conversation theme identification) [50] operates in an an unsupervised way. Such method contains a real-valued encoder and a quaternion-valued decoder, where the latter is expected to enable both the network's embeddings and output to present meaningful intra-channel correlations that can be exploited by a quaternion-valued classifier network.

In this paper, we introduce RH-emo, a hybrid real-quaternion autoencoder-classifier architecture that is trained in a semi-supervised fashion in order to optimize each axis of the embedding dimension to different emotional characteristics: the first channel is optimized for discrete emotion recognition and the 3 other channels are individually optimized for the classification of the valence, arousal and dominance (as shown in Figure 1). RH-emo is intended to be used as a feature extractor that permits using QNNs for SER tasks with real-valued signals without additional preprocessing. This approach has two advantages: it improves the performance of SER models even in situations where data is scarce and it drastically reduces the number of network parameters, consequently reducing the resource demand. We extend the approach of the quaternion autoencoder in [50] by specializing the learnt quaternion representation for our specific task (SER), where the different axes are optimized for detection of different emotional characteristics that are coherent with the most used criteria of emotion classification. Moreover, we implement it with a more complex architecture (deep convolutional autoencoder) and we apply it to a different domain: emotion recognition from speech audio.

ijk

III. Quaternion Convolutional Neural Networks

Operations between quaternion numbers are defined in the quaternions algebra $\mathbb{H}$. A quaternion Q is a four-dimensional extension of a complex number, defined as $\mathbf{q} = q_0 + q_1\hat{\imath} + q_2\hat{\jmath} + q_3\hat{\kappa} = q_0 + q$, where, q_0, q_1, q_2 are real numbers, and $\hat{\imath}$, $\hat{\jmath}$ and $\hat{\kappa}$ are the quaternion unit basis. In this representation q_0 is the real part and $q_1\hat{\imath} + q_2\hat{\jmath} + q_3\hat{\kappa}$ is the imaginary part, where $\hat{\imath}^2 = \hat{\jmath}^2 = \hat{\kappa}^2 = -1$ and $\hat{\imath}\hat{\jmath} = -\hat{\jmath}\hat{\imath}$. From the latter assumption follows that the quaternion vector multiplication is not commutative. A quaternion can also be represented as a matrix of real numbers as:

$x \quad q \quad x \quad x$

$$\mathbf{q} = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix} \quad x \qquad (1)$$

Analogously to real and complex numbers, a set of operations can be defined in the quaternion space:

- **Addition**: $\mathbf{q} + \mathbf{p} = (q_0 + p_0) + (q_1 + p_1)\hat{\imath} + (q_2 + p_2)\hat{\jmath} + (q_3 + p_3)\hat{\kappa}$
- **Conjugation**: $\mathbf{q}^* = q_0 - q_1\hat{\imath} - q_2\hat{\jmath} - q_3\hat{\kappa}$
- **Scalar multiplication**: $\lambda\mathbf{q} = \lambda q_0 + \lambda q_1\hat{\imath} + \lambda q_2\hat{\jmath} + \lambda q_3\hat{\kappa}$
- **Element multiplication** (or **Hamilton product**):

$$\begin{aligned} \mathbf{q} \otimes \mathbf{p} &= (q_0 + q_1\hat{\imath} + q_2\hat{\jmath} + q_3\hat{\kappa})\,(p_0 + p_1\hat{\imath} + p_2\hat{\jmath} + p_3\hat{\kappa}) \\ &= (q_0p_0 - q_1p_1 - q_2p_2 - q_3p_3) \\ &+ (q_0p_1 + q_1p_0 + q_2p_3 - q_3p_2)\,\hat{\imath} \\ &+ (q_0p_2 - q_1p_3 + q_2p_0 + q_3p_1)\,\hat{\jmath} \\ &+ (q_0p_3 + q_1p_2 - q_2p_1 + q_3p_0)\,\hat{\kappa}. \end{aligned} \qquad (2)$$

The quaternion convolutional neural network (QCNN) is an extension of the real-valued convolutional neural network to the quaternion domain. For each input vector of a quaternion layer the dimensions are split into four parts to compose a quaternion representation. In a quaternion-valued fully-connected layer, the parameters matrices are treated as a single quaternion entity with four components, although they are manipulated as matrices of real numbers [51]. In a quaternion layer, the dot product operations used in real layers are replaced with the Hamilton product (Equation 2) between the input vector and a quaternion-represented weight matrix. This allows to process all input channels together as a single entity maintaining original intra-channels dependencies, because the weights submatrices are shared among the input channels. Consequently, quaternion layers permit to spare the 75% of free parameters compared to their real-valued equivalents because, as shown in Equation 2, the same components are re-used to build the output matrix.

In a QCNN, the convolution of a quaternion filter matrix with a quaternion vector is performed as the Hamilton product between the real-valued matrices representation of the input vector and filters. A quaternion convolution between a quaternion input vector $\mathbf{x} = x_0 + x_1\hat{\imath} + x_2\hat{\jmath} + x_3\hat{\kappa}$ and a quaternion filter $W = W_0 + W_1\hat{\imath} + W_2\hat{\jmath} + W_3\hat{\kappa}$ can be defined as:

$W x$

$$W * x = \begin{bmatrix} W_0 & -W_1 & -W_2 & -W_3 \\ W_1 & W_0 & -W_3 & W_2 \\ W_2 & W_3 & W_0 & -W_1 \\ W_3 & -W_2 & W_1 & W_0 \end{bmatrix} * \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} y'_0 \\ y'_1\hat{\imath} \\ y'_2\hat{\jmath} \\ y'_3\hat{\kappa}' \end{bmatrix} \qquad (3)$$

The optimization of quaternion-valued networks is identical to the one of a real network and can be achieved through regular backpropagation. This is possible because of the use of split activation and loss functions, as introduced in [50], [52]. These functions map a quaternion-like entity back to the real domain, consequently enabling the use of standard loss functions for the network training.

IV. The Proposed RH-emo Model

A. Approach

The main aim of RH-emo is to map real-valued spectrograms to the quaternion domain, building compact emotion-related quaternion embeddings where each axis is optimized for a different emotional characteristic. In the embedded dimension, the real axis of the quaternion is optimized for

สร้างต้น rep ⇒ ใช้ AE ในการ

the discrete classification of 4 emotions: *neutrality, anger, happiness, sadness* and the 3 complex axes are optimized for the prediction of emotion in a *valence, arousal* and *dominance* 3D space. This representation exploits the natural predisposition of quaternion algebra to process data where a 4 or 3-channels representation is meaningful. Nevertheless, in most machine learning applications of quaternion algebra, the input data is naturally organized with a meaningful shape, as happens for instance with RGB/RGBA images (where the color/alpha channels are treated as different quaternion axes) and 1st order Ambisonics audio signals (where the 4 spatial channels are considered as the quaternion axes). In our case, instead, such quaternion representation is created through a semi-supervised learning procedure, where the different axes are forced to contain information related to different complementary emotion characteristics. Therefore, in a certain sense, the axes of this embedded dimension can be thought as different "emotional point of views" of an audio signal.

RH-emo is intended to be used as a pretrained feature extractor to enable the use of quaternion-valued neural networks for SER tasks applied to monoaural audio signals. On the one hand, the emotion-related disentanglement among channels helps to enhance the performance of SER models, especially under conditions of data scarcity. Whereas, on the other hand, the reduced dimensionality together with the enabled possibility to classify the data with quaternion-valued networks permit to spare a large amount of network parameters, consequently lowering the resources demand and speeding up the training.

B. *RH-emo Architecture*

RH-emo is a hybrid real/quaternion autoencoder network. Its structure is similar to R2Hae [50], nevertheless RH-emo is based on a convolutional design and it embraces multiple classification branches, as opposed to R2Hae. We used a public PyTorch implementation of convolution layers and operators[3]. As Figure 1 shows, our RH-emo is composed of three components: an encoder $E(X)$ acting on the (real-valued) input spectrogram, producing an embedded vector. The output of the encoder is then fed separately to a (quaternion-valued) decoder $D(Z)$ to reconstruct the original spectrogram, and to a classification head $C(Z)$ for performing emotion recognition. The classifier outputs four separate predictions y_D, y_v, y_a and y_d which are, respectively, a discrete and a continuous (in the valence, arousal, dominance space) categorization of the emotional content of the spectrogram. The specific architecture for each of these blocks, as well as the loss function we optimize and the training strategy we adopt are described more in detail in the following paragraphs.

1) Encoder: The input data, a magnitudes-only real-valued spectrogram in our case, is forward propagated through a real-valued autoencoder made up of 3 convolution blocks. Each block contains a 2D convolution layer (ReLU activations, 3x3 kernels, single-pixel stride, increasing channels number: 1, 2, 4), followed by max-pooling layers of dimension [2x2], [2x1], [2x1]. Moreover, only between the first and the second block, a batch normalization layer is present. The encoder produces an embedded vector that presents a dimensionality reduced by a factor of 0.25 compared to the input. In our experiments we use input spectrograms with a shape of 1x512x128 (channels, time-steps, frequency-bins) and the embedded dimension created by the encoder has a shape of 4x64x64. The embedded vector is then forward propagated in parallel into four distinct real-valued classifiers and also into a quaternion-valued decoder. It is therefore important that the embedded vector contains a number of elements that is multiple of four, in order to be properly treated as a quaternion by the decoder section of the network.

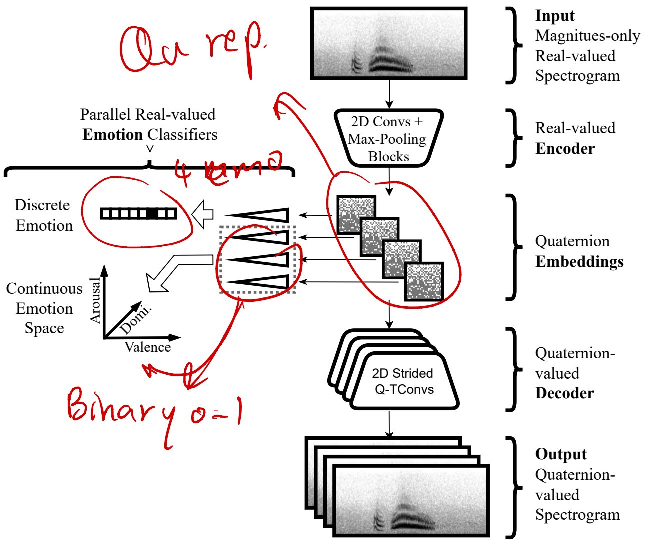


Fig. 1. RH-emo Block Diagram. An input magnitudes-only spectrogram is first propagated into a real-valued convolutional encoder that generates embeddings with a [4x64x64] shape. The network is then split in two branches: a completely unsupervised quaternion-valued decoder that reconstructs the input spectrogram projecting it in a four-channel quaternion space and a set of 4 parallel real-valued supervised classifiers, each connected to one of the four channels of the embeddings and separately classifying different emotion characteristics: discrete emotion, valence, arousal, and dominance.

2) Classifiers: Each classifier consists of a sequence of 3 real-valued fully connected layers, where the first 2 contain 4096 neurons and are followed by a dropout layer. In the first classifier, the output layer contains 4 output neurons (the number of emotional classes to be classified) and softmax activation. Instead the other 3 classifiers are identical and have one single output neuron with sigmoid activation, as they are individually aimed at a binary classification task: the prediction of "high" or "low" valence, arousal, and dominance, respectively.

3) Decoder: The decoder mirrors the encoder's structure, but using quaternion-valued 2D transposed convolutions with a stride that mirrors the pooling dimensions of the encoder, instead of the sequence of 2D real-valued convolutions and 2x2 max-pooling and a quaternion-valued batch normalization layer instead of its real-valued counterpart. The output of the decoder is therefore a matrix with the same dimensions as the input, but with 4 channels instead of a single one.

[3] https://github.com/Orkis-Research/Pytorch-Quaternion-Neural-Networks

AE - arch

TABLE I
PRETRAINING RESULTS ON IEMOCAP

Arch.	Method	Params	Train acc.	Test acc.
RH-emo	/	1.3×10^8	80.34	60.7
VGG16	Real	1.6×10^8	74.88	62.87
	RH-emo+Quat	1×10^7	72.25	71.10
AlexNet	Real	5.7×10^7	71.02	63.33
	RH-emo+Quat	1×10^7	71.81	70.31
ResNet-50	Real	2.3×10^7	61.05	57.20
	RH-emo+Quat	4.9×10^6	73.03	**71.20**

C. Loss Function

The loss function we minimize during the training of RH-emo is a weighted sum of the binary crossentropy reconstruction loss between the input spectrogram and the decoder's output, the categorical crossentropy classification loss of the emotion labels predicted by the supervised classifier in the middle of the network (discrete, valence and arousal).

The objective function we minimize is therefore:

$$\mathcal{L} = \overset{\text{AE}}{\text{BCE}(X, Y_r)} + \boxed{\beta} \cdot \{\overset{\text{dis}}{\text{CE}(p,t)} + \boxed{\alpha} \cdot [\underbrace{\text{BCE}(v_p, v_t)}_{v} + \underbrace{\text{BCE}(a_p, a_t)}_{a} + \overset{\text{domain}}{\underline{\text{BCE}(d_p, d_t)}}]\} \tag{4}$$

where BCE is the binary crossentropy loss, CE is the categorical crossentropy loss, β and α are scalar weight factors, X is the input spectrogram, Y_r is the decoder's output remapped to the real domain through the split activation function (as discussed below), p and t are respectively the discrete emotion prediction and truth label, v_p/v_t, a_p/a_t and d_p/d_t are respectively the valence, arousal and dominance prediction and truth labels.

For the reconstruction loss computation it is necessary to map the quaternion-valued decoder output back to the real domain, in order to have the same shape as the input vector. For this purpose we use a stratagem similar to the "split activation" described in [50], [52]: we perform an element-wise mean across the channel dimension of the quaternion output, bringing back the 4-channels vector to a single-channel shape. During the training, this forces the model to not weight the intra-channel correlations among the quaternion axes in the reconstruction term of the loss (the leftmost term of Equation 4). Our expectation is that this leaves room to the emotion recognition term of the loss (the rightmost term of Equation 4) for tuning these correlations, making them related to the emotional information.

D. Training strategy

For the RH-emo training we use the Interactive Emotional Dyadic Motion Capture Database (Iemocap) dataset [53], which includes: 5 speakers, 7529 utterances, 9:32 hours of audio, 10 emotion labels and it is in English language. We selected this specific dataset for the following reasons: it is one of the most popular SER datasets, it contains a large amount of datapoints, it is not limited to a restricted set of sentences, emotions are expressed by actors with a natural feeling rather than being over-emphasised (for details, please refer to the original paper), and it is labelled both in the discrete and continuous (valence, arousal, dominance) emotional domains.

We apply 4 preprocessing stages to the raw data: we first extract 4-seconds non-overlapped fragments (or zero-pad if a datapoint is shorter that this duration). Then, we compute the short-time Fourier transform (STFT) using 16 ms sliding windows with 50% overlap, applying a Hamming window and discarding the phase information. After this point, we normalize the whole dataset between 0 and 1 and, in the end, we zero-pad the spectrograms to match a shape of 512 (time-steps) x 128 (frequency-bins).

To permit proper convergence, we perform the training in 2 consecutive stages: we first train the network until convergence with the β weight set to 0. This removes the rightmost term from equation 4, consequently eliminating the emotion classification part of the loss. Doing so, we train the network in a completely unsupervised way only to perform a quaternion projection of the real input spectrogram, without taking into account any emotion-related information. After this stage, we re-train the network adding also the classification term in the loss in order to specialize the learnt representations to the emotion recognition task, but also maintaining the embedded vector in a quaternion-compatible shape that is meaningful for the decoder part of the network. For this stage we performed a grid search to find the best combination of the emotion classification weights β and α and we ended up using $\beta = 0.01$ and $\alpha = 100$. This means that overall we weight more the reconstruction error in the loss function (thanks to the low β), and we weight more the dimensional emotion classification compared to the discrete classification (thanks to the high α).

While for the first, completely unsupervised, training stage we use all data available with Iemocap, in the second supervised stage we use only a subset of the dataset, including only the datapoints related to 4 emotions (*angry, happy, neural, sad)*. This is a standard procedure with Iemocap, as the other labels contained in the dataset are highly unbalanced. For both training stages we use subsets of approximately 70% of the data for training, 20% for validation and 10% for the test set. We use a learning rate of 0.001 in the first stage and of 0.000001 in the second one, a batch size of 20 and the Adam optimizer [54]. We use dropout at 50% in the classification branches for the second training stage. We apply early stopping by testing at the validation loss improvement with a patience of 100 epochs in the first stage and of 30 epochs for the second one.

After these 2 training stages we obtain a test reconstruction loss (the isolated leftmost term of Equation 4) of 0.00413 and competitive test classification accuracy: 60.7% for the discrete classification and respectively 65.4%, 75.3% and 70.2% for the valence, arousal and dominance dimensions.

V. EVALUATION

In order to test the capabilities and properties of RH-emo, we compare the classification accuracy for SER tasks obtained with real-valued CNN networks and equivalent quaternion-valued versions of them (QCNNs). For the quaternion versions we keep the same architecture of the real CNNs, but we use quaternion-valued convolution and quaternion-valued fully connected layers instead of the canonical real-valued ones, with the exception of the final layer of the networks, which are real valued also in the QCNNs. For the real networks we use the magnitudes-only spectra as input, while for the quaternion networks we use the embeddings generated with RH-emo pretrained on Iemocap. Moreover, we compare and combine our approach with a standard transfer learning method performed on the same dataset (Iemocap): pretraining with weight initialization. Therefore we have two distinct types of pretraining: the pretraining of the RH-emo network, that we use to compute the emotional embeddings, and the pretraining of the CNNs, that we use to perform the actual SER task. Both pretrainings are performed on the Iemocap dataset. To avoid confusion, from here on we will refer to the first as RH-emo pretraining and to the latter as CNNs pretraining.

Figure 2 depicts all cases we include in our experimental setup. The color coding of Figure 2 shows the 3 consecutive stages of our experiments: first we pretrain RH-emo (yellow), then we pretrain the CNNs (orange) on Iemocap and finally we train or retrain the CNNs on other datasets. We have two types of baseline: the first one, shown in the upper row of Figure 2, is a standard real-valued CNN with randomly-initialized weights. As further baseline, as depicted in the second row of Figure 2, we test a standard transfer learning approach applied to the real-valued CNNs: we pretrain on Iemocap (the same dataset used to train RH-emo) and we then initialize all weights of the SER CNNs but the ones of the final classification layer. The last two rows of Figure 2, instead, show our approach, where we use RH-emo as a feature extractor to feed quaternion-valued CNNs. In the third row only RH-emo pretraining happens, while in the last row both RH-emo and CNNs pretraining are performed. In the latter case, we first pretrain RH-emo, then we pretrain the CNN on Iemocap and finally we re-train the same CNN on different datasets.

A. Experimental Setup

We evaluate RH-emo with 3 benchmark SER datasets:

1) Ravdess, the Ryerson Audio Visual Database of Emotional Speech and Song [55]. 24 speakers, English language, 2542 utterances, 2:47 hours of audio, 8 emotion labels.
2) EmoDb, a Database of German Emotional Speech [56]. 10 speakers, German language, 535 utterances, 25 min of audio, 7 emotion labels.
3) Tess, the Toronto Emotional Speech Set [57]. 2 speakers, English language, 2800 utterances, 1:36 hours of audio, 7 emotion labels.

The preprocessing pipeline for these datasets is identical to the one we applied to Iemocap, as described in Section IV, except for the final normalization step. For the quaternion networks we normalize data between 0 and 1 (as required by RH-emo) and for the real-valued network we normalize to 0 mean and unity standard deviation to permit proper convergence.

We apply this approach to 3 popular CNN architectures with increasing capacity: VGG16 [58], AlexNet [59] and ResNet-50 [60], basing on the Torchvision implementations[4]. These implementations present an adaptive average pooling layer

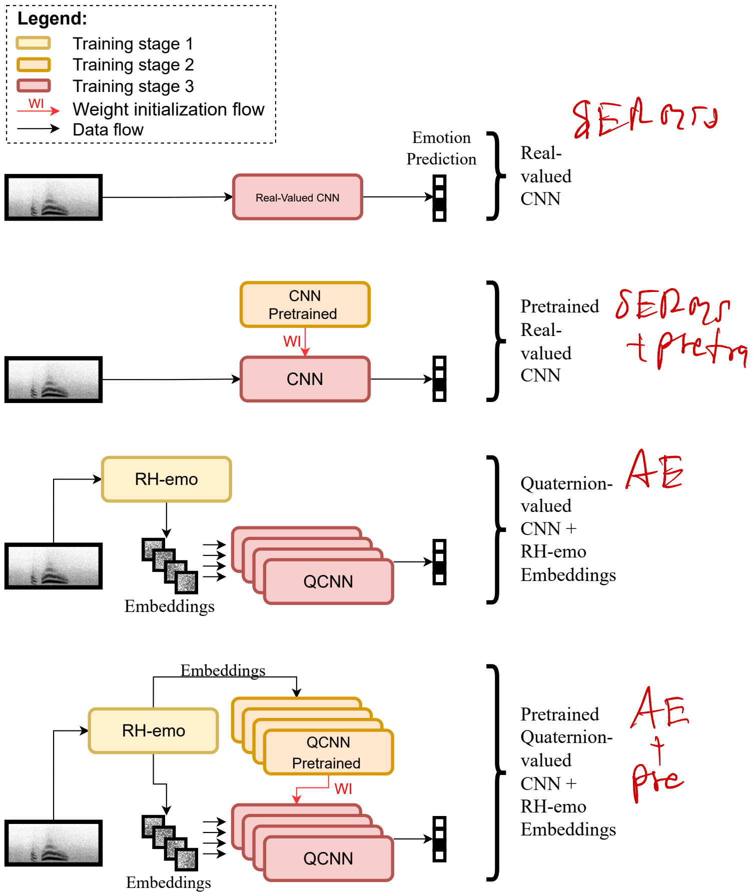


Fig. 2. Block diagram of our experimental setup. The yellow-to-red color coding reflects 3 consecutive training stages. There are 2 separate pretraining stages: RH-emo pretraining (yellow) and CNN pretraining (orange). The real-valued baseline is a regular CNN with random weight initialization, upper row. The pretrained real-valued baseline is the same network, but its weights are initialized with the ones of an identical network pretrained on Iemocap (the same dataset used to train RH-emo), second row. The quaternion-valued network is a quaternion-valued version of the real-valued baselines, which (4 channel) input is generated by forward propagating the input spectrogram in RH-emo's encoder, third row. The pretrained quaternion-valued network is identical to the latter, but the weights of the CNN are initialized with the ones of an identical network pretrained on Iemocap, last row.

TABLE II
RESULTS FOR RAVDESS

Arch.	Method	Params	Train acc.	Test acc.
VGG16	Real	1.6×10^8	47.10	41.06
	RH-emo+Quat	1×10^7	55.50	49.85
	Real-Pre	1.6×10^8	67.86	45.30
	RH-emo+Quat-Pre	1×10^7	67.08	53.79
AlexNet	Real	5.7×10^7	54.55	46.36
	RH-emo+Quat	1×10^7	50.62	43.94
	Real-Pre	5.7×10^7	83.54	51.06
	RH-emo+Quat-Pre	1.4×10^7	63.16	47.58
ResNet-50	Real	2.3×10^7	72.84	43.48
	RH-emo+Quat	4.9×10^6	91.29	**55.15**
	Real-Pre	2.3×10^7	22.16	18.79
	RH-emo+Quat-Pre	4.9×10^6	89.54	52.42

[4] https://pytorch.org/vision/stable/_modules/torchvision.html

TABLE III
RESULTS FOR EMODB

Arch.	Method	Params	Train acc.	Test acc.
VGG16	Real	1.6×10^8	73.42	47
	RH-emo+Quat	1×10^7	79.54	50
	Real-Pre	1.6×10^8	78.16	52
	RH-emo+Quat-Pre	1×10^7	75.00	47
AlexNet	Real	5.7×10^7	63.1	47
	RH-emo+Quat	1×10^7	82.3	49
	Real-Pre	5.7×10^7	71.45	67
	RH-emo+Quat-Pre	1.4×10^7	77.63	71
ResNet-50	Real	2.3×10^7	99.47	48
	RH-emo+Quat	4.9×10^6	99.73	**73**
	Real-Pre	2.3×10^7	100.00	72
	RH-emo+Quat-Pre	4.9×10^6	99.73	46

TABLE IV
RESULTS FOR TESS

Arch.	Method	Params	Train acc.	Test acc.
VGG16	Real	1.6×10^8	99.54	97.62
	RH-emo+Quat	1×10^7	98.87	97.62
	Real-Pre	1.6×10^8	99.95	99.52
	RH-emo+Quat-Pre	1×10^7	98.72	97.85
AlexNet	Real	5.7×10^7	99.18	98.01
	RH-emo+Quat	1×10^7	99.54	98.56
	Real-Pre	5.7×10^7	100.00	98.01
	RH-emo+Quat-Pre	1.4×10^7	99.75	98.81
ResNet-50	Real	2.3×10^7	100.00	97.38
	RH-emo+Quat	4.9×10^6	100.00	**99.76**
	Real-Pre	2.3×10^7	59.88	57.53
	RH-emo+Quat-Pre	4.9×10^6	100.00	99.28

TABLE V
RESULTS STATISTICS

Dataset	Mean improvement			Absolute improvement
	No pret.	Pret.	Overall	
Iemocap	9.74	/	/	7.87
Ravdess	6.01	12.88	9.45	4.09
EmoDb	10.00	-9.00	0.05	1.00
Tess	0.97	13.63	7.30	0.24

between the convolution-based feature extractor and the fully-connected classifier. This permits to obtain an identical output shape from the feature extractor for any input dimension. We removed this layer from the only VGG16, in order to test the behavior of our approach also in this situation. Doing this, in fact, the feature extractor presents a reduced output dimensionality when the networks are fed with the quaternion embeddings (75% smaller than using the real spectrograms), enabling to spare a major number of network parameters.

We configured our experimental setup in order to show the performance difference between real and corresponding quaternion CNNs fed with the emotional quaternion embeddings. Therefore, we paid attention to perform each experiment in as-close-as-possible conditions, rather than optimizing each architecture for each different dataset. For this reason, our results can not be directly compared to the the state-of-the-art for these datasets. For all experiments we used a learning rate of 0.00001, ADAM optimizer, a batch size of 20 samples, we apply early stopping with patience of 20 epochs on the validation loss and we split the training, validation and test sub-sets with 70%, 20% and 10% of the data, respectively.

B. Experimental Results

Table I shows the pretraining results we obtained on Iemocap, while Table III, II, and IV expose the results on Ravdess, EmoDb and Tess, respectively. Table V, instead, shows the average and absolute improvement provided by our approach, among all CNN architectures for each dataset.

The results clearly show that our approach enhances the model's performance while improving their efficiency. For all datasets the quaternion CNNs fed with RH-emo embeddings provide the best test accuracy overall, with an improvement of 6.01 percentage points (pp) for Ravdess, 10 pp for EmoDB, and 0.97 for Tess in the case we do not apply CNN pretraining. The only case where our approach does not improve the test accuracy is with the EmoDb dataset, applying CNN pretraining, where we have a performance drop of 9 pp. In the other cases where we applied CNN pretraining, our approach provides a strong average improvement of 12.88 and 13.63 pp, respectively for Ravdess and Tess. The results computed on Iemocap (Table I and first row of Table V) depict a limit case, where knowledge is not transferred to different data, because the same dataset is used for the RH-emo pretraining and for SER. Therefore here we did not apply any CNN pretraining. Also in this special case is evident that models benefit from the use of quaternion-valued SER CNNs fed with emotional embeddings, with an average improvement of 9.74 pp among all CNN designs we tested.

VI. ABLATION STUDIES

In order to further explore the properties of our approach and to support its foundations, we performed additional experiments and ablation studies. For these studies we applied the same experimental setup presented in Section V, altering only specific details, as described below.

A. Removing RH-emo components

In this study, we alter the RH-emo structure and test the emotion recognition accuracy using the embeddings generated from the modified RH-emo networks. Here we used only the EmoDB dataset. We compared the full RH-emo, as described in Section IV, to the following altered versions:

- Real: identical to the regular network, but the decoder part is real-valued and no split activation is applied to the reconstructed output in the loss function.
- Reconstruction only: we removed the supervised classification branch, resulting in a completely unsupervised real-quaternion hybrid autoencoder.
- Emotion only: we removed the unsupervised reconstruction branch from the network, obtaining a completely supervised and real-valued emotion classification CNN. In this configuration, there are still 4 target outputs, each with a dedicated classifier (discrete emotion, valence, arousal, dominance).

discret Emo

Angr/Hap
Sad/Neu No AE

[illegible] AF

→ v/a/d

→ discrete emo

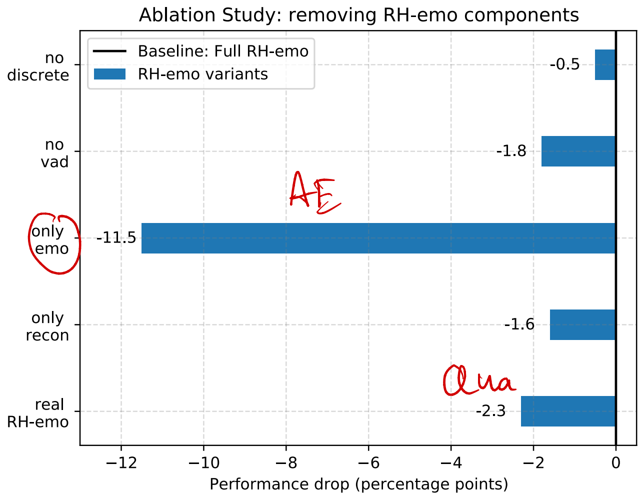


Fig. 3. Ablation study results. The x axis shows the average drop in test accuracy (among the quaternion-valued VGG16, AlexNet and ResNet-50) obtained with different variants of RH-emo. Each row refers to a variant of RH-emo where we removed a specific component, namely: a completely real-valued network, only reconstruction, only emotion recognition, no valence-arousal-dominance estimation, no discrete emotion classification.

avg

- Discrete emotion only: we removed the valence, arousal and dominance classifiers, keeping only the discrete emotion classification branch. The rest of the network is unaltered. D V/A/D
- VAD only: we removed the discrete emotion recognition branch, keeping only the branches for valence, arousal and dominance. The rest of the network is unaltered.

Figure 3 exposes the results of this ablation studies. In the figure we show the mean test accuracy improvement obtained with the quaternion-valued VGG16, AlexNet and ResNet-50 over the real-valued baselines (random initialization and pretrained on Iemocap) fed with the embeddings created with the above-described variants of RH-emo. These results consistently confirm the foundation of our approach. The performance of all variants is inferior to the full RH-emo. In addition, we recall that the quaternion-valued CNNs fed with the emotional embeddings use a considerably lower amount of parameters. The results point out that the unsupervised branch of RH-emo is fundamental to obtain useful embeddings, in fact the emotion-only version, where the decoder part of RH-emo is removed, provides a severe drop in performance compared to all variants and also both baselines. As we expected, the quaternion-valued decoder of the actual RH-emo outperforms the completely real-valued version (by 2.3pp). This supports our hypothesis that a quaternion-value decoder is able to create embeddings that present more suitable intra-channel correlations for the quaternion-valued CNNs. Moreover, also here, the quaternion approach leads to faster (pre)training and less memory demand due to the lower amount of parameters. The completely unsupervised variant (recognition-only) is conceptually similar to R2Hae [50], but it relies on a convolutional design and it is applied to a different domain. This ablation study shows that the addition of a classification branch to R2Hae provides a substantial improvement in performance (by 1.6 pp in our case) and therefore the semi-supervision can be considered a valuable extension to R2Hae. From this ablation study emerges also that the classification of emotion in the valence-arousal-dominance space is more influential in the creation of stronger embeddings. In fact, the RH-emo variant without discrete classification does not only provide a superior accuracy compared to the discrete-only version (by 1.3 pp), but is also the closest variant to the full quaternion-valued RH-emo in terms of performance. This is further supported by fact that, as a result of an extensive grid search, we apply a stronger weight to to VAD term of the loss function (the α term in Equation 4).

B. Removing RH-emo pretraining and backpropagation

We performed an additional ablation study where we alter how the RH-emo weights are initialized and backpropagated during the SER training. Figure 4 depicts the results of this study, showing the average difference in test accuracy per-dataset among all CNN designs. On the one hand, we initialized the weights of RH-emo with random values while we regularly backpropagate the gradients of the RH-emo's encoder layers (blue rows). By doing this, we completely ignore the RH-emo pretraining and we force the QCNN network to perform an end-to-end training, directly learning how to map the real-valued input spectrograms into quaternion-compatible representations to feed the QCNNs. This approach is conceptually similar to (R2He) [50]. On the other hand, we regularly initialize the weights of RH-emo with the pretrained RH-emo network, but we don't backpropagate the RH-emo layers (orange rows). The results of this experiment strongly support the foundation of our approach. The removal of RH-emo pretraining causes a consistent and substantial decrease of the QCNNs test performance, of 29.4, 3.25, and 6.97 pp for Ravdess, EmoDb and Tess, respectively. This confirms the importance of the prior training of the RH-emo encoder, as exposed in Section IV, for the development of adequate quaternion emotional embeddings. On the contrary, the lack of backpropagation of the RH-emo layers does no provide a consistent performance drop. While the performance decreases for EmoDb (25 pp) and for Tess (0.22 pp), a narrow accuracy boost is evident for Ravdess (0.91 pp). Moreover, the performance difference is averagely inferior compared to the no-pretraining case.

C. Reducing training data

As further study, we re-trained all CNNs and QCNNs on the Ravdess dataset, progressively decreasing the amount of training and validation data. The size of the test set, instead, is kept unaltered, in order to obtain results that can be directly compared with the other results presented in this paper. Figure 5 shows the outcomes of this experiment. Each line shows the trend of the average test accuracy among all CNN architectures, at different reduction rates of the data. The yellow and red lines are the baselines, respectively with and without CNN pretraining on Iemocap. Instead, the green and blue lines show the trend for the QCNNs + RH-emo, respectively with and

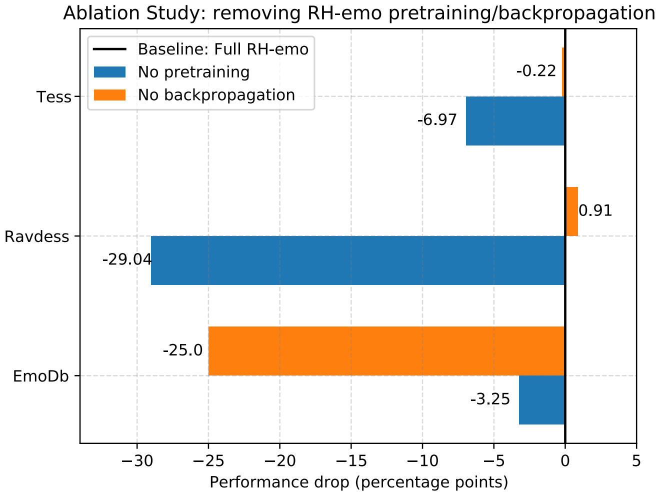


Fig. 4. Ablation study results. The *x* axis shows the average difference in test accuracy (among the quaternion-valued VGG16, AlexNet and ResNet-50) obtained by removing the RH-emo pretraining (blue lines) and backpropagation (orange lines).

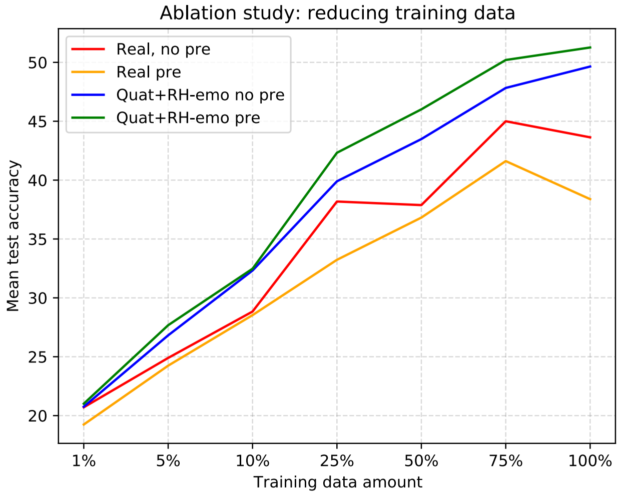


Fig. 5. Ablation study results. The *y* axis shows the average test accuracy among the real-valued (red, yellow) and quaternion-valued (blue, green) VGG16, AlexNet and ResNet-50 architectures for the Ravdess dataset. The *x* axis shows the percentage of available training and validation data used. The 100% of the data corresponds to 1779 utterances for training and 381 for the validation set. The test set is left at 100% in all cases, as for all other presented results. The red line shows the

without CNN pretraining. For this dataset, the 100% of the data corresponds to 1779 utterances for the training set and to 381 utterances for the validation set.

The results of this ablation study clearly point out that our method is an effective solution to overcome the scarcity of emotion-labelled speech data for SER tasks. In all cases our approach (both with and without CNN pretraining) surpasses both real-valued baselines, providing a consistent test accuracy improvement even in the conditions with fewer data. The improvement over the baselines is increasingly larger, as the amount of available data increases. In the 1% data case we have an average improvement of 0.05/1.77 pp respectively for the networks without and with CNN pretraining, which extents to 6.01/12.88 pp when we use the 100% of the data.

VII. Discussion

A. *Resource savings*

RH-emo permits to spare a considerable amount of parameters. Compared to the real counterparts, the quaternion VGG16 uses the ~6% of the parameters, while the quaternion AlexNet and ResNet-50 use the ~25%. The difference between the VGG16 and the others is due to the lack of adaptive average pooling (as described above). Therefore, on the one hand, the use of quaternion-valued layers instead of real-valued ones permits to drop the number of parameters by a factor of 0.25, while, on the other hand, the smaller feature dimensionality obtained with the embeddings further cuts down the number of parameters by a factor of 0.25. This in turn permits to reduce the model's memory requirements and the training time. In our implementation, the embeddings computation happens during the training for every batch and, therefore, both the main network and the RH-emo feature extractor are loaded into the memory. This simulates a plausible application scenario of RH-emo, where the embeddings need to be computed in real-time. Although it is possible to pre-compute the embeddings as part of the pre-processing pipeline, further reducing the memory demand and computation time. As regards the memory demand, in our setup the quaternion networks require on average the 84.2% of memory, compared to their real-valued equivalents. For the VGG16 (where we don't apply average pooling) the memory demand is approximately the 70%, for AlexNet the 89% and for ResNet-50 the 93%. Regarding the training time, the epoch duration of our quaternion networks compared to the real networks is approximately the 15.9% for VGG16, 88.1% for AlexNet and 162.6% for ResNet-50. These outcomes show that the maximum efficiency in terms of both memory demand and computation time is obtained for VGG16, where we take advantage of the reduced dimensionality of the embeddings. On the other hand, the accuracy improvement for ResNet-50 comes at the cost of an increased computation time with respect to the real networks, but still reducing the model's memory demand.

B. *Reconstruction properties*

Figure 6 shows an example of the decoder's output of the pretrained RH-emo model. The *Input* subplot is the input magnitudes-only spectrogram and the *Output: mean* is the element-wise mean of the quaternion separate axes and, therefore, the actual matrix that is compared to the input in the loss function. The sub-plots labelled as *Output: real*, $\hat{\imath}$, $\hat{\jmath}$, $\hat{\kappa}$ depict the separate quaternion axes, which are generated from the emotional embeddings: *real* from the discrete emotion classification matrix, and $\hat{\imath}$, $\hat{\jmath}$, $\hat{\kappa}$ from the valence, arousal and dominance channels, respectively.

By comparing the *Input* and the *Output: mean* plots, it is evident that the reconstruction is not perfect. While the time-wise articulation of the speech seems to be accurately

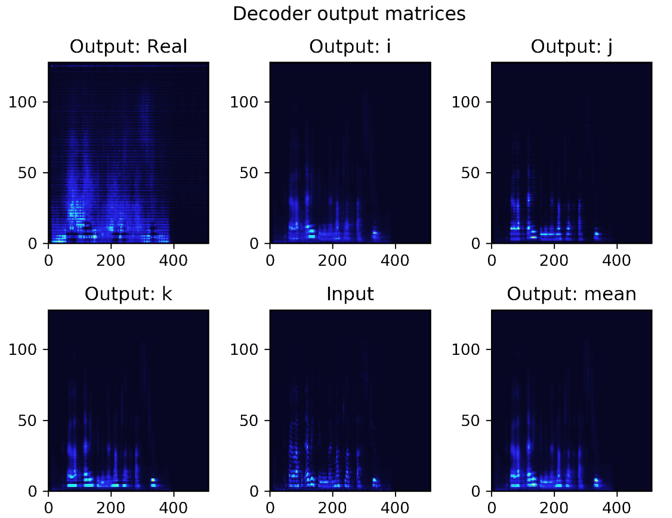


Fig. 6. Example of RH-emo quaternion reconstruction of a speech spectrogram. *Input* is the magnitudes-only input spectrogram, *Output: real*, $\hat{\imath}, \hat{\jmath}, \hat{\kappa}$ are the four output matrices of RH-emo, respectively reconstructed from the discrete emotion, valence, arousal and dominance axes of the embeddings, *Output: mean* is the pixel-wise average of *Output: real*, $\hat{\imath}, \hat{\jmath}, \hat{\kappa}$ and is the matrix that is compared to the input in the loss function.

reproduced, the model is not able to reconstruct in detail the most feeble harmonics of the signal. Although it is interesting the way the different quaternion axes are differentiated. In the real axis the model seems to perform an operation similar to amplitude compression (obtainable, for instance, computing the square root of the matrix), bringing up the signal's quietest portions around the speech region. Instead, in the 3 complex axes ($\hat{\imath}, \hat{\jmath}, \hat{\kappa}$) different aspects of the signal are highlighted, focusing on different harmonics and/or temporal areas. Our intuition is that these representations may represent different ”emotional point of views” of the input speech signal.

C. Limitations

Besides the numerous advantages that our approach provides, there are also some intrinsic limitations. The main constraint of our approach is that a pretrained RH-emo network can be used for only a fixed time scale. In this paper we considered a temporal window of 4 seconds, which is well suited for most SER tasks and datasets. If a different time scale is needed, then a specific RH-emo has to be trained on purpose. Another limitation is that training with an end-to-end fashion is not possible, as a pre-trained RH-emo is needed and the omission of the RH-emo pretraining stage leads to a drastic decrease of model's performance, as shown in Section VI-B.

D. Applications and future work

The advantages provided by the combination of RH-emo and quaternion-valued networks suggest several application scenarios. Due to the substantial saving of trainable parameters, memory and training time, our approach is particularly suited for situations where limited resources are available and performance can not be sacrificed. Another useful property of RH-emo is that while the embeddings carry the necessary information to perform SER tasks (as proven by our experimental results), they also provide speaker anonymity, as it is not possible to reconstruct the input spectrogram without the RH-emo pretrained weights. This could be exploited in situations where sensible speech data must me used for SER tasks.

The positive results we obtained justify further investigation of this approach. An immediate research objective is to test RH-emo with different datasets, architectures (including recurrent networks), with multiple time scales and to different tasks. In particular, we intend to apply the same principle of RH-emo (based on a semi-supervised autoencoder where each embedded channel is optimized for the classification of a different characteristic of an entity) for different tasks, where a quadral representation of input data can not be directly inferred from data, as for speech emotion. An example of this is music genre recognition tasks, where the embedded dimensions of the autoencoder are optimized for tempo, harmonic key, spoken words and instrument type recognition.

VIII. Conclusions

In this paper we presented RH-emo, a semi-supervised approach to obtain quaternion emotional embeddings from real speech spectrograms. This method enables to perform speech emotion recognition tasks with quaternion-valued convolutional neural networks, using real-valued magnitudes spectrograms as input. We use RH-emo pretrained on Iemocap to extract quaternion embeddings from speech spectrograms, where the individual axes are optimized for the classification of different emotional characteristics: valence, arousal, dominance, and overall discrete emotion.

We compare the performance on SER tasks of real-valued CNNs fed with regular spectrograms and quaternion-valued CNNs fed with RH-emo embeddings. We evaluate our approach on a variety of cases, using 4 popular SER datasets (Iemocap, Ravdess, EmoDb, Tess) and with 3 widely-used CNN designs of increasing capacity (ResNet-50, AlexNet and VGG16). Our approach provides a consistent improvement in the test accuracy for all datasets, while using a considerably lower amount of resources. We obtained an average improvement of 6.01 pp for Ravdess, of 10 pp for EmoDb and of 0.97 pp for Tess and we spared up to 94% of the trainable parameters, up to the 30% of GPU memory and up to the 84.1% of training time. Moreover, we performed additional experiments and ablations studies that confirm the properties and foundations of our approach. The results show that the combination of RH-emo and QCNNs is a suitable strategy to circumvent the major challenges of SER: the data scarcity and the high resources demand of models.

The positive results justify further investigation of this approach. An immediate research objective is to test RH-emo with different datasets, architectures (including recurrent networks), with multiple signal dimensions and to different tasks.

References

[1] J. Rybka and A. Janicki, “Comparison of speaker dependent and speaker independent emotion recognition,” *Int. J. of Applied Math. and Comput. Science*, vol. 23, no. 4, 2013.

[2] V. Hozjan and Z. Kačič, “Context-independent multilingual emotion recognition from speech signals,” *Int. J. of Speech Techn.*, vol. 6, no. 3, pp. 311–320, 2003.

[3] S. Rigoulot, E. Wassiliwizky, and M. D. Pell, “Feeling backwards? How temporal order in speech affects the time course of vocal emotion recognition,” *Frontiers in Psychology*, vol. 4, p. 367, 2013.

[4] S. Khorram, Z. Aldeneh, D. Dimitriadis, M. McInnis, and E. M. Provost, “Capturing long-term temporal dependencies with convolutional networks for continuous emotion recognition,” in *INTERSPEECH*, 2017, pp. 1253–1257.

[5] Z. Lian, J. Tao, B. Liu, and J. Huang, “Unsupervised representation learning with future observation prediction for speech emotion recognition,” in *INTERSPEECH*, 2019, pp. 3840–3844.

[6] M. W. Bhatti, Y. Wang, and L. Guan, “A neural network approach for human emotion recognition in speech,” in *IEEE Int. Symp. on Circuits and Syst. (ISCAS)*, vol. 2. IEEE, 2004, pp. II–181.

[7] R. Cowie, E. Douglas-Cowie, N. Tsapatsoulis, G. Votsis, S. Kollias, W. Fellenz, and J. G. Taylor, “Emotion recognition in human-computer interaction,” *IEEE Signal Process. Mag.*, vol. 18, no. 1, pp. 32–80, 2001.

[8] J. Nicholson, K. Takahashi, and R. Nakatsu, “Emotion recognition in speech using neural networks,” *Neural Comput. & Applicat.*, vol. 9, no. 4, pp. 290–296, 2000.

[9] D. Ververidis and C. Kotropoulos, “Fast and accurate sequential floating forward feature selection with the bayes classifier applied to speech emotion recognition,” *Signal Process.*, vol. 88, no. 12, pp. 2956–2970, 2008.

[10] X. Mao, L. Chen, and L. Fu, “Multi-level speech emotion recognition based on HMM and ANN,” in *WRI World Congress on Comput. Science and Informat. Engin.*, vol. 7. IEEE, 2009, pp. 225–229.

[11] T. L. Nwe, S. W. Foo, and L. C. De Silva, “Speech emotion recognition using hidden Markov models,” *Speech Communic.*, vol. 41, no. 4, pp. 603–623, 2003.

[12] J. Zhou, G. Wang, Y. Yang, and P. Chen, “Speech emotion recognition based on rough set and SVM,” in *IEEE Int. Conf. on Cognitive Informat.*, vol. 1. IEEE, 2006, pp. 53–61.

[13] H. Hu, M.-X. Xu, and W. Wu, “GMM supervector based SVM with spectral features for speech emotion recognition,” in *IEEE Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP)*, vol. 4. IEEE, 2007, pp. IV–413.

[14] D. Neiberg, K. Elenius, and K. Laskowski, “Emotion recognition in spontaneous speech using GMMs,” in *Int. Conf. on Spoken Language Process.*, 2006.

[15] M. El Ayadi, M. S. Kamel, and F. Karray, “Survey on speech emotion recognition: Features, classification schemes, and databases,” *Pattern Recogn.*, vol. 44, no. 3, pp. 572–587, 2011.

[16] A. M. Badshah, J. Ahmad, N. Rahim, and S. W. Baik, “Speech emotion recognition from spectrograms with deep convolutional neural network,” in *Int. Conf. on Platform Techn. and Service (PlatCon)*. IEEE, 2017, pp. 1–5.

[17] T.-W. Sun, “End-to-end speech emotion recognition with gender information,” *IEEE Access*, vol. 8, pp. 152 423–152 438, 2020.

[18] D. Issa, M. F. Demirci, and A. Yazici, “Speech emotion recognition with deep convolutional neural networks,” *Biomed. Signal Process. and Control*, vol. 59, pp. 1–11, 2020.

[19] J. Lee and I. Tashev, “High-level feature representation using recurrent neural network for speech emotion recognition,” in *INTERSPEECH*, 2015, pp. 1537–1540.

[20] V. Chernykh and P. Prikhodko, “Emotion recognition from speech with recurrent neural networks,” *arXiv preprint arXiv:1701.08071*, 2017.

[21] G. Trigeorgis, F. Ringeval, R. Brueckner, E. Marchi, M. A. Nicolaou, B. Schuller, and S. Zafeiriou, “Adieu features? End-to-end speech emotion recognition using a deep convolutional recurrent network,” in *IEEE Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP)*. IEEE, 2016, pp. 5200–5204.

[22] P. Meyer, Z. Xu, and T. Fingscheidt, “Improving convolutional recurrent neural networks for speech emotion recognition,” in *IEEE Spoken Language Techn. Workshop (SLT)*. IEEE, 2021, pp. 365–372.

[23] M. A. Qamhan, A. H. Meftah, S.-A. Selouani, Y. A. Alotaibi, M. Zakariah, and Y. M. Seddiq, “Speech emotion recognition using convolutional recurrent neural networks and spectrograms,” in *IEEE Canadian Conf. on Electrical and Comput. Engin. (CCECE)*. IEEE, 2020, pp. 1–5.

[24] Y. Kim, H. Lee, and E. M. Provost, “Deep learning for robust feature generation in audiovisual emotion recognition,” in *IEEE Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP)*. IEEE, 2013, pp. 3687–3691.

[25] Q. Mao, M. Dong, Z. Huang, and Y. Zhan, “Learning salient features for speech emotion recognition using convolutional neural networks,” *IEEE Trans. Multimedia*, vol. 16, no. 8, pp. 2203–2213, 2014.

[26] Z. Huang, M. Dong, Q. Mao, and Y. Zhan, “Speech emotion recognition using CNN,” in *ACM Int. Conf. on Multimedia*. ACM, 2014, pp. 801–804.

[27] K. Han, D. Yu, and I. Tashev, “Speech emotion recognition using deep neural network and extreme learning machine,” in *INTERSPEECH*, 2014, pp. 223–227.

[28] N. Rossenbach, A. Zeyer, R. Schlüter, and H. Ney, “Generating synthetic audio data for attention-based speech recognition systems,” in *IEEE Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP)*. IEEE, 2020, pp. 7069–7073.

[29] A. Laptev, R. Korostik, A. Svischev, A. Andrusenko, I. Medennikov, and S. Rybin, “You do not need more data: Improving end-to-end speech recognition by text-to-speech data augmentation,” in *Int. Congress on Image and Signal Process., BioMed. Engin. and Informatics (CISP-BMEI)*. IEEE, 2020, pp. 439–444.

[30] M. Macary, M. Tahon, Y. Estève, and A. Rousseau, “On the use of self-supervised pre-trained acoustic and linguistic features for continuous speech emotion recognition,” in *IEEE Spoken Language Techn. Workshop (SLT)*. IEEE, 2021, pp. 373–380.

[31] L. Pepino, P. Riera, and L. Ferrer, “Emotion recognition from speech using Wav2vec 2.0 embeddings,” in *INTERSPEECH*, 2021, pp. 3400–3404.

[32] E. Guizzo, T. Weyde, and G. Tarroni, “Anti-transfer learning for task invariance in convolutional neural networks for speech processing,” *Neural Netw.*, vol. 142, pp. 238–251, 2021.

[33] S. Padi, D. Manocha, and R. D. Sriram, “Multi-window data augmentation approach for speech emotion recognition,” *arXiv preprint arXiv:2010.09895*, 2020.

[34] A. Shilandari, H. Marvi, and H. Khosravi, “Speech emotion recognition using data augmentation method by cycle-generative adversarial networks,” *Signal, Image and Video Process.*, 2022.

[35] P. Fewzee and F. Karray, “Dimensionality reduction for emotional speech recognition,” in *Int. Conf. on Privacy, Security, Risk and Trust and Int. Conf. on Social Comput.* IEEE, 2012, pp. 532–537.

[36] N. Patel, S. Patel, and S. H. Mankad, “Impact of autoencoder based compact representation on emotion detection from audio,” *J. of Ambient Intelligence and Humanized Comput.*, pp. 1–19, 2021.

[37] Y. Tay, A. Zhang, L. A. Tuan, J. Rao, S. Zhang, S. Wang, J. Fu, and S. C. Hui, “Lightweight and efficient neural natural language processing with quaternion networks,” in *Proc. of the 57th Ann. Meeting of the Assoc. for Computat. Linguistics*, 2019, pp. 1494–1503.

[38] E. Grassucci, D. Comminiello, and A. Uncini, “A quaternion-valued variational autoencoder,” in *IEEE Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP)*. IEEE, 2021, pp. 3310–3314.

[39] E. Grassucci, E. Cicero, and D. Comminiello, “Quaternion generative adversarial networks,” in *Generative Adversarial Learning: Architectures and Applications*, R. Razavi-Far, A. Ruiz-Garcia, V. Palade, and J. Schmidhuber, Eds. Cham: Springer International Publishing, 2022, pp. 57–86.

[40] E. Grassucci, A. Zhang, and D. Comminiello, “PHNNs: Lightweight neural networks via parameterized hypercomplex convolutions,” *arXiv preprint arXiv:2110.04176*, 2021.

[41] A. B. Greenblatt and S. S. Agaian, “Introducing quaternion multi-valued neural networks with numerical examples,” *Information Sciences*, vol. 423, pp. 326–342, 2018.

[42] T. Parcollet, Y. Zhang, M. Morchid, C. Trabelsi, G. Linarès, R. De Mori, and Y. Bengio, “Quaternion convolutional neural networks for end-to-end automatic speech recognition,” in *INTERSPEECH*, 2018, pp. 22–26.

[43] A. Muppidi and M. Radfar, “Speech emotion recognition using quaternion convolutional neural networks,” in *IEEE Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP)*. IEEE, 2021, pp. 6309–6313.

[44] T. Bulow and G. Sommer, “Hypercomplex signals-a novel extension of the analytic signal to the multidimensional case,” *IEEE Trans. Signal Process.*, vol. 49, no. 11, pp. 2844–2852, 2001.

[45] D. P. Mandic, C. Jahanchahi, and C. C. Took, “A quaternion gradient operator and its applications,” *IEEE Signal Process. Lett.*, vol. 18, no. 1, pp. 47–50, 2010.

[46] D. Comminiello, M. Scarpiniti, R. Parisi, and A. Uncini, “Frequency-domain adaptive filtering: From real to hypercomplex signal processing,” in *IEEE Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP)*. IEEE, 2019, pp. 7745–7749.

[47] R. K. Furness, “Ambisonics–an overview,” in *AES 8th Int. Conf. The Sound of Audio*. Audio Engineering Society, 1990.

[48] D. Comminiello, M. Lella, S. Scardapane, and A. Uncini, “Quaternion convolutional neural networks for detection and localization of 3D sound events,” in *IEEE Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP)*. IEEE, 2019, pp. 8533–8537.

[49] X. Qiu, T. Parcollet, M. Ravanelli, N. Lane, and M. Morchid, “Quaternion neural networks for multi-channel distant speech recognition,” in *INTERSPEECH*, 2020, pp. 329–333.

[50] T. Parcollet, M. Morchid, X. Bost, G. Linarès, and R. De Mori, “Real to H-space autoencoders for theme identification in telephone conversations,” *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 28, pp. 198–210, 2019.

[51] C. J. Gaudet and A. S. Maida, “Deep quaternion networks,” in *IEEE Int. Joint Conf. on Neural Netw. (IJCNN)*. IEEE, 2018, pp. 1–8.

[52] B. C. Ujang, C. C. Took, and D. P. Mandic, “Quaternion-valued nonlinear adaptive filtering,” *IEEE Trans. Neural Netw.*, vol. 22, no. 8, pp. 1193–1206, 2011.

[53] C. Busso, M. Bulut, C.-C. Lee, A. Kazemzadeh, E. Mower, S. Kim, J. N. Chang, S. Lee, and S. S. Narayanan, “IEMOCAP: Interactive emotional dyadic motion capture database,” *Language Resources and Evaluation*, vol. 42, no. 4, pp. 335–359, 2008.

[54] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[55] S. R. Livingstone and F. A. Russo, “The Ryerson audio-visual database of emotional speech and song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English,” *PloS One*, vol. 13, no. 5, 2018.

[56] F. Burkhardt, A. Paeschke, M. Rolfes, W. F. Sendlmeier, and B. Weiss, “A database of German emotional speech,” in *Europ. Conf. on Speech Commun. and Techn.*, 2005.

[57] K. Dupuis and M. K. Pichora-Fuller, “Recognition of emotional speech for younger and older talkers: Behavioural findings from the toronto emotional speech set,” *Canadian Acoust.*, vol. 39, no. 3, pp. 182–183, 2011.

[58] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *Int. Conf. on Learning Representations (ICLR)*, 2015.

[59] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Advances in Neural Inform. Process. Syst. (NIPS)*, vol. 25, pp. 1097–1105, 2012.

[60] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *IEEE Conf. on Comp. Vision and Pattern Recog.*, 2016, pp. 770–778.