City, University of London: MSc in Data Science

Dissertation

2022

Exploring the Use of Quaternion-Valued CNN Layers for Music Audio Source Separation

Benedict Alois Le Fleming Collins

Supervised by: Dr Tillman Weyde

May 23, 2023

Signed Declaration

By submitting this work, I declare that this work is entirely my own except those parts duly identified and referenced in my submission. It complies with any specified word limits and the requirements and regulations detailed in the assessment instructions and any other relevant programme and module documentation. In submitting this work I acknowledge that I have read and understood the regulations and code regarding academic misconduct, including that relating to plagiarism, as specified in the Programme Handbook. I also acknowledge that this work will be subject to a variety of checks for academic misconduct.

Signed: Benedict Alois Le Fleming Collins

Abstract

The incorporation of quaternion-valued layers in neural networks has shown many examples of improving performance whilst almost quartering the number of trainable parameters in contexts that share model architectures with the field of music source separation. In this work, the use of quaternion-valued layers is explored, in the context of music source separation, using a general U-Net architecture that takes Complex as Spectrogram input for all model variants. Performance, training time, model size and the quantity of trainable parameters are all compared between a number of pairs of real-valued and quaternion-valued U-Nets. While the indications of the experiment results suggest that the use of quaternion-valued layers results in lower performance in the context of music source separation, the efficacy of a novel quaternion-valued dropout layer - for use within quaternion models - is demonstrated.

Contents

1 Introduction and Objectives

In recent years, music audio source separation has developed into an automated AI estimation of isolated instrument audio - referred to as sources or stems - that compose a song mixture. Widespread potential applications of this task and similar tasks has drawn interest from academics working in the fields of telecommunications, AI development, audio processing and music information retrieval. The result is a wide body of research into the task, in turn establishing well-defined datasets, model architectures and evaluation methods.

Convolutional neural networks (CNNs) are the most common models used for the task of audio segmentation, inspired by their initial use in the task of image classification [Krizhevsky et al., 2012]. The U-Net architecture, first designed for the task of image segmentation [Ronneberger et al., 2015], has acted as the basis for many different audio source separation tasks since its introduction. Recent incorporation of quaternion values into convolutional neural network (CNN) models have shown promising results for the tasks of speech recognition [Parcollet et al., 2018a], image classification and denoising [Zhu et al., 2019] and emotion classification in speech [Guizzo et al., 2022]. The successful implementation of quaternion CNNs suggest that these models benefit from the added ability to model complex relationships between components of multidimensional inputs, whilst decreasing model complexity through the reduction of learned parameters. Models traditionally employed in these contexts, and CNNs used for music audio source separation, share a lot of similarities and components. Consequently, the results of the above papers suggest that it would be worthwhile to investigate the implementation of quaternion-valued CNNs for stereophonic music audio source separation - a context in which quaternions are yet to be explored.

This project has the potential to add to the body of research, via the introduction of quaternion-based models to the popular context of stereophonic music audio separation. Successful implementation could present an adoption of new methods for music audio separation, benefitting academics and the field of MIR. Adoption of new methods in the field would also present commercial and educational knock-on effects by providing resources to musicians, the general public, the karaoke industry and the music industry as a whole.

1.1 Aims and Objectives

The primary aim of this paper is to establish the effects of the substitution of quaternion-valued layers into a general U-Net model in the context of stereophonic audio source separation. Music audio source separation is a novel context for the use of quaternion-valued models, meaning extensive analysis of effects on performance, training time and model memory size are desirable. Comparisons will be made between standard and quaternion U-Nets with equivalent architectures of different sizes. Due to the fact that quaternion layers will have as little as a quarter of

the trainable parameters that equivalent standard layers contain, comparisons will also be made between models that have been scaled to have an approximately equivalent number of trainable parameters.

The novelty of the context also means that an initial investigation was required to identify a meaningful quaternion input structure for stereophonic audio. On top of this, it is beneficial to have identical input for both standard and quaternion models, as both variations of the U-Net architectures will have access to the same amount of information, allowing comparison to focus on the effects of quaternion-valued operations as opposed to real-valued.

Research question: How does the incorporation of quaternion-valued layers affect the training and performance of convolutional neural networks in the context of music audio source separation?

With this research question in mind, the **objectives** of the project are as follows:

1. Acquire an appropriate dataset for the task of music source separation. Test: Acquisition of at least 1 relevant dataset.
2. Acquire an input structure for the selected dataset that can be utilised in both real-valued and quaternion-valued models. Test: structure and working code for input data.
3. Define a general U-Net architecture. Test: at least one defined structure with working code.
4. Acquire quaternion-valued equivalents of the layers employed within the general U-Net architecture. Test: working code forconvolutional layers as a minimum.
5. Build equivalent standard-valued and quaternion-valued U-Nets based on the general U-Net architecture. Test: working code for both variants
6. Build standard-valued and quaternion-valued U-Nets, scaled to have approximately the same number of trainable parameters. Test: at least 2 implemented model with parameters that have been scaled appropriately
7. Compare the performance, training time, model memory size and the number of trainable parameters for standard-valued and quaternion-valued U-Nets. Test: comparisons in tables and graphs for multiple metrics or variants of quaternion and standard networks.

1.2 Beneficiaries

Improving performance in the task of music source separation presents valuable benefits to musicians by producing higher-quality stems - useful for remixing and sampling, as well as aiding musical analysis for educational purposes. The most notable commercial applications include the removal of vocals from collections of songs for use in the karaoke industry as well as Kanye West's release of the commercially successful STEMPLAYER, which incorporated the - at the time of release - state-of-the-art model for this task [Défossez, 2021]. Further academic interest

for this task results from its potential to enhance the performance of other Music Information Retrieval (MIR) tasks such as audio classification, singer or instrument identification and automated music notation and lyric transcription. Similar methods are utilised for tasks relating to telecommunications, AI development and audio processing, including audio denoising, speech separation and audio separation from environmental noise.

1.3 Work Plan

The original project objective was to explore the effects of substituting quaternion layers into a model that performs at a similar level to the current state-of-the-art models. Initial research into papers relating to stereophonic music audio source separation models yielded a suitable data format, identical for both standard and quaternion-valued models, referred to as complex-as-channel (CaC) [Choi et al., 2020]. This data format is employed in the model that was state-of-the-art at the time of planning the project [Défossez, 2021]. A month was spent trying to train a standard Hybrid Demucs model using the code from the relevant GitHub repository, however, the use of packages with little to no documentation online caused major complications. Moving onto other recent high-performing models, similar complications prevented training, resulting in further delays. The objectives of this task were then altered to analyse the effects of substituting quaternion layers into a standard U-Net model, built from scratch, with both models using a CaC representation for the input.

Audio data preprocessing was the next task to address, consisting of splitting audio into smaller chunks before formatting the chunks into a CaC representation. The following phase involved establishing a general U-Net architecture, based on architectures in related literature. Once the different component layers of this architecture were confirmed, PyTorch implementations of these layers had to be contributed by other academics or coded from scratch, depending on whether implementations were already available. After implementations of the required layers had been acquired, the standard U-Net architecture could be copied and altered to create an equivalent quaternion U-Net architecture. To allow for comparison between models of different sizes, as well as models scaled to have the same number of parameters, the default standard and quaternion U-Net classes had to be altered to take, as arguments, the number of hierarchical layers as well as the number of feature map channels in the output of the first encoder block. This would allow for efficient construction of model variants allowing for wider comparison. Other desirable quaternion U-Net parameters included options for using quaternion or standard valued normalisation, batch normalisation and dropout - allowing for the evaluation of the effects of using quaternion algebra in these types of layers. Model training also needed to be coded in such a way that training hyperparameters and other options are easily adjustable, providing further chance for efficient experimentation.

The process of evaluating whether the objectives have been met must remain the same for all models. The metrics of Source to Distortion Ratio (SDR) and Sources to Artefacts Ratio

(SAR) [Vincent et al., 2006] are used to evaluate the performance of the trained models by analysing the quality of the model output with the desired isolated audio. Source to Interference Ratio (SIR) was also going to be used, however, the package being used to calculate these metrics was outputting infinite values for SIR, so it was removed from the set of evaluation metrics. The training time, as well as the average training time of an epoch is also saved upon training each model, allowing for comparison in training time between models.

1.4 Report Structure

Context (Section 2) contains a description of the development of the U-Net architecture, details and common features of models within the context of audio source separation, an overview of quaternions and their operations, as well as a brief overview of the recent employment of quaternion-valued layers in recent deep learning models.

Methods (Section 3) provides the implementation details of data processing, the previously defined quaternion convolution and batch normalisation layers, quaternion normalisation and novel quaternion dropout. There is then a brief overview of the general U-Net architecture employed during experimentation, as well as the details of the training procedure, the evaluation procedure and, finally, how experiments were planned.

Results (Section 4) is split into 4 subsections, which look at the preliminary results, the results for the task of vocal separation (arranged by architecture size), the results for the task of other source separation and the results for experimental quaternion models, which incorporate real-valued layers.

Discussion (Section 5) contains 3 subsections. The first analyses the performance of pairs of equivalent standard and quaternion U-Nets, with the second analysing pairs that have been scaled to have the same number of parameters. The final subsection contains an analysis of the effects of changing quaternion-valued layers to real-valued layers within quaternion-valued models.

Evaluation, Reflections and Conclusions (Section 6) provides a review of the objectives stated in the introduction, followed by a review of the planning and changes that were made to the process. The penultimate subsection contains ideas for potential extensions to this project that could be explored in the future. The final subsection discusses personal conclusions obtained from the project.

1.4.1 Code Submission Details

Due to the very large amount of memory taken up by the dataset (34.8GB), the formatted dataset input chunks (49.1GB) and the model results (over 200GB), they could not be submitted in the zip file containing the code. All dataset files can be found in my archive on the hyperion remote cluster (filepath: /mnt/data/users/adch204). Model results can be found within the results direc-

tory on the hyperion remote cluster (filepath: /users/adch204/quaternion_unet/model_results). All code is included in the zip file.

Audio output data (in wav format), for each source as well as the original test set wav, are also included in a separate zip file named sample_output_wavs.

2 Context

2.1 CNNs and the U-Net architecture

Utilising CNNs in the context of image processing and analysis tasks, such as image segmentation and classification, was an approach initially explored by Krizhevsky et al. (2012). Ronneberger et al. (2015) further developed the use of CNNs in image segmentation through the introduction of the U-Net architecture, providing pixel-by-pixel localisation. This architecture, outlined in the introduction, has been the primary source of inspiration for subsequent developments of segmentation methods in both image and audio processing.

This general music source separation U-Net architecture consists of an encoder path, a symmetrically structured decoder path and intermediate block between the two. The encoder path consists of the repeated application of blocks containing 2D convolutions, often with batch normalisation, followed by non-linear activation functions. The resulting feature map is then stored in order to be used in skip connections, passing information from the encoder blocks to the corresponding decoder blocks within the architecture hierarchy. Each application of the encoder block scales down the resolution of the feature map by factor w - either through 2D convolution, downsampling or a combination of the two. Simultaneously, the number of feature map channels are doubled - except for in the first block, where the number of feature map channels in the output is specified. After the encoder path, there is an intermediate block, prior to the decoder path, which applies the same layers as those that are contained within an encoder block. This intermediate block, however, doesn't apply the encoder block layer that scales down the resolution of the feature map and doubles the number of channels. The decoder path involves a repeated application of a block that scales up the resolution of the feature map by factor w - via 2D transposed convolution or upsampling - and concatenates the result with the feature map passed from the corresponding encoder block via skip connections. 2D convolutions, and often batch normalisation, followed by non-linear activation functions, are then applied to the concatenated feature map. Each application of an encoder block, inversely to the decoder blocks, halves the number of feature map channels - except for in the final block, where the number of feature map channels in the output is specified.

2.2 Audio Source Separation

The U-Net architecture was initially adapted to handle the task of music source separation by Jansson et al. (2017). The principle of using 2D convolutions on 2D images was preserved through the use of amplitude spectrograms, acquired through Short Time Fourier Transformations, as model input. These amplitude spectrograms have a similar format to images - rather than distance, the x and y axes correspond to time frame and frequency respectively with each

of the time-frequency bins, similarly to the pixels of an image, having an activation. Output for the U-Net in this context is an ideal ratio mask (IRM) corresponding to the model's prediction of the desired isolated audio source. The mask is applied to the original spectrogram through element-wise multiplication, scaling each bin, resulting in a predicted amplitude spectrogram for the isolated audio, whilst taking the phase spectrogram to be the same as that of the mixed audio. Finally, an inverse Short Time Fourier Transform is applied to the spectrograms to acquire the final audio signal output.

Models designed for the task of music audio source separation use a number of different datasets such as iKala and the more recent Slakh2100. The most frequently used datasets within this context are the two versions of the MUSDB18 dataset, either the original compressed version [Rafii et al., 2017] or the uncompressed version, MUSDB18-HQ [Rafii et al., 2019]. Due to the lack of readily available royalty-free, commercially-mixed music and audio stems, these two versions of MUSDB18 have seen general acceptance as benchmark datasets. MUSDB18 has been utilised in an evaluation campaign [Stöter et al., 2018] and the Open-Unmix reference implementation for music source separation [Stöter et al., 2019], as well as the training of many music source separation models [Stoller et al., 2018; Cohen-Hadria et al., 2019; Kadandale el al., 2020; Choi et al., 2020; Defossez et al., 2021; Takahashi and Mitsufuji, 2021]. MUSDB18-HQ has become the current standard for use in training high-performing models [Liu et al., 2020; Defossez, 2021; Liu et al., 2021; Rouard et al., 2022] with competitions such as the Music Demixing (MDX) Challenge 2021 [Mitsufuji et al., 2022], specifying a leaderboard for methods trained specifically on this dataset.

Performance metrics for audio source separation models - established by Vincent et al. (2006) - are standardised, all being measured in decibels, with Source to Distortion Ratio (SDR) acting as the primary metric for evaluation. Due to the different types of undesired audio the models can produce, multiple metrics were designed to provide the ability to separately analyse a model's propensity to produce interferences or artefacts, as it may be desired that a model specialises in prioritising minimising one over the other. The two metrics that often accompany SDR in the evaluation of music source separation models are Source to Interferences Ratio (SIR) and Source to Artefacts Ratio (SAR). The mathematical definition of these metrics are provided in section 3.9.

Since the first audio source separation U-Net, the incorporation phase estimation has been one of the key focal points of the development of subsequent architectures for audio separation tasks, with a wide variety of different implementations. Examples include speech separation models that take complex spectrograms as input, outputting complex IRMs which are then multiplied with the original spectrogram to acquire the estimated isolated source spectrogram (Williamson, Wang and Wang, 2016). Further investigation into phase-incorporation via IRM configurations (Staines, Weyde and Galkin, 2019) includes pairs of mask outputs - either magnitude with phase, magnitude with additive phase, or real with imaginary - combined with a novel loss function designed to account for the circularity of phase. These concepts were tested

on models that take a combination of magnitude, phase, real or imaginary spectrograms as input. The model taking phase and magnitude input with corresponding mask outputs displayed the best SDR, while some of the other model variants also displayed favourable SDR compared to the magnitude spectrogram baseline.

There have been numerous other methods of incorporating phase estimation in audio separation tasks, such as direct estimation of a complex-valued spectrogram [Fu et al., 2017]. Initially introduced for the task of music audio source separation, Stoller et al. (2018) established Wave-U-Net as an adaptation of the U-Net architecture. Substituting 2D layers for 1D layers as well as altering the model to take samples of raw audio signal as input, outputting the estimated raw audio signal for the desired source. This alteration allowed the model to directly incorporate phase into its estimations by working in the waveform domain.

The U-Net Complex as Channel (CaC) framework, introduced by Choi et al. (2020), outlays a data formatting process for audio source separation U-Nets. Input spectrograms are acquired by taking complex-valued short-time Fourier transforms (STFTs) of audio samples with n channels. The real and imaginary components of the spectrogram are then split into two separate real-valued channels, resulting in spectrograms with $2n$ channel. In the case of stereo audio, the 2 channels in the raw audio would result in 4-channel CaC input spectrograms. The model is designed to estimate a spectrogram of the same format as the model input, before transforming it into a complex-valued spectrogram. An inverse STFT (iSTFT) is then applied, producing the output audio signal.

Recent high-performing CNN architectures, for music source separation on the MUSDB18 dataset [Rafii et al., 2017] or the HQ version [Rafii et al., 2019], include additions to the DenseNet [Takahashi and Mitsufuji, 2021] and ResUNet [Liu et al., 2021] architectures, as well as a convolutional extension to the TasNet architecture [Luo and Mesgarani, 2019], which was later adapted for the task of music source separation [Défossez et al., 2021 p. 4]. The current state of the art architecture, Hybrid Transformer Demucs [Rouard et al., 2022] is an expansion on the Hybrids Demucs model [Défossez, 2021], which was state of the art at the time of planning this project. Hybrid Demucs is a dual U-Net architecture, which uses separate spectral and temporal branches for the encoder and decoder, with some shared low-resolution encoder and decoder layers. Both branches provide phase estimation capabilities - the temporal branch takes raw audio signal as input whereas the frequency branch employs a CaC spectrogram framework, displaying the employment of CaC in high-performance models.

2.3 Quaternions

Quaternions are the set of 4th dimensional extensions of complex numbers, denoted $\mathbb{H}$, that take the form:

$$Q = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$$

where a, b, c and d are real numbers and $1, \mathbf{i}, \mathbf{j}$ and $\mathbf{k}$ are the quaternion basis elements that make up the 4 dimensions. In a quaternion, the basis element 1 corresponds to the scalar (real) part, while $\mathbf{i}, \mathbf{j}$ and $\mathbf{k}$ correspond to the vector (imaginary) part.

Having been introduced by Sir William Rowan Hamilton in 1843 as a means of representing 3D rotations prior to the invention of vectors in the late 19th century, quaternion operations - addition, conjugation, scalar multiplication and the Hamilton product are well-defined.

Quaternion operations are defined by the products of the quaternion basis elements:

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

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

Hamilton's defined operations include:

Scalar multiplication, where γ is a real number:

$$\gamma Q = \gamma a + \gamma b\mathbf{i} + \gamma c\mathbf{j} + \gamma d\mathbf{k}$$

Given two quaternions, $Q_1 = a_1 + b_1\mathbf{i} + c_1\mathbf{j} + d_1\mathbf{k}$ and $Q_2 = a_2 + b_2\mathbf{i} + c_2\mathbf{j} + d_2\mathbf{k}$, Hamilton also defined addition:

$$Q_1 + Q_2 = (a_1 + a_2) + (b_1 + b_2)\mathbf{i} + (c_1 + c_2)\mathbf{j} + (d_1 + d_2)\mathbf{k}$$

The Hamilton product is the operation of multiplying 2 quaternions. This corresponds to the sum of each element of the first quaternion, multiplied by each element of the second, before using the products of quaternion basis elements to simplify the result:

$$\begin{aligned} Q_1 \otimes Q_2 =& (a_1a_2 - b_1b_2 - c_1c_2 - d_1d_2) + (a_1b_2 + b_1a_2 + c_1d_2 - d_1c_2)\mathbf{i} \\ &+ (a_1c_2 - b_1d_2 + c_1a_2 - d_1b_2)\mathbf{j} + (a_1d_2 + b_1c_2 - c_1b_2 + d_1a_2)\mathbf{k} \end{aligned} \tag{1}$$

Homogeneously to complex numbers, the conjugate of quaternion Q is given by:

$$Q^* = a - b\mathbf{i} - c\mathbf{j} - d\mathbf{k}$$

The norm of a quaternion Q, often referred to as the Euclidean norm, is equal to the square root of the Hamilton product calculated between itself and its conjugate, Q^*:

$$\|Q\| = \sqrt{QQ^*} = \sqrt{Q^*Q} = \sqrt{a^2 + b^2 + c^2 + d^2}$$

A unit quaternion, is then calculated as:

$$Q_{unit} = \frac{Q}{\|Q\|}$$

2.4 Deep Learning with Quaternions

Quaternion convolution and batch normalisation, introduced by Gaudet and Maida [2017], has acted as the inspiration for further exploration into potential implementations [Grassucci et al., 2021] and uses of quaternion layers in neural networks. In recent years, quaternion-valued layers have been substituted into CNNs in the contexts of speech recognition, speech emotion recognition and image processing. The use of quaternion algebra allows for the modelling of complex relationships between components of multidimensional inputs, whilst significantly reducing the number of learning parameters and, thus, the risk of models overfitting.

Input for quaternion-valued deep learning models take either a 3-dimensional quaternion vector form - with the real component taking the value of 0 - or the standard full quaternion form. Quaternion models for speech recognition have been tested on multiple different quaternion input configurations, such as the distant speech recognition LSTM model that treats 4 microphones as the 4 separate components of a quaternion [Qiu et al., 2020]. Quaternion convolutional neural networks (QCNNs) in this context employ input structures including energy for a specified time and frequency bin, it's first-order time derivative and it's second-order time derivative structured as a 3D quaternion vector [Parcollet et al., 2018b], or it's 4D extension, which incorporates the third-order time derivative of the energy [Parcollet et al., 2018a]. All three studies displayed the ability of quaternion models, within the context of speech recognition, to outperform equivalent real-valued architectures whilst reducing the number of trainable parameters by a factor of almost 4. Quaternion values have also been applied to CNNs for coloured image classification and denoising, employing the representation of RGB input as the i, j and k components of a quaternion vector [Zhu et al., 2019]. Another application of quaternions-valued models involves quaternion value embedding within a speech emotion recognition model, where the real-valued amplitude input is embedded with a quaternion vector composed of 3 elements, corresponding to valence, arousal and dominance [Guizzo et al., 2022].

Despite appearing in many other contexts that utilise CNNs, until this point, quaternion value-structures have not been employed in the field of music source separation to the best of my knowledge. Exploring the use of quaternion-valued layers in music source separation CNNs is, thus, the primary focus of this project.

3 Methods

3.1 Dataset

MUSDB18 [Rafii et al., 2017], and its HQ variant [Rafii et al., 2019], combines data from the now-outdated DSD100 and MedleyDB datasets, resulting in 150 full length track mixtures. Each track mixture is also paired with 4 stems - that of the vocals, bass, drums and other - that are used as the target audio when calculating loss during training and evaluating performance afterwards. Due to the frequent use of MUSDB18, the separation of these particular 4 sources has become the primary objective in music source separation models. All tracks are stereophonic wav files, with the standard sample rate of 44.1 kHz, and the HQ version of this dataset is uncompressed to allow for utilisation of frequencies up to 22 kHz. The dataset selected for this project is MUSDB18-HQ, which employs a standard train-test split of 100 tracks for the training set, and 50 used for the test set.

3.2 Data Processing

One source that is also utilised in music source separation is accompaniment [Jansson et al., 2017; Kadandale el al., 2020; Takahashi and Mitsufuji, 2021] - the track components excluding only the vocals. The stem for this source is not readily provided in the MUSDB18 dataset. Consequently, the first step of data preprocessing is to load the components of the accompaniment - bass, drums and other - and save a wav containing the summation over these three audio sources.

Studies that employ MUSDB18, or MUSDB18-HQ, often downsample the audio as part of the preprocessing - from 44.1 kHz to sample rates such as 8192 Hz [Cohen-Hadria et al., 2019 p. 2], 10880 Hz [Kadandale et al., 2020 p. 4], 11050 Hz [Liu et al., 2020 p. 4]and 22050 Hz [Stoller et al., 2018 p. 4]. This is due to the potential of requiring a large amount of memory for forward passes of input audio containing a large number of samples. In contrast, there are other recent studies [Takahashi and Mitsufuji, 2021], including the previous three state-of-the-art models [Defossez, 2021; Luo and Yu, 2022; Rouard et al., 2022], that use these datasets without downsampling the data. This suggests, following intuition, that models trained on 44.1 kHz data perform better due to the retention of more detailed information within the audio. Although this is desirable, downsampling the data would cause a large decrease to the training and evaluation time of the models, due to the reduction of the number of samples per audio track. Constraints relating to hardware computation-speed, as well as memory capacity, result in the downsampling of the MUSDB-HQ dataset from 44.1 kHz to 22050 Hz for the purpose of this project, reducing the data size whilst not losing too much information.

After downsampling, the next step is to split the tracks into chunks of audio containing

a specified number of samples - determined by the desired frame and frequency dimension sizes for the spectrogram input to the model. Due to the halving and doubling of the receptive field that is employed in the traditional U-Net structure, it is desirable for these spectrogram dimension sizes to be powers of 2 that are large enough for repeated applications of receptive field scaling. The number of samples in each audio chunk was, thus, set to 65536.

Prior to splitting the songs, tracks, and their associated stems, are loaded and then stacked as a singular matrix, with each element of the first dimension corresponding to a different source. Audio from the train set is then cropped so that the total number of frames is a multiple of 65536, removing the need to pad training audio with artificial silence. Conversely, audio in the test set is padded with zeros such that the number of samples in each track is a multiple of 65536. Songs in the train and test sets are then split into chunks of 65536 frames, via matrix reshaping, resulting in a matrix containing the audio chunks for the track and each source. Iterating through each chunk in the matrix, the audio for every source is saved as a singular numpy array for later use. The signal energy of each individual source is then calculated independently for each saved chunk matrix, with the information being saved to an energy profile for the use of filtering out silent audio. A validation split is implemented, with 5% of train set chunks being moved to a validation subset. The final step is to, using the energy profile, acquire the list of chunks with non-silent target audio. This process is carried out for each source separately, with the resulting sizes of each data subset - along with the sizes of the unfiltered subsets - displayed in Table 1. The objectives of this filtering are discussed in section 3.9.

	Vocals	**Drums**	**Bass**	**Other**	**Accompaniment**	**Unfiltered**
Train	5070	6302	6174	6957	7079	7271
Test	2991	3666	3414	3997	4060	2991
Val	260	326	313	359	365	383

Table 1: Number of chunks of audio in filtered and unfiltered MUSDB18-HQ subsets

Data loaders, for the single-source and multi-source separation cases, have the option to filter out chunks with silent audio for the desired isolated sources. If filtering is applied, the data loaders will load tracks only from the list of filtered chunk paths for the desired separated source or sources. When obtaining an audio chunk, each data loader loads the matrix containing the audio for all sources in the chunk. The data loaders then remove the audio for sources that are not to be isolated. The next step is to apply an STFT to the relevant data, before formatting the complex-valued spectrogram into the Complex-as-Channel (CaC) format [Choi et al., 2020].

The STFT is applied to a flattened representation of the stereo audio, with a window size of 1024 and a hop length of 256. These values were established alongside the choice of 65536 samples per chunks, as the output number of spectrogram frequency bins is equal to $(windowlength/2)+1$ and the number of spectrogram frame bins is equal to $(samples/hoplength)$ when the input is padded such that the nth frame is centred at timestep $n*hoplength$. Thus, the

STFT results in a spectrogram with 513 frequency bins and 256 frame bins, with the final frequency bin being removed so that both dimension sizes are powers of 2 ($512 = 2^9$ and $256 = 2^8$). The data is then reshaped to preserve separation between different audio channels and sources. After acquiring the spectrograms for a chunk, the spectrograms' complex dimension is then split into two separate real-valued channels. The dimension containing the real and imaginary channels is then combined with the dimension containing the left and right audio channels, forming a new dimension that contains 4 channels - the real and imaginary components for each of the 2 audio channels.

If the model to be trained is designed to output audio, the corresponding data loader applies the STFT and CaC processing to the mixture audio input and returns this alongside the matrix containing target audio for all specified sources. In the case where the model is designed to output a CaC spectrogram, the corresponding data loader applies the STFT and CaC processing to both the mixture audio input and the target audio matrix, for all specified sources, returning them as a pair.

The final steps of the data pre-processing is to establish functions to reverse the conversion of audio data into a CaC spectrogram, as this will be required to obtain audio output from the models. The first of these takes a spectrogram, in the CaC format, and splits the channels dimension into two dimensions, one for the two audio channels and one for the real and imaginary components. The real and imaginary dimension is then removed, while the components within are combined, to form a complex-valued spectrogram. The second function applies an iSTFT with a window length of $2 * f - 2$, where f is the number of frequency bins in the input spectrogram, and a hop length of 256. The resulting output is stereophonic audio with 65536 samples, matching the length of the model's input chunks.

3.3 Quaternion Convolution

Quaternions, similarly to complex numbers, can be represented as a matrix of real numbers:

$$Q = \begin{bmatrix} a & -b & -c & -d \\ b & a & -d & c \\ c & d & a & -b \\ d & -c & b & a \end{bmatrix}$$

This allows quaternion convolution to be performed using the Hamilton product, as demonstrated by Gaudet and Maida [2017]. Given a quaternion input vector $\mathbf{x} = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$ and a quaternion filter matrix $F = A + B\mathbf{i} + C\mathbf{j} + D\mathbf{k}$, quaternion convolution can be expressed as:

$$F \otimes \mathbf{x} = \begin{bmatrix} A & -B & -C & -D \\ B & B & -D & C \\ C & D & A & -B \\ D & -C & B & A \end{bmatrix} * \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} a' \\ b'\mathbf{i} \\ c'\mathbf{j} \\ d'\mathbf{k} \end{bmatrix}$$

The result is the ability to use convenient real-valued matrix representations of quaternion filters and quaternion input. Quaternion input vectors have the feature of having the channel dimension split into 4 chunks with each representing the different components of a quaternion. Thus, a matrix with n quaternion channels is internally represented as a matrix with $4n$ real-valued channels.

3.4 Quaternion Batch Normalisation

Batch normalisation is a layer frequently employed in many different types of neural networks. One such layer is applied, directly before model activation functions, in order to centre batched input around a mean of 0, with a variance of 1. This has the intended effect of stabilising training, allowing for faster convergence, hence its popularity.

Implementations of quaternion batch normalisation has varied over time. First employed as an extension of complex batch normalisation [Trabelsi et al., 2018], the whitening procedure is adapted to a 4 x 4 covariance matrix through the use of Cholesky decomposition [Gaudet and Maida, 2017].

Due to the intensive computation that is required for this implementation, as well as the disregard for inter-component relations, an alternative batch normalisation implementation is utilised in this project.

A covariance matrix, augmented for use in quaternion algebra, [Cheong Took and Mandic, 2011, p. 5] is utilised along with the assumption that quaternion model input follows the $\mathbb{H}$-properness properties [Cheong Took and Mandic, 2011, p. 5]. This assumption allows for the normalisation of a quaternion, Q, to be calculated as [Grassucci et al., 2021, p. 12]:

$$\tilde{Q} = \frac{Q - \mu}{\sqrt{\text{Var}[Q] + \varepsilon}} \tag{2}$$

with:

$$\mu = \frac{1}{N}\sum_{n=1}^{N} a_n + b_n\mathbf{i} + c_n\mathbf{j} + d_n\mathbf{k} = \bar{a} + \bar{b}\mathbf{i} + \bar{c}\mathbf{j} + \bar{d}\mathbf{k},$$

$$\text{Var}[Q] = \frac{(a-\bar{a})^2 + (b-\bar{b})^2 + (c-\bar{c})^2 + (d-\bar{d})^2}{N},$$

where N is the number of full quaternions in the input, and $\bar{a}$, $\bar{b}$, $\bar{c}$ and $\bar{d}$ are the means of a, b, c and d respectively. The mean of each component can be internally calculated by taking the mean of the relevant chunk of the input channels, with the variance also being calculated via this splitting of the input into the 4 quaternion components, as is consistent with the internal representation of quaternion input and quaternion convolution.

The final two components of this implementation of quaternion batch normalisation are the two trainable parameters - the real-valued scaling factor, γ, and the quaternion offset factor, β. Quaternion batch normalisation, of a quaternion Q, is thus defined, using (Eq. 2), as [Grassucci

et al., 2021, p. 12]:

$$QBN(Q) = \gamma\tilde{Q} + \beta$$

β is internally represented in the same way as the quaternion input vector, with the 4 separate chunks of βs channels corresponding to the 4 quaternion components. β is initialised as a tensor of zeros with the same number of channels as the input. γ is initialised as a tensor of ones with the number of channels equal to the number of full sets of quaternion channels, which is equal to a quarter of the input channels.

3.5 Quaternion Normalisation

The process of normalisation is applied to the batched input data on each forward pass through the quaternion model. The quaternion models defined in this study will be tested on three different normalisation settings to observe the effects on training. These three settings consist of no normalisation, standard-valued normalisation and quaternion-valued normalisation (Eq. 2).

3.6 Quaternion Dropout

QCNNs that utilise dropout, have only used standard dropout as opposed to quaternion-valued dropout [Parcollet et al., 2018a; Parcollet et al., 2018b; Qiu et al., 2020; Guizzo et al., 2022]. Consequently, a PyTorch implementation of quaternion dropout is defined here by the author. The layer defines a mask tensor of random numbers from a uniform distribution using the interval $[0,1)$. This tensor has the same size as the input - excluding the channel dimension, which is divided by 4 so that each channel represents a quaternion. The elements of the tensor are then compared against the p hyperparameter, which corresponds to the probability of an element being zeroed. The resulting boolean tensor contains False values for elements equal to or below the value of p and True values for elements above p. Converting the tensor's data type to torch.int64 results in the substitution of zeros for False values and ones for True values.

Due to the arrangement of the channels of the input, this mask tensor is then concatenated with three copies of itself, along the channel dimension, resulting in a tensor of the same size as the input. This allows the same mask to be applied to the r-channels, i-channels, j-channels and k-channels - arranged in blocks within the channel dimension - having the effect of zeroing out all 4 quaternion components of a CaC spectrogram time-frequency bin rather than treating the different components independently.

3.7 Model Architectures

Once all of the necessary quaternion layers are coded or acquired from other academics in the field, the next task to tackle is building a generalised U-Net structure that uses these layers.

From this general structure, pairs of standard-valued U-Nets and quaternion-valued U-Nets will be defined, trained and then compared after evaluation.

3.7.1 General Encoder Block Structure

The first component of the encoder block involves a 2D convolution with a filter of kernel size 1 x 1, a stride of 1 and no padding. In the first application of an encoder block, the number of output channels for this convolution is specified - with a default of 64 channels for this project. In all subsequent applications of the block, this convolution doubles the number of channels.

The second component of the encoder block is the set of downsampling layers. This begins with a 2D convolution with a filter of kernel size 4 x 4 and a stride of 2, applied to the spectrogram after zero-padding with one zero at the top and bottom of the frame and frequency dimensions. These values were selected such that an application of this convolution reduces the frame and frequency dimensions of the feature map by a factor of 2, effectively downsampling the input. Another important caveat when selecting these values is the need for the kernel size to be larger than the stride so that aliasing - the creation of distortion in the process of downsampling - doesn't occur.

Analogously to similar models, after each convolution, batch normalisation is applied before a leaky ReLU with leakiness 0.2 [Jansson et al., 2017, Staines et al., 2019]. There is also the option of employing dropout with specified p after the leaky ReLU utilised following the 1 x 1 convolution, although the default does not employ dropout.

The code written for this encoder block class first employs the downsampling layers prior to the other layers, with the downsampling layers omitted from the first encoder layer. This means that the intermediate layers, residing between the encoder and decoder paths in the U-Net structure, are provided by the final call of this encoder block class. In order to implement the key U-Net feature of skip connections, the output of each encoder block, excluding the final one which contains the intermediate layers, is stored in a list.

3.7.2 General Decoder Block Structure

The first component of the decoder block is the upsampling layer. This layer is comprised of a 2D transposed convolution with a filter of kernel size 4 x 4 and a stride of 2, applied to the spectrogram after padding with one zero at the top and bottom of the frame and frequency dimensions. This convolution has the inverse effect of the downsampling convolution, increasing the size of the frame and frequency dimensions by a factor of 2. This convolution also decreases the number of channels by a factor of two to prepare for the execution of the skip connections.

In order to implement the U-Net skip connections, output from the upsampling layer is then concatenated, along the channel dimension, with the stored output of the decoder layer of the same hierarchy. These two feature maps have the same sized frame and frequency dimensions,

as well as the same number of channels. Thus, the concatenated tensor has the same number of channels as the input to the transposed convolution.

The second component of the decoder block is a 2D convolution with a filter of kernel size 1 x 1, a stride of 1, identical to the one applied in each encoder block. In the last application of a decoder block, the number of output channels for this convolution is specified. In all prior applications of the block, this convolution halves the number of channels.

Mimicking the decoder block, after each convolution, batch normalisation is applied before a leaky ReLU with leakiness 0.2. There is also the option of employing dropout with specified p after the leaky ReLU utilised following the 1 x 1 convolution. The default general U-Net utilises dropout on the first $\lfloor \frac{n}{2} \rfloor$ layers, where n is the number of hierarchical layers in the U-Net model.

Due to how the encoder and decoder classes are defined, the internal representation replaces the final application of the decoder block class with a 1 x 1 convolution. Batch normalisation and activation layers are ommited, due to the need to predict potential negative real or imaginary components that could be found in a CaC spectrogram. This final convolution also reduces the number of channels to that of the original input.

3.7.3 General U-Net Structure

The U-Net takes a batch of CaC spectrograms as input, normalising this data by default - although model variants are also trained without this initial normalisation. Data is then fed through the U-Net architecture, acquiring the output CaC spectrogram. If the model is designed to output audio, which is the default setting, it will then apply the two functions discussed in the final paragraph of section 3.2. The first of these functions transforms the CaC spectrogram into a complex-valued spectrogram, before the second applies an iSTFT in order to acquire the final audio output.

Below is a table describing the general U-Net structure for this project:

	Layers and Operations
Encoder Block 1	Conv2D($C_{in} = 4, C_{out} = m, K = 1, S = 1, P = 0$) Batch Normalisation, then Leaky ReLU Store feature maps for skip connection 1 Conv2D($C_{in} = m, C_{out} = m, K = 4, S = 2, P = 1$) Batch Normalisation, then Leaky ReLU
Encoder Block for $i = 2, ..., L$	Conv2D($C_{in} = n, C_{out} = n*2, K = 1, S = 1, P = 0$) Batch Normalisation, then Leaky ReLU Store feature maps for skip connection i Conv2D($C_{in} = n*2, C_{out} = n*2, K = 4, S = 2, P = 1$) Batch Normalisation, then Leaky ReLU
Intermediate Block	Conv2D($C_{in} = n, C_{out} = n, K = 1, S = 1, P = 0$) Batch Normalisation, then Leaky ReLU
Decoder Block for $i = L, ..., L-D+1$	TransposeConv2D($C_{in} = n, C_{out} = n/2, K = 4, S = 2, P = 1$) Batch Normalisation, then Leaky ReLU Concatenate with feature maps from skip connection i Dropout(Conv2D($C_{in} = n, C_{out} = n/2, K = 1, S = 1, P = 0$)))) Batch Normalisation, then Leaky ReLU
Decoder Block for $i = L-D, ..., 2$	TransposeConv2D($C_{in} = n, C_{out} = n/2, K = 4, S = 2, P = 1$) Batch Normalisation, then Leaky ReLU Concatenate with feature maps from skip connection i Conv2D($C_{in} = n, C_{out} = n/2, K = 1, S = 1, P = 0$) Batch Normalisation, then Leaky ReLU
Decoder Block 1	TransposeConv2D($C_{in} = n, C_{out} = n/2, K = 4, S = 2, P = 1$) Batch Normalisation, then Leaky ReLU Concatenate with feature maps from skip connection 1 Conv2D($C_{in} = n, C_{out} = 4, K = 1, S = 1, P = 0$))

Table 2: General U-Net architecture: where L = number of layers, D = number of dropout layers and m = number of output channels for the first encoder block

3.8 Training Procedure

Previous studies that use spectrogram masking have employed the $L_{1,1}$ norm, calculated as the sum of absolute values of a matrix, applied to the matrix containing the error between the masked input spectrogram and the target spectrogram [Jansson et al., 2017 p. 2; Cohen-Hadria et al., 2019 p. 2]. In contrast, other studies have used mean squared error (MSE) calculated between the target audio and the output audio in a batch [Stoller et al., 2018, p. 4].

Further examples of loss calculated directly between audio include the use of mean absolute error (MAE) - also referred to as L1 loss - as well as mean squared error, both of which are shown to be viable options in the context of source separation [Défossez et al., 2021, p. 6]. As a result, we calculate loss directly between the output and target audio, providing command line options to use either L1 loss or MSE as the loss function.

A common choice of optimiser in the field of deep learning is the ADAM optimiser [Kingma and Ba, 2017]. In the work encountered during planning, it was the most common choice of optimiser for music source separation models [Jansson et al., 2017 p. 2; Cohen-Hadria et al., 2019 p. 2; Défossez et al., 2021 p. 8]. In many cases, the ADAM optimiser was utilised with beta values $\beta_1 = 0.9$ and $\beta_2 = 0.999$, which are the PyTorch defaults [Stoller et al., 2018 p. 4; Perez-Lapillo et al., 2019 p. 2; Rouard et al., 2022 p. 3]. During training we also utilise the ADAM optimiser with betas $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and an initial learning rate of $\eta = 0.001$, which will act as a command line argument so that it can be adjusted easily. Initial experiments with smaller models are carried out using 20 training epochs and a batch size of 16, in order to check the training procedure executes properly as well as ascertaining preliminary results.

The training loop uses automated model checkpointing and documentation of the training time, training hyperparameters and the hyperparameters that define the model architecture. Later experiments also save the model memory size as well as producing a sample of corresponding model output and target spectrograms in addition to training graphs.

Finally, the trained model is applied to a full-length sample track. The sample wav is zero-padded such that its length is a multiple of the audio chunk length used for the model input, 65536, before being split into chunks through reshaping. The different chunks are then converted into CaC spectrograms, as in section 3.2. The spectrograms are squeezed, to simulate the model's batched input, and then fed through the model. The model's audio output for each chunk is iteratively concatenated into a single tensor, from which the full audio output is acquired through reshaping. The audio is then saved, allowing for verification that the models are outputting audio.

3.9 Evaluation

The standard metrics used for evaluation of music audio source separation models are - measured in dB - the Source to Distortion Ratio (SDR), Source to Interferences Ratio (SIR) and Sources to Artefacts Ratio (SAR), as established by Vincent, Gribonval and Fevotte (2006, p. 4). These metrics are defined using the decomposition of an estimate of an audio source, denoted $\hat{s}$, into four components:

$$\hat{s} = s_{target} + e_{interference} + e_{noise} + e_{artefact}$$

SDR corresponds to how much distortion can be heard relative to the separated source and is generally the most highly prioritised metric for this task. SDR is calculated as:

$$SDR = 10\log_{10} \frac{\|s_{target}\|^2}{\|e_{interference} + e_{noise} + e_{artefact}\|^2}$$

SIR is used as a measurement of the suppression of leakage from other audio sources into the desired separated audio. The formula for SIR is:

$$SIR = 10\log_{10} \frac{\|s_{target}\|^2}{\|e_{interference}\|^2}$$

SAR corresponds to the amount of artefacts - undesired audio that is created by the alteration of audio - relative to the desired source signal. SAR is calculated as:

$$SAR = 10\log_{10} \frac{\|s_{target} + e_{interference} + e_{noise}\|^2}{\|e_{artefact}\|^2}$$

A key limitation of SDR, and SIR, is that values are very low for near-silent target audio and undefined for silent target audio, due to the logarithmic dB scale. This is because, for silent target audio, $\|s_{target}\| = 0$ resulting in $SDR = SIR = 10\log_{10}(0)$, which is undefined. These very small, or undefined, values act as outliers that have the effect of distorting the mean and standard deviation (STD) of these metrics. Previous studies in the field have tackled this issue by removing track sections with silent or near-silent target audio [Rouard et al., 2022, p. 3]. Similar projects simply evaluate source separation models with these metrics by calculating their median value [Jansson et al., 2017, p. 4; Defossez et al., 2021, p. 8; Defossez, 2021, p. 7; Rouard et al., 2022, p. 4], with some also including the median absolute deviation (MAD), which acts as an alternative to STD that is less sensitive to outliers [Stoller et al., 2018, p. 5; Perez-Lapillo et al., 2019, p. 3; Cohen-Hadria et al., 2019, p. 4].

A new SDR measure, introduced by the MDX challenge [Mitsufuji et al., 2022, p. 3], denoted nSDR, simplifies the calculation:

$$nSDR = 10\log_{10} \frac{\sum_n \|s(n)\|^2 + \varepsilon}{\sum_n \|s(n) - \hat{s}(n)\|^2 + \varepsilon}$$

where $s(n)$ is the target audio waveform, and $\hat{s}(n)$ is the estimated audio waveform, at timestep n. This definition allows for faster evaluation and, thus, less computationally expensive evaluation.

In this project, the original SDR metric is calculated alongside the SIR and SAR metrics - included in order to provide more detailed distinction between each model's performance - using Python's museval package [Stoter et al., 2019]. The new SDR definition is also included in the evaluation code in order to provide the option to speed up model evaluation during preliminary and future experimentation. Metric values are calculated between the model output audio and the target audio for each audio chunk of the length 65536, as in section 3.2. The

mean, STD, median and MAD of each metric will be calculated over the metric values for every chunk in the subset used for evaluation. Evaluation is separately carried out on the test set and the validation set, as well as the train set so that the models can be checked for overfitting. Each statistic calculated for each metric is also presented alongside the maximum and minimum metric values across all audio chunks in the dataset.

3.10 Experimental Design

Experimentation was designed in order to explore the effects of changes in a number of different hyperparameters and settings, altering both model-building and the training procedure. Due to initial delays in the project - as well as long training times and limitations in available hardware for model training - the number of hyperparameters, used for experimentation, had to be reduced. Models are trained using two different loss functions - L1 and MSE - while learning rate values are altered from 0.001 to 0.0001 on some architectures.

The primary model building hyperparameter that is altered is the number of real-valued channels in the output of the first encoder block, this is due to the desire to compare real-valued and quaternion-valued models with the same number of trainable parameters. The initial values of 16, 32 and 64 are initially selected, as they are consecutive powers of 2. 256 is later added after discovering it is the largest quaternion-valued model that could be trained on the hardware available. Experiments are also conducted using quaternion models that substitute quaternion-valued components - normalisation and dropout - for real-valued equivalents, allowing to test the effects of these layers.

4 Results

The layout of each sub-section, within the results section, is defined by the number of real-valued channels in the output of the first encoder block in the model architecture. Thus a model with n real-valued channels in the output of the first encoder block is labelled as an n-channel model.

4.1 Preliminary Results

Preliminary training is executed in order to ascertain whether training converges for the U-Net and quaternion U-Net, as well as gaining an initial understanding of comparative performance. In order to minimise the training time for these early experiments, training is run for a fixed 20 epochs, with a batch size of 16 and a learning rate of 0.001 for all models in this section. All models are designed to separate vocals, with other sources experimented on further through the project. Two different models are trained for each standard and quaternion architecture - one using MSE loss and the other using L1 loss - to allow for observations to be made about the effects of using different loss functions during training.

4.1.1 64-Channel

Preliminary training is initially executed on the default U-Net architecture described in section 3.7 - with 5 hierarchical layers and 64 real-valued output channels for the first application of the encoder block - alongside its equivalent quaternion architecture.

The first pair of models are trained using MSE loss. Model evaluation results are calculated for each dataset subset for comparison.

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	5.835771	3.825913	5.732732	2.230071	23.15747	-14.3041
SAR	5.929101	5.582066	6.509476	2.701495	26.1496	-28.3171

Table 3: U-Net preliminary test set results: MSE loss, 64-channel

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	4.601176	3.394407	4.57761	1.888581	17.3616	-22.2751
SAR	4.495072	5.524061	5.222907	2.787936	20.33386	-27.0955

Table 4: Quaternion U-Net preliminary test set results: MSE loss, 64-channel

The test set results for the U-Net (Table 3) and quaternion U-Net (Table 4) exhibit comparatively favourable results for the standard U-Net for the task of vocal separation with equivalent

architectures trained on the hyperparameters specified above. The standard U-Net achieves average - both mean and median - SDR and SAR of more than a decibel higher than the corresponding metric statistics achieved by the quaternion-valued network. The spread of the SDR metric values - represented by the STD and MAD statistics - is shown to be higher in the standard U-Net, implying a higher amount of variation in the quality of the separated audio. In contrast, the STD and MAD values for the SAR metric is very similar in both models, with a slightly higher STD for the standard U-Net and a slightly higher MAD for the quaternion U-Net. Although the standard U-Net achieves higher maximum values for SDR and SAR and a significantly higher minimum value for SDR, the quaternion U-Net achieves a slightly higher minimum value for SAR.

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	6.205712	3.911344	6.185704	2.197719	18.95565	-10.8825
SAR	6.280129103	5.744461319	6.577696552	2.466440289	20.95915684	-19.43160148

Table 5: U-Net preliminary validation set results: MSE loss, 64-channel

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	4.642601	3.82274	4.646908	1.721425	13.28946	-15.7794
SAR	4.232657	5.826219	4.834763	2.773452	16.69075	-19.6422

Table 6: Quaternion U-Net preliminary validation set results: MSE loss, 64-channel

Validation set results for the U-Net (Table 5) and quaternion U-Net (Table 6) display similar results to those of the test set. The standard U-Net, evaluated on the validation set, achieves higher SDR and SAR averages than when it's evaluated on the test set. Improvements in SDR for the quaternion U-Net are not as large as those seen for the standard U-Net, while there is also a deterioration in the averages for SAR. Interestingly, for both metrics, the STD increases while the MAD decreases along with the range - calculated by subtracting the minimum value from the maximum value - in comparison to results on the test set.

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	6.826276	3.525469	6.708968	2.056972	27.46035	-12.0674
SAR	7.022191	5.079432	7.403355	2.449097	30.61623	-26.8347

Table 7: U-Net preliminary training set results: MSE loss, 64-channel

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	4.875746	3.628533	4.927438	1.892867	22.27138	-18.293
SAR	4.662672	5.53595	5.307461	2.743568	24.68746	-33.549

Table 8: Quaternion U-Net preliminary training set results: MSE loss, 64-channel

Finally, the training results (Tables 7 and 8) are examined. The standard U-Net results display improved SDR averages of approximately a decibel, alongside reductions in the spread statistics - contrasting with an increase in range - in comparison to the test set evaluation scores. SAR for the standard U-Net also displays the same changes when compared with the scores obtained during evaluation on the test set.

Increased SDR and SAR averages are also observed for the quaternion U-Net's training set results in comparison to the test set results, however the increases are much smaller than the improvements seen for the standard U-Net. Spread statistics for SDR and SAR also stay approximately the same as the corresponding test set values, whereas the range increases for both.

The number of trainable parameters in the standard model is 18116480, decreasing to 4533488 in the quaternion model. It is then calculated that the substitution of quaternion-valued layers, into the specified U-Net architecture, has the effect of reducing the number of trainable parameters by a factor of 3.996146014. Analogously, the model memory size also decreases from 69.15345001220703 MB, for the standard model, to 17.294208526611328 MB, for the quaternion equivalent. This corresponds to a reduction of the model size by a factor of 3.998647866.

In contrast to the number of trainable parameters and model size, the training time is almost doubled for the quaternion model. The standard model has a total training time of 1727.001056432724 seconds, resulting in an average epoch training time of 86.3500528216362 seconds. Total training time for the quaternion model is almost doubled - 3342.2590157985687 seconds - with an average epoch time of 167.11295078992845 seconds.

The second set of 64-channel models have identical architectures to the first pair, however these models are trained using L1 loss.

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	4.713275	3.696071	4.570106	2.137917	23.16937	-12.3732
SAR	4.444644	5.890102	5.047628	2.844553	25.75351	-33.2723

Table 9: U-Net preliminary test set results: L1 loss, 64-channel

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	4.476955	3.424512	4.15546	2.028541	18.43189	-14.326
SAR	4.616993	5.899991	5.399611	2.901087	20.48502	-32.4258

Table 10: Quaternion U-Net preliminary test set results: L1 loss, 64-channel

Similary to the models trained using MSE loss, the standard U-Net (Table 9) outperforms the quaternion U-Net (Table 10)on the mean and median of the SDR metric - however the metric

averages are severely reduced in comparison to the standard U-Net trained using MSE loss. In contrast, the performance metric statistic values for the quaternion U-Net are similar to those observed in the quaternion model trained using MSE loss - with a minor drop in mean SDR, a slightly larger drop in the median SDR and increases in STD and MAD for both metrics. Furthermore, resulting from the above changes, the SAR averages for the quaternion U-Net are higher than the corresponding values for the U-Net for this pair of models as well as the quaternion U-Net trained using MSE loss.

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	5.161135	3.449743	5.010743	2.090883	25.79542	-15.6387
SAR	4.787753	5.600672	5.417637	2.696444	28.47515	-33.2763

Table 11: U-Net preliminary train set results: MSE loss, 64-channel

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	5.025814	3.412826	4.888021	2.141777	22.538	-15.804
SAR	5.063452	5.716503	5.803413	2.758695	26.42569	-34.5299

Table 12: Quaternion U-Net preliminary train set results: MSE loss, 64-channel

Evaluation on the training set for this pair of models (Tables 11 and 12) - similarly to the first pair of models - displays improved metric averages for both models in comparison to values acquired by evaluation on the test set. Due to the improvement of the evaluation metric scores when evaluating on the training set in comparison to the test set, there are implications of model overfitting. Thus, additions are made to the training loop in order to produce graphs of the relevant models allowing for the exploration of potential overfitting.

The number of trainable parameters, model memory size and training time are not examined here, as this pair of models have identical architectures to the first pair of models.

4.1.2 16-Channel Standard Models

Due to the ability for quaternion-valued layers to reduce of the number of trainable parameters, extra standard-valued U-Net architectures are trained with 16 real-valued output channels for the first encoder layer. This adjustment allows for the development of an understanding of the effects of architecture scaling on the performance of the standard U-Net architecture, as well as the relationship between the number of output channels for the first encoder layer and the number of trainable parameters within the general U-Net architecture utilised in this project.

	Mean	STD	Median	MAD	Max	Min
SDR	4.919728	3.476843	4.760507	2.021099	18.2006	-25.3163
SAR	5.003039	5.638425	5.683038	2.738616	20.86163	-27.4779

Table 13: U-Net preliminary test set results: MSE loss, 16-channel

	Mean	STD	Median	MAD	Max	Min
SDR	5.326405	3.53354	5.246371	2.060344	24.89658	-16.3736
SAR	5.385695	5.488152	6.035876	2.651521	27.35725	-26.383

Table 14: U-Net preliminary train set results: MSE loss, 16-channel

Evaluation on the test set (Table 13) reveals large drops, of approximately a decibel, in the averages of both performance metrics for the 16-channel standard-valued U-Net model in comparison to its 64-channel equivalents. In spite of this, the measures of spread - STD and MAD - show little sign of change when compared with the relevant 64-channel models. Meanwhile, train set evaluation (Table 14) displays even larger drops in the averages of both metrics, with decreases of approximately 1.5 decibels.

The model total training time is 893.5962133407593 seconds, almost half of the total training time for the equivalent 64-channel model also trained using MSE loss. Contrary to the assumption that dividing the number of channels by 4 would result in the number of trainable parameters decreasing by a factor of 4, the number of trainable parameters in this model is 1134560, with a model memory size of 4.339241027832031 MB. Dummy models are thus created, without training, to aid in the establishment of an approximate relationship between the number of output channels in the first encoder block and the number of trainable parameters in both standard-valued and quaternion-valued versions of the general U-Net architecture. The number of trainable parameters for each variant of the 5-layer U-Net architecture, utilised throughout this project, are displayed in Table 15. This table indicates that standard-valued U-Nets - with n real-valued output channels in the first encoder block - have approximately the same number of trainable parameters as a quaternion-valued U-Net using $2n$ channels. This relationship is also present for the memory sizes (Table 16) of standard and quaternion models across the different architecture sizes due to both models using the same data types. As a result, model memory size is no longer investigated due to its dependency on the number of trainable parameters.

Number of Channels	Number of trainable parameters	
	U-Net	Quaternion U-Net
16	1,134,560	284,732
32	4,532,160	1,135,224
64	18,116,480	4,533,488
256	289,717,760	72,446,912

Table 15: Number of trainable parameters for different architectures

Number of Channels	Model Memory Size (MB)	
	U-Net	Quaternion U-Net
16	4.339241027832031	1.0864906311035156
32	17.31116485595703	4.330860137939453
64	69.15345001220703	17.294208526611328
256	1105.363410949707	276.3633613586426

Table 16: Model memory size of the highest performing models for each vocal separation architecture

An equivalent 16-channel U-Net model is also trained using L1 loss, as opposed to MSE loss. Evaluation of this model, both on the test set and the train set, yields higher performance metric averages than the model trained using MSE. The STD and MAD of these metrics also increases for the model trained using L1 loss, as well as increases in SAR range by more than 10 decibels, indicating increased inconsistency in separation quality.

	Mean	STD	Median	MAD	Max	Min
SDR	5.093339	3.700315	4.948138	2.139421	21.21442	-15.2168
SAR	5.01484	6.118094	5.632425	2.788815	23.15054	-49.4901

Table 17: U-Net preliminary test set results: L1 loss, 16-channel

	Mean	STD	Median	MAD	Max	Min
SDR	5.753205	3.472431	5.551107	2.130665	26.31933	-14.9944
SAR	5.709234	5.497539	6.305062	2.585521	28.06687	-35.1222

Table 18: U-Net preliminary train set results: L1 loss, 16-channel

During preliminary experiments, evaluation of models on the training set consistently achieved higher mean and median values than evaluations conducted using the test set. Due to these indications of preliminary models potentially overfitting, early stopping is introduced to the training loop. The minimum number of epochs, maximum number of epochs and early stopping patience

all act as command line arguments - with default values 12, 100 and 10, respectively. Finally, batch size is reduced from 16 to 8 in order to allow for the training of larger model architectures.

4.2 Vocal Separation Models

Less experimentation was performed with the training hyperparameters of the standard-valued U-Nets, as they are utilised primarily to provide a benchmark against which to measure the performance of the quaternion-valued models. With this in mind, the results of only the highest performing standard-valued U-Net will be presented for each architecture configuration.

4.2.1 16-Channel

Over the range of selected 16-channel models, the average values for both SDR and SAR, obtained via evaluation on the test set, are higher - by a minimum of a bit over a decibel - for the standard-valued 16-channel (Table 19) in comparison to both of the quaternion models. The standard U-Net returns the largest values for both measures of spread of SDR, whilst also obtaining the lowest SAR value of any of the three models. The quaternion model trained using the same training hyperparameters (Table 20) displays the lowest mean and median values of SDR, simultaneously obtaining the lowest measures of spread for SDR and the highest spread for SAR. Contrary to the results observed during preliminary experimentation, the quaternion model trained using MSE loss (Table 21) more than doubles the median values of SDR and SAR achieved by the quaternion model trained using L1 loss, with large improvements also observed in their mean values. Training the quaternion model with MSE loss, as opposed to L1, results in higher values of STD and MAD for the SDR metric and lower corresponding values for the SAR metric, indicating less consistency in the quality of vocal separation but more consistency in the omition of artefacts. The minimum SDR is also over 9 decibels higher for the L1 loss model, despite its comparatively worse performance in the averages of both metrics. Despite the highest STD and MAD values for SAR occurring for the L1 loss quaternion model, the other two models also display high levels of deviation with very little distinction between them.

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	5.582864	3.871594	5.392664	2.390384	22.97492	-16.3678
SAR	5.574512	6.092423	6.15584	2.919498	24.84248	-52.3313

Table 19: U-Net test set results: L1 loss, lr=0.001

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	2.36916	2.932411	1.793844	1.632472	14.04381	-17.3192
SAR	0.3275	7.68605	2.0771	3.739406	16.61967	-46.0685

Table 20: Quaternion U-Net test set results: L1 loss, lr=0.001

	Mean	STD	Median	MAD	Max	Min
SDR	4.237322	3.654784	4.094345	2.108813	20.55311	-26.5401
SAR	3.696657	6.074296	4.428539	2.969401	23.81847	-32.2967

Table 21: Quaternion U-Net test set results: MSE loss, lr=0.001

Evaluation on the validation set yields higher average metric values across all three models. The largest improvements observed in mean and median SDR values are produced by the standard U-Net (Table 22) and the MSE loss quaternion U-Net (Table 23), respectively. Results for the quaternion model, trained using L1 loss (Table 24), displayed only minor increases in metric averages, whilst the measures of spread increases for SDR and decreases for SAR. Decreases in STD, along with increases in MAD, are observed for both metrics upon evaluating the standard-valued model. Conversely, the highest-performing quaternion model observed decreases in MAD for both metrics. This quaternion model also displays an increase of STD for the SDR metric and a decrease in STD of the SAR metric.

	Mean	STD	Median	MAD	Max	Min
SDR	6.368008	3.744597	6.085261	2.044687	20.97386	-7.11027
SAR	6.369621	6.117284	6.639651	2.605624	21.80697	-40.2291

Table 22: U-Net validation set results: L1 loss, lr=0.001

	Mean	STD	Median	MAD	Max	Min
SDR	2.688231	3.37182	2.266165	2.008176	13.09707	-16.2214
SAR	0.884491	7.274973	2.10534	3.685963	16.40967	-39.5292

Table 23: Quaternion U-Net validation set results: L1 loss, lr=0.001

	Mean	STD	Median	MAD	Max	Min
SDR	4.815676	3.855305	4.911467	2.013823	15.44411	-15.6782
SAR	4.383084	6.037874	5.093011	2.809509	16.89461	-28.9793

Table 24: Quaternion U-Net validation set results: MSE loss, lr=0.001

Evaluation on the training set displays another increase in the average values of both metrics for all models, with larger improvements observed for SAR than SDR. There are also decreases in all STD and MAD values, excluding the MAD of the standard U-Net (Table 25) and the quaternion U-Net trained using L1 loss (Table 26). Due to the repeated increased performance of models when evaluating on the train set, the degree of this increase will be examined to gain an understanding of the extent of model overfitting.

	Mean	STD	Median	MAD	Max	Min
SDR	7.953787	3.551718	7.691455	2.135141	28.91406	-2.84031
SAR	8.374232	4.951311	8.583656	2.36604	30.08893	-31.2507

Table 25: U-Net train set results: L1 loss, lr=0.001

	Mean	STD	Median	MAD	Max	Min
SDR	2.869638	3.25549	2.630385	2.011097	13.63327	-22.4845
SAR	1.32166	6.904456	2.785495	3.31003	16.35099	-45.9299

Table 26: Quaternion U-Net train set results: L1 loss, lr=0.001

	Mean	STD	Median	MAD	Max	Min
SDR	5.258603	3.481855	5.246813	1.904324	22.03517	-17.9754
SAR	5.099978	5.350655	5.769394	2.442574	23.00035	-32.4305

Table 27: Quaternion U-Net train set results: MSE loss, lr=0.001

4.2.2 32-Channel

Analogously to the 16-channel models, when using 32-channels, the standard U-Net architecture (Table 28) outperforms the various quaternion U-Net architectures in mean and median values of both SDR and SAR. Again, the highest of these values for quaternion models are all displayed by the model trained using MSE loss with a learning rate of 0.001 (Table 30). In contrast to the 16-channel models, when comparing quaternion models trained using L1 loss (Table 29) with those trained using MSE loss, the difference in mean and median metric values is severely reduced in comparison to the equivalent 16-channel models.

The standard U-Net model achieves the largest values for the STD and MAD of SDR, indicating inconsistent quality of separation in comparison to the other models. The largest STD and MAD values for SAR belong to the quaternion model trained using L1 loss, which also achieves the lowest minimum SDR and SAR test set values of all models discussed in this section. Maximum SDR and SAR values are the highest for the standard U-Net, with a maximum SDR at least 7 decibels higher than corresponding values for the other models.

	Mean	STD	Median	MAD	Max	Min
SDR	5.836774	3.971137	5.632511	2.411428	25.44237	-15.253
SAR	5.923149	5.988278	6.605744	2.916905	27.61469	-41.2524

Table 28: U-Net test set results: L1 loss, lr=0.001

	Mean	STD	Median	MAD	Max	Min
SDR	4.38884	3.702107	4.293699	2.232679	17.60092	-25.4663
SAR	3.871382	6.732614	4.892762	2.971587	21.479	-44.0517

Table 29: Quaternion U-Net test set results: L1 loss, lr=0.001

	Mean	STD	Median	MAD	Max	Min
SDR	4.55314	3.558264	4.370769	2.072747	18.41647	-15.271
SAR	4.485651	5.832959	5.164406	2.895976	19.71293	-34.4414

Table 30: Quaternion U-Net test set results: MSE loss, lr=0.001

Validation set results are omitted for many of the following sections, due to the lack of insight they provide - with changes to values, in comparison to corresponding test set results, showing little distinction from experiments run on different-sized architectures. The validation set results that have been removed can be found in Appendix A.

Results calculated on the training set display increases in the averages of SDR and SAR for all models, with the largest increases observed for the standard U-Net model (Table 31). Decreases in STD and MAD also occurs for both metrics across all models, indicating improved consistency in the quality of vocal separation. Minimum values for SAR decreases for both quaternion-valued models, while all other minimum and maximum values - of both metrics - increase across all models. Evaluation on the training set, contrary to results observed using the test set, results in the quaternion L1 loss model (Table 32) achieving higher average SDR values than the quaternion MSE loss model (Table 33). The quaternion model trained using MSE loss, however, still outperforms its L1 loss counterpart in average SAR.

	Mean	STD	Median	MAD	Max	Min
SDR	8.030107	3.502931	7.78261	2.122544	28.6666	-1.42101
SAR	8.450013	4.742193	8.605224	2.335654	29.58273	-27.9614

Table 31: U-Net train set results: L1 loss, lr=0.001

	Mean	STD	Median	MAD	Max	Min
SDR	5.256613	3.55769	5.155786	2.11414	20.86234	-19.323
SAR	5.019255	5.898254	5.804974	2.584023	23.81614	-53.0235

Table 32: Quaternion U-Net train set results: L1 loss, lr=0.001

	Mean	STD	Median	MAD	Max	Min
SDR	5.205697	3.466497	5.120089	2.034572	22.78765	-15.9963
SAR	5.260854	5.633733	6.037759	2.587076	23.38523	-34.6179

Table 33: Quaternion U-Net train set results: MSE loss, lr=0.001

Due to the best quaternion U-Net test results being produced by the model trained on MSE loss, a final quaternion model is also trained, using MSE loss with a reduced learning rate of 0.0001, in order to test the effects of a lower learning rate on training. In comparison to the equivalent network trained with a learning rate of 0.001, this model's test set results (Table 34) display almost a halving of the averages of SDR. Very large decreases in the SAR averages, to negative values, are also observed. The measures of spread - except for the MAD of SAR - also decrease in comparison to the model with lower learning rate, while the minimum SDR decreases and the minimum SAR increases. Despite the reduction in the difference of average metric values between test set (Table 34) and training set (Table 35) results indicating a reduction in model overfitting, models trained with a learning rate of 0.0001 are no longer investigated in order to save time.

	Mean	STD	Median	MAD	Max	Min
SDR	1.768884	2.527301	2.342067	0.93146	5.533298	-21.9475
SAR	-0.85872	4.976645	-0.43361	2.945875	14.41159	-23.6995

Table 34: Quaternion U-Net test set results: MSE loss, lr=0.0001

	Mean	STD	Median	MAD	Max	Min
SDR	1.630606	2.879434	2.237544	0.990447	6.048157	-24.5
SAR	-1.20871	4.930776	-0.80197	2.728906	15.76702	-27.241

Table 35: Quaternion U-Net train set results: MSE loss, lr=0.0001

Comparing of the highest performing 32-channel quaternion-valued model (Table 30) with the highest performing 16-channel real-valued model (Table 19), the standard-valued U-Net achieves average metric scores of approximately 1 decibel higher across the board. The measures of spread for SDR and SAR are higher for the standard 16-channel model - reflected by the model's comparatively higher maximum, and lower minimum, metric values. Inspection of the results acquired on the training set (Table 33), the standard-valued U-Net (Table 25) appears to be subject to a higher degree of overfitting, with larger observed increases in the averages of SDR and SAR when evaluating the model on the training set in comparison to the test set results.

Observing Table 15, the number of trainable parameters in each model are approximately equivalent - with the quaternion-valued model having an extra 664 trainable parameters - as

a result of the different architecture sizes. The total training time for the standard model was 5578.64185667038 seconds, with early stopping terminating training after 99 epochs, resulting in an average epoch training time of 56.34991774 seconds. Total training time for the 64-channel quaternion-valued model was 9779.628585338593 seconds, with early stopping terminating training after 56 epochs, resulting in an average epoch training time of 174.6362247 seconds.

It is calculated that the average epoch training time is scaled up by a factor of 3.099138946 for the 32-channel quaternion-valued model.

4.2.3 64-Channel

Across all 64-channel models that were trained, the standard U-Net (Table 36) achieves the highest average SDR and SAR values. Although this is consistent with the results for the 16-channel and 32-channel models, the extent to which the standard models outperforms the quaternion models is reduced. The highest average SDR and SAR values for the 64-channel quaternion models - conversely to the test results for models using 16 or 32 channels - was obtained by the model trained using L1 loss (Table 37), as opposed to MSE loss (Tables 38 and 39). Comparing the standard and quaternion models trained using L1 loss, the standard model returns higher measures of spread for both metrics - excluding the MAD of SAR. The quaternion model achieves higher minimum SDR and SAR values while the standard model obtains higher maximum values for these two metrics, which provides further distinction between the performances of the models that may act as a point of preference for particular music source separation contexts.

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	6.232954	3.942907	6.017421	2.471897	25.25316	-13.8895
SAR	6.277123	6.165673	6.990503	2.894587	27.05304	-40.609

Table 36: U-Net test set results: L1 loss, lr=0.001

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	5.617242	3.616579	5.370573	2.124341	18.98443	-4.78683
SAR	5.621715	6.04678	6.372746	2.927007	19.99186	-20.8777

Table 37: Quaternion U-Net test set results: L1 loss, lr=0.001

Results for two separate models, both trained using MSE loss, are included in this paper due to a lack of convergence of the validation loss of the first MSE loss model. Figure 1 displays incredibly erratic loss values for the validation set during training, with the lowest value occurring at the end of the first epoch. The erratic validation loss values has the effect of triggering the early stopping to complete training early.

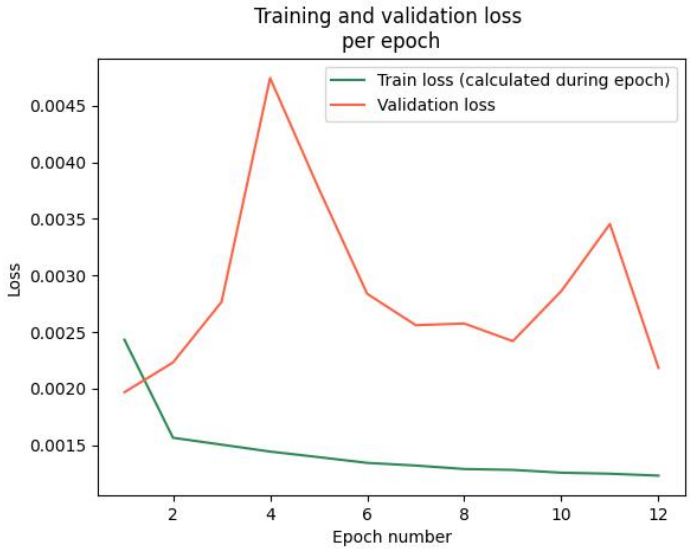

Figure 1: Quaternion U-Net validation and training loss: MSE loss, lr=0.001, First Version

The equivalent graph for the second model (Figure 2) also displays unstable validation loss values, however the validation loss provides a much better approximation of the training loss for this model. Due to the use of the same model architectures, as well as the same training hyperparameters, it is suggested that weight initialisation could be having a large effect on validation loss due to the size of the validation set. The minimum number of epochs employed in early stopping was subsequently increased from 12 to 20, to allow for more extensive training while using the same training data as previously trained models to maintain the capability for comparative evaluation.

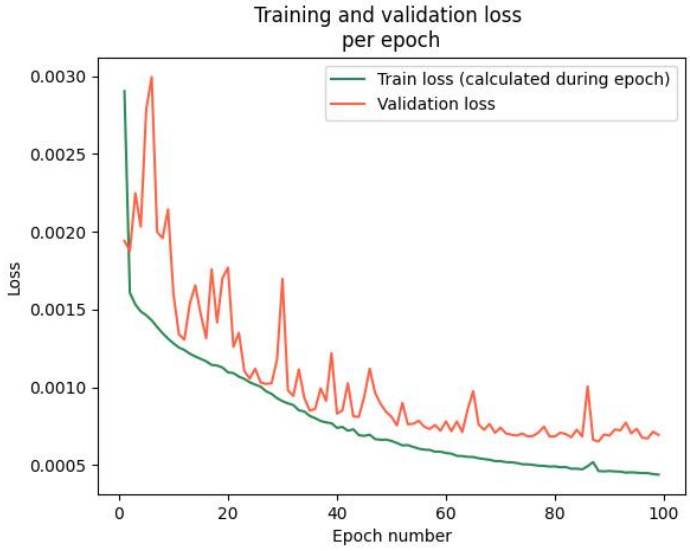

Figure 2: Quaternion U-Net validation and training loss: MSE loss, lr=0.001, Second Version

Comparing the test set results of both quaternion models trained using MSE loss, the second

model (Table 39) achieves significantly higher mean and median values for SDR - almost doubling - as well as SAR - by over 5 decibels. STD also increases for both SDR and SAR, while the MAD increases for SDR and decreases for SAR, indicating a general decrease in separation quality consistency. While extended training causes an increase in the maximum metric values and the minimum SDR value, the minimum value for SAR decreases in the evaluation of the second model. The training set results for the first of these models is omitted from this section, and can be found in the appendix, due to the lack of insights it provides.

Analysing the performance of the second model (Table 39) against the quaternion model trained using L1 loss (Table 37), higher metric averages are obtained by the L1 model. The MSE model shows wider spread of SDR values, with increased STD and MAD, contrasting with a narrower spread of SAR values, when compared with the L1 model. The MSE model also displays lower minimum values and higher maximum for both metrics, which may be desirable considering specified application of a model to a particular piece of music.

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	2.143383	3.318417	2.594932	1.415215	10.19643	-24.4469
SAR	-0.11704	5.02425	0.462411	2.957291	12.98247	-22.3906

Table 38: Quaternion U-Net test set results: MSE loss, lr=0.001, First Version

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	5.346951	3.787185	5.190694	2.309784	20.21257	-12.3026
SAR	5.290759	5.656159	5.900801	2.810758	21.97038	-28.4202

Table 39: Quaternion U-Net test set results: MSE loss, lr=0.001, Second Version

Examining the training results for the standard U-Net and two highest performing quaternion U-Nets, again, demonstrate models overfitting due to increased performance on the training set. The most extreme case of overfitting is displayed by the standard U-Net (Table 40), which observes an increase of more than 3 decibels in the averages of each metric. When evaluated on the training set, the quaternion model trained using MSE loss (Table 42) achieves higher mean and median metric values than the quaternion model trained using L1 loss (Table 41). This indicates that the overfitting is more extreme for the MSE loss variant. Upon inspection of the training graph for the L1 variant (Figure 3), the training loss curve appears to be steeper at the end of training than the training loss curve for the MSE variant (Figure 2). This suggests that training for the L1 variant may not have fully converged prior to the early stopping.

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	9.953022	3.756681	9.683413	2.276367	31.8644	-0.0773
SAR	10.58777	4.542486	10.54643	2.38611	32.67658	-25.7704

Table 40: U-Net train set results: L1 loss, lr=0.001

	Mean	STD	Median	MAD	Max	Min
SDR	6.371895	3.41219	6.251405	2.099394	22.55998	-13.4804
SAR	6.614106	5.504797	7.124061	2.458191	24.65818	-44.5305

Table 41: Quaternion U-Net train set results: L1 loss, lr=0.001

	Mean	STD	Median	MAD	Max	Min
SDR	7.799047	3.443909	7.736507	1.977655	22.26256	-15.8172
SAR	8.084976	4.706616	8.374075	2.206734	24.47694	-25.9403

Table 42: Quaternion U-Net train set results: MSE loss, lr=0.001, Second Version

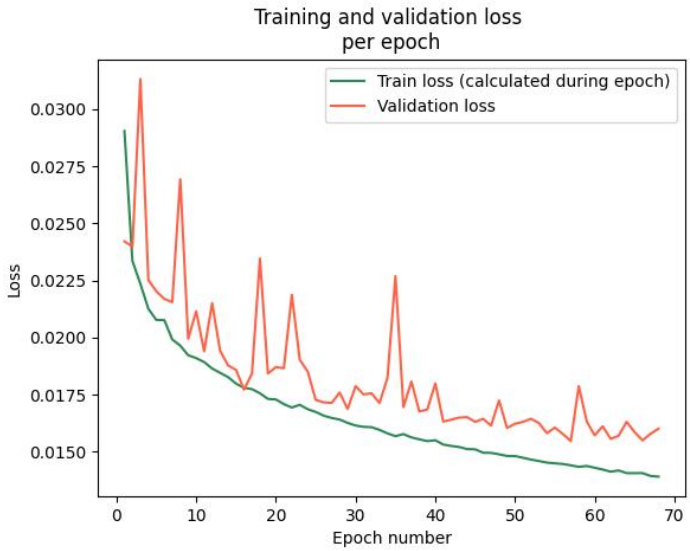

Figure 3: Quaternion U-Net validation and training loss: L1 loss, lr=0.001

Upon comparison of the highest performing 64-channel quaternion-valued model (Table 37) with the highest performing 32-channel real-valued model (Table 28), the standard-valued U-Net displays higher averages in both SDR and SAR. The measures of spread for SDR is higher for the standard 32-channel model, while measures of spread for SAR are lower than those displayed for the 64-channel quaternion model. The quaternion-valued U-Net achieves similar maximum metric values to the standard U-Net, whilst also improving the minimum values for both metrics. Examination of the results acquired on the training set (Table 41), the standard-valued model (Table 31) shows signs of higher degrees of overfitting, with proportionally larger changes in the averages of SDR and SAR.

The number of trainable parameters in each model are approximately equivalent - with the quaternion-valued model containing an extra 1328 trainable parameters - due to the model scaling, as shown in Table 15. The total training time for the standard model was 6957.648246526718

seconds, with the model training for the full 100 epochs, resulting in an average epoch training time of 69.57648246526718 seconds. In contrast, the total training time for the 64-channel quaternion-valued model was 30000.502602100372 seconds, with early stopping terminating training after 74 epochs, resulting in an average epoch training time of 405.4121973 seconds. It is calculated that the average epoch training time is scaled up by a factor of 5.826856762 for the 64-channel quaternion-valued model.

4.2.4 256-Channel

Congruous to the set of models tested for each of the other *n*-channel architectures, the standard-valued U-Net (Table 43) achieves the highest averages for both metrics when evaluated on the test set. The standard model displays the highest values of STD and MAD for SDR, indicating a wider variation of vocal separation quality in comparison to the other two models. The highest maximum SDR and SAR values also belong to this model, however the corresponding minimum values are neither the highest or the lowest of the three models.

Analogously to the 64-channel models, the quaternion model trained using L1 loss (Table 44) outperformed its MSE loss equivalent (Table 45). When trained using L1 loss, the mean SDR is improved by almost 2 decibels, while the median SDR increases by over 1 decibel. Simultaneously, increases in the mean SAR, by over 3.5 decibels, and the median SAR, by just over 5 decibels, demonstrate improved training when using L1 loss. Despite the improved average values, the measures of spread for SAR, as well as the MAD of SDR, are all higher for the L1 variant.

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	6.435215	4.064535	6.225792	2.51831	27.79866	-15.598
SAR	6.534247	6.11491	7.120888	2.930959	30.48662	-48.7758

Table 43: U-Net test set results: L1 loss, lr=0.001

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	4.579137	3.406075	4.204361	2.21677	20.63548	-10.3419
SAR	4.559978	6.974361	5.689054	2.9667	22.53869	-53.6499

Table 44: Quaternion U-Net test set results: L1 loss, lr=0.001

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	2.689449	3.443192	3.143835	1.546177	11.24622	-22.5193
SAR	0.878673	5.09819	1.62627	2.71963	13.49011	-20.3416

Table 45: Quaternion U-Net test set results: MSE loss, lr=0.001

Upon inspection of the training set results, the standard U-Net (Table 46) again displays signs of overfitting, with the averages of both metrics increasing by more than 4 decibels. While signs of overfitting are also observed in the quaternion model trained using L1 loss (Table 47), training set results for the MSE loss variant (Table 48) display lower averages for SDR and lower mean for SAR, indicating model underfitting. This is contrary to the findings produced by examinations of the training set results for all other models, with model underfitting suggesting a very low number of completed training epochs. It is for this reason that the training and validation loss graph is examined for the quaternion MSE loss model (Figure 4).

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	11.15162	3.90678	10.93926	2.41413	33.03405	-0.56431
SAR	11.81333	4.583651	11.80181	2.454517	34.51073	-27.4011

Table 46: U-Net train set results: L1 loss, lr=0.001

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	5.604105	3.329818	5.409813	2.039429	23.6659	-12.6457
SAR	6.14515	5.880838	6.873967	2.543217	24.79128	-33.416

Table 47: Quaternion U-Net train set results: L1 loss, lr=0.001

	Mean	**STD**	**Median**	**MAD**	**Max**	**Min**
SDR	2.552746	3.813698	3.124353	1.724035	12.75951	-25.3127
SAR	0.73494	5.30895	1.640854	2.857551	14.304	-26.9422

Table 48: Quaternion U-Net train set results: MSE loss, lr=0.001

Similarly to the first 64-channel quaternion U-Net, trained using MSE loss (Figure 4), erratic validation loss values result in the termination of training via early stopping (Figure 4). The lowest validation loss value occurs at epoch 9. Combining this with the use of early stopping with 10 epoch patience and a minimum of 20 training epochs, the training is terminated after the first 20 epochs. Due to the fact that this issue has only been encountered in quaternion models trained on MSE, as well as the desire to test many architectures and configurations despite time constraints on training, MSE loss is not utilised in subsequent experiments.

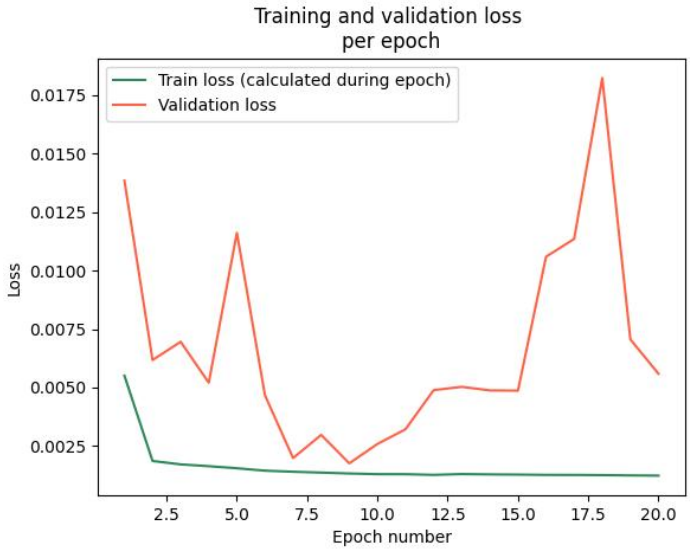

Figure 4: Quaternion U-Net validation and training loss: MSE loss, lr=0.001

4.3 Separation of Other Sources

Due to time constraints, experimentation was minimised for the separation of sources other than vocals. Correspondingly, all models in this section are 64-channel architectures. Due to the improved training of larger models - both standard and quaternion - using L1 loss and a learning rate of 0.001 (Sections 4.2.3 and 4.2.4), these hyperparameters are employed for all further experiments.

4.3.1 Accompaniment Separation Models

SDR and SAR averages - of the standard (Table 49) and quaternion models (Table 50) - for the task of accompaniment separation are the highest of any model presented in this project. Mirroring the results displayed for vocal separation, the standard U-Net outperforms the equivalent quaternion model in the task of accompaniment separation, with higher averages for both SDR and SAR. Measures of spread are also higher for the standard model - indicating less consistency in performance - however, the minimum and maximum values for both metrics are higher than those achieved by the quaternion model.

Overfitting appears to be more prominent for the standard-valued model due to a larger increase in average metric values between the test set and training set results. The large differences in validation and training set averages also indicate that, for this model, the validation set is not a good approximation of performance on the training set, which may further explain issues with erratic validation loss values in the earlier experiments.

The training set results for the quaternion-valued model indicate a lower degree of overfitting, as well as similar values for median metric values to that of the validation set. Interestingly,

while the standard model's training set results display higher maximum and minimum values for both metrics than those in the test set results, the inverse is true for the quaternion-valued U-Net.

Subset	Metric	Mean	STD	Median	MAD	Max	Min
Test	**SDR**	15.08544	8.712816	12.6327	3.738855	45.32058	-6.10818
	SAR	16.22481	9.351183	13.63543	4.176816	51.15787	-6.50498
Validation	**SDR**	17.35087	9.710734	14.05873	4.32703	45.04376	-0.23287
	SAR	18.32518	10.28187	15.29003	4.695661	49.42438	-3.60333
Train	**SDR**	20.76048	11.07073	15.80132	4.391774	45.82883	0.33068
	SAR	21.95163	11.76099	16.68152	4.522986	51.98884	-4.42274

Table 49: U-Net results: L1 loss, lr=0.001

Subset	Metric	Mean	STD	Median	MAD	Max	Min
Test	**SDR**	13.81727	8.309661	11.28576	3.600422	37.0812	-6.14313
	SAR	15.01436	9.343492	12.08728	4.093801	40.065	-6.58748
Validation	**SDR**	15.13381	8.884538	12.17004	4.219612	36.56494	-3.77144
	SAR	16.37923	9.935442	13.15249	4.682178	39.03326	-8.01684
Train	**SDR**	16.09238	9.188875	12.4807	3.88948	36.66331	-9.08258
	SAR	17.25641	10.25115	13.24812	4.23058	40.04753	-11.1714

Table 50: Quaternion U-Net results: L1 loss, lr=0.001

4.3.2 Bass Separation Models

Test set metric averages for both bass separation models are lower than equivalent architectures trained for the tasks of vocal or accompaniment separation. Again, the standard-valued model (Table 51) outperforms the quaternion model (Table 52) in both SDR and SAR averages. The standard model also returns higher measures of spread for SDR and lower STD for SAR, while the MAD of SAR is approximately equivalent for both models. The highest minimum SDR and SAR values are produced by the standard and quaternion models, respectively. Models for bass separation also both display high levels of overfitting, with each metric average in the training set results being more than double the corresponding values in the test set results. In the case of the quaternion model, the validation set results are approximately equivalent to the training set results, with the validation results displaying higher median SDR and SAR.

Subset	Metric	Mean	STD	Median	MAD	Max	Min
Test	**SDR**	4.197456	3.888721	4.14628	2.490231	19.68792	-20.6719
	SAR	3.676531	5.837919	4.184225	3.595133	26.74426	-23.9363
Validation	**SDR**	8.125424	4.181346	7.930857	2.844968	25.96892	-4.74156
	SAR	8.664471	4.978929	8.927799	3.054349	26.3113	-10.5153
Train	**SDR**	9.69845	4.196525	9.239576	2.636116	35.96665	-0.10532
	SAR	10.63564	4.741591	10.41861	2.946763	37.40092	-25.4783

Table 51: U-Net results: L1 loss, lr=0.001

Subset	Metric	Mean	STD	Median	MAD	Max	Min
Test	**SDR**	3.18898	3.36666	2.914809	2.077623	16.11151	-21.9613
	SAR	1.54595	6.336211	2.143071	3.594187	24.27208	-23.1354
Validation	**SDR**	6.370653	3.737765	6.290589	2.49532	22.86945	-5.51228
	SAR	6.455394	5.300751	6.842626	2.751838	28.22214	-19.2684
Train	**SDR**	6.522257	3.692377	6.116384	2.374318	30.27339	-10.2954
	SAR	6.741536	5.069995	6.834146	2.971208	30.95591	-22.0563

Table 52: Quaternion U-Net results: L1 loss, lr=0.001

4.3.3 Drums Separation Models

Models trained for the task of drum separation achieve test set SDR and SAR averages - for standard and quaternion models - lower than those achieved by equivalent architectures for the task of vocal or accompaniment separation but higher than corresponding values for bass separation models. The standard-valued model (Table 53) obtains higher averages for SDR and SAR, higher measures of spread for SDR and lower measures of spread for SAR. Overfitting is present for both models, although the degree of overfitting appears to be lower for the quaternion-valued model (Table 54). The highest maximum and minimum values for both performance metrics are produced by the standard-valued model.

Subset	Metric	Mean	STD	Median	MAD	Max	Min
Test	**SDR**	6.073506	3.490375	5.700678	2.146691	24.81768	-20.3864
	SAR	6.414838	4.598421	6.609257	2.885529	26.73359	-21.0133
Validation	**SDR**	7.91012	4.048783	7.74754	2.294959	28.78817	-1.30492
	SAR	8.405447	4.900629	8.851462	2.66655	29.59589	-10.0712
Train	**SDR**	10.33822	3.81298	10.09555	2.149001	36.02221	-0.35608
	SAR	11.40663	4.234706	11.34932	2.213417	39.74397	-17.9667

Table 53: U-Net results: L1 loss, lr=0.001

Subset	Metric	Mean	STD	Median	MAD	Max	Min
Test	**SDR**	5.088226	3.21404	4.795695	2.055281	19.46602	-22.3744
	SAR	5.015278	4.694793	5.299137	2.998973	23.16887	-21.364
Validation	**SDR**	6.050782	3.307755	5.927343	1.983862	18.29133	-3.07831
	SAR	6.115807	4.756173	6.431597	2.774406	21.04115	-8.86353
Train	**SDR**	7.208921	3.07831	7.084904	1.897297	20.84171	-7.1679
	SAR	7.792204	4.246994	8.034665	2.178851	26.59053	-17.8304

Table 54: Quaternion U-Net results: L1 loss, lr=0.001

4.3.4 Other Separation Models

Evaluation - of models designed to separate the source labeled other - using the test set yields the lowest average SDR and SAR values across all of the sources for standard-valued models, and the second lowest across all sources for quaternion-valued models. Although the standard-valued model outperforms the quaternion-valued model in the averages of both metrics, the difference in SDR averages between the two models' test set results is smaller than those observed for many other pairs of equivalent architectures discussed earlier in the report. The test set results also display both models achieving the same MAD values, to the first decimal place, for both SDR and SAR. In contrast, the STD for SDR and SAR were higher for the standard U-Net when compared with the corresponding values for the quaternion U-Net. The standard model, evaluated on the test set, performs similarly in the median for both SDR and SAR, whereas the mean SDR has a greater mean. The quaternion U-Net achieves lower average SAR scores than SDR in the test set results. Moving onto the validation and training set results, both models have improved averages for SAR in comparison to SDR. Comparing - for both models - the metric averages achieved on the test set and the corresponding values from the training set results, both models also show clear signs of overfitting.

Subset	Metric	Mean	STD	Median	MAD	Max	Min
Test	**SDR**	3.972642	4.645552	3.49337	1.934755	35.16065	-13.6961
	SAR	3.497832	6.340477	3.472211	2.91061	40.99497	-30.7918
Validation	**SDR**	7.296549	4.652094	6.559775	2.026277	36.13033	-4.69221
	SAR	7.670099	5.661225	7.17201	2.6631	41.90463	-13.4939
Train	**SDR**	9.767245	5.300837	8.625038	2.145026	36.72201	-0.22976
	SAR	10.55506	6.016683	9.33538	2.359305	42.22781	-20.3213

Table 55: U-Net results: L1 loss, lr=0.001

Subset	Metric	Mean	STD	Median	MAD	Max	Min
Test	**SDR**	3.232785	3.792035	3.023874	1.924449	28.25863	-17.4902
	SAR	2.593017	5.585293	2.651699	2.95106	31.9797	-20.5641
Validation	**SDR**	6.090442	3.823773	5.477445	1.86471	23.06536	-12.3948
	SAR	6.316161	4.729658	5.999687	2.492818	24.72057	-10.7791
Train	**SDR**	7.441164	4.015098	6.682565	1.91202	28.81768	-5.52723
	SAR	7.85056	4.819155	7.186415	2.317735	32.92558	-19.6219

Table 56: Quaternion U-Net results: L1 loss, lr=0.001

4.4 Experimental Quaternion Models

Experimental quaternion models are trained with the objective of exploring the impact of swapping quaternion-valued components of the model with real-valued equivalents. Two vocal separation 64-channel quaternion U-Nets are constructed - one using real-valued normalisation and one using real-valued dropout. Training is carried out using L1 loss with a learning rate of 0.001. Consequently, the following results will be examined in comparison with (Tables 37, 65 and 41)

4.4.1 Real-Valued Normalisation

The substitution of real-valued normalisation is applied to the initial normalisation of the input data, as well as the batch normalisation layers.

The averages of both SDR and SAR are slightly lower in comparison to those seen in Table 37, with both models displaying their highest metric average in median SAR. There are no significant differences in the STD and MAD metric values between the two models - however the experimental model returned much lower minimum, as well as slightly higher maximum, metric values. Moving onto the validation set and training set results (Tables 65 and 41), the experimental model is shown to achieve higher average values for all metrics. These increases indicate higher levels of overfitting in the experimental quaternion model than its fully-quaternion equivalent.

Subset	Metric	Mean	STD	Median	MAD	Max	Min
Test	**SDR**	5.462533	3.74537	5.269037	2.301573	23.55411	-15.0236
	SAR	5.281708	6.136785	6.092586	2.80323	24.87692	-38.4146
Validation	**SDR**	6.346358	3.877002	6.134769	2.113373	21.79753	-13.2541
	SAR	6.299192	6.333326	6.917721	2.638505	22.26943	-37.3306
Train	**SDR**	8.065947	3.406779	7.89006	2.03355	25.95053	-4.68736
	SAR	8.354293	4.865139	8.575488	2.19527	27.58201	-27.4835

Table 57: Quaternion U-Net (Real-Valued Normalisation) results: L1 loss, lr=0.001

4.4.2 Real-Valued Dropout

Congruous to the quaternion U-Net using real-valued normalisation, real-valued dropout results in a slight decrease across all metric averages when evaluated on the test set. The average SDR values are very slightly lower than those for the real-valued normalisation model (Table 57), while the average SAR values are slightly higher. Similarly to the first experimental quaternion model, but to a further degree, the validation and training set results show increased indications of model overfitting in comparison to the fully-quaternion equivalent (Tables 65 and 41).

When comparing results for the experimental models, the use of real-valued dropout in quaternion-valued vocal separation U-Nets appears to have a larger negative affect on SDR performance and model overfitting than the employment of real-valued normalisation.

Subset	Metric	Mean	STD	Median	MAD	Max	Min
Test	**SDR**	5.407203	3.863065	5.191738	2.37779	21.40688	-14.4798
	SAR	5.533989	6.000885	6.224441	2.994406	25.34263	-44.8758
Validation	**SDR**	6.507951	3.822347	6.237516	2.319763	21.43202	-6.90908
	SAR	7.024491	5.291269	7.213495	2.843715	23.09744	-17.5115
Train	**SDR**	8.125649	3.506544	7.912907	2.15261	24.79192	-10.5045
	SAR	8.769566	5.007976	9.032747	2.389152	29.03674	-37.9495

Table 58: Quaternion U-Net (Real-Valued Dropout) results: L1 loss, lr=0.001

5 Discussion

5.1 Comparing Standard and Quaternion U-Net Vocal Separation Performance

5.1.1 Equivalent Architectures

Contrary to the findings of previous experimentation with quaternion neural networks - compared against real-valued models that use the same input data format - in other contexts [Parcollet et al., 2018a; Parcollet et al., 2018b; Zhu et al., 2019], quaternion vocal separation models are consistently outperformed by their real-valued equivalents. This is true for all n-channel vocal separation models, thereby suggesting that vocal separation models - using the proposed architecture - don't benefit from the use of quaternion-valued layers. The quaternion model that most closely matched the performance of its real-valued equivalent was the 64-channel model, although the average values for both SDR and SAR are still over half a decibel lower.

Complications in early stopping, resulting from the erratic validation loss values - observed in Figures 1 and 4 - may have caused premature termination of the training of quaternion models, as this problem was encountered only during the training of quaternion models. While this is very unlikely to have occurred across all quaternion architectures, this may have had the effect of increasing the extent to which the standard models outperform the quaternion models when using architectures that were subjected to less experimentation. This would explain why the smallest contrast between quaternion and standard model performance occurs for the 64-channel models, as more experiments were carried out using this architecture.

Another possible reasons for lower performance in quaternion models is the appropriateness of CaC spectrogram inputs for use in quaternion models. Despite adhering to the caveat of a meaningful 4-channel data representation [Guizzo et al., 2022 p. 4], the MUSDB-HQ dataset may be largely composed of tracks that utilise only a small degree of panning. Low levels of panning could result similar values across the 4 quaternion components. Similar values across different quaternion components would indicate that CaC spectrogram representation, employed on this dataset, does not take advantage of the ability of quaternion models to learn complex inter-component relationships. It is also possible that the relationship between the real and imaginary values in the audio are too complicated to model for the relatively simple quaternion U-Nets proposed in this paper.

Examining the average epoch training times for the highest performing quaternion-valued and real-valued vocal separation models (Table 59), it is clear that the training of quaternion models takes longer despite reducing the number of trainable parameters by a factor of just under 4 (Table 15). This is due to the extra computations required to execute quaternion op-

erations using real numbers. Over the 16, 32 and 64 channel models, increases in the average epoch training time - in relation to the number of channels - are more gradual for the real valued models, while the rate of increase is considerably higher for the quaternion-valued models. This is reflected by the increase factors of average epoch training time for each pair of models - calculated by dividing the time for the quaternion model by the time for it's real-valued equivalent. This factor increases over the first 3 architectures before decreasing to a value similar to the one observed for the 16-channel model. Due to the decrease in this factor for the 256-channel models, it is acknowledged that the training times observed here could also be affected by factors other than the model size, such as changes in the state of the hardware used to train the models.

	Real-Valued U-Net	**Quaternion U-Net**	**Training time increase factor**
16-channel	56.34991774	109.652945	1.945929105
32-channel	69.57648246526718	174.6362247	2.509989274
64-channel	102.93514178276062	405.4121973	3.938520803
256-channel	564.0257756	1157.328697	2.051907461

Table 59: Average epoch training time (seconds) of the highest performing models for each vocal separation architecture

5.1.2 Architectures with Similar Numbers of Trainable Parameters

Comparisons are made between the highest performing 16 and 32-channel real-valued U-Nets against the highest performing 32 and 64-channel quaternion-valued U-Nets, respectively. Further opposing the results obtained in other contexts [Parcollet et al., 2018a; Parcollet et al., 2018b; Zhu et al., 2019], quaternion vocal separation models are consistently outperformed by real-valued models scaled to have approximately the same number of parameters. Simultaneously, by inspecting Table 59, the average epoch training time for the 32-channel quaternion-valued model is 3.099138946 time longer than the corresponding value for the 16-channel real-valued U-Net. The average epoch training time for the 64-channel quaternion-valued model is 5.826856762 time longer than the corresponding value for the 32-channel real-valued U-Net.

The decreased performance, combined with the large increases in average epoch training times, further reinforce the indications from section 5.1 that the use of quaternion-valued layers causes degradation in both the training time and the performance of CaC spectrogram U-Net models in the context of vocal source separation.

5.2 Comparing Standard and Quaternion U-Net Other Sources Separation Performance

Analogously to the vocal separation models, the real-valued U-Net architectures obtained greater mean and median performance metric values than their quaternion-valued counterparts when evaluated on the test set. Thus, the primary subject of discussion for this section is the comparison of test result performance for each source in comparison to the other sources.

Congruous to previous music source separation models [Jansson et al., 2017 p. 4; Kadandale et al., 2020 p. 5; Takahashi and Mitsufuji, 2021 p. 5], evaluation on the test set yields the highest SDR metric averages for the task of accompaniment separation.

The task of vocal separation yielded the second highest SDR averages, while the source for which the third highest SDR average results were obtained is drums, matching the source result rankings observed in similar studies [Kadandale et al., 2020 p. 5; Takahashi and Mitsufuji, 2021 p. 5]. SDR averages for vocal and drum separation are equal to approximately half of the corresponding values achieved in the task of accompaniment separation.

Test results of the standard-valued U-Net for the task of bass separation display the fourth highest SDR averages across all sources, with results for separation of the source named other ranking in last place. This follows the same rankings observed in previous studies [Kadandale et al., 2020 p. 5; Takahashi and Mitsufuji, 2021 p. 5]. In contrast, the quaternion-valued models achieved the lowest SDR averages for the bass separation task in comparison to the other separation task.

The ranking of real-valued and quaternion-valued model SAR averages are identical to the SDR rankings described for the corresponding model type. This indicates that SAR tends to scale with SDR across the test results for each model trained during experimentation.

5.3 Ablation Studies: Experimental Quaternion U-Nets

5.3.1 Standard or Quaternion-Valued Normalisation

Quaternion-valued normalisation is replaced by the standard-valued equivalent in a 64-channel quaternion vocal separation model (Table 57), with training loss graphs and test set evaluation results examined in comparison with the fully-quaternion equivalent (Table 37). The use of standard-valued normalisation results in a slight degradation of performance based on SDR and SAR.

This alteration also has the effect of increasing the degree of model overfitting. This is unexpected, as one of the main objectives of batch normalisation is to speed up training. Although

the fully-quaternion model used normalisation methods consistent with the algebra employed in the rest of the model, model training takes place over more epochs than for the experimental model. The second objective of batch normalisation is to increase the stability of training.

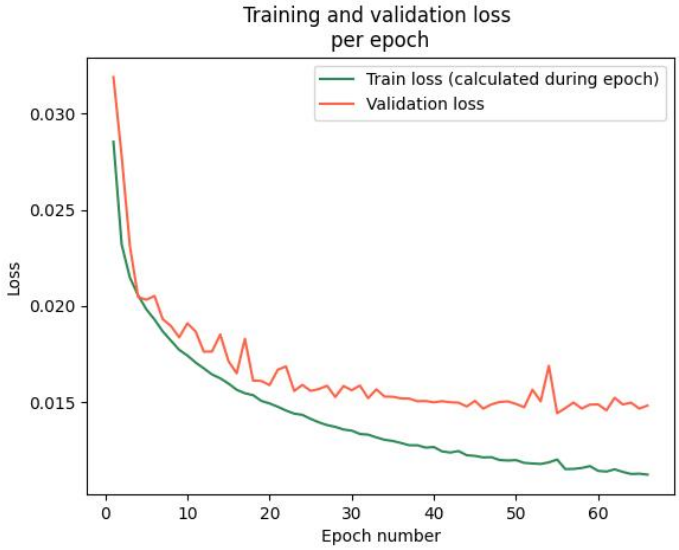

Figure 5: Quaternion U-Net (Real-Valued Normalisation) validation and training loss: L1 loss, lr=0.001

Examining the training graphs for both models (Figures 3 and 5), there is little change in stability of the training loss whereas, validation loss appears more stable in the experimental model. The most clear difference in the training of both models is the earlier, and considerably flatter, plateau of the validation loss. In contrast to the validation set plateaus, the training loss appears to still be decreasing at the end of training for both models.

The results observed suggest that the use of real-valued normalisation in quaternion models has negative effects on the training of vocal source separation models.

5.3.2 Standard or Quaternion-Valued Dropout

Quaternion-valued dropout is replaced by the standard-valued equivalent in a 64-channel quaternion vocal separation model (Table 58), with training loss graphs and test set evaluation results investigated in conjunction with those of the fully-quaternion equivalent (Table 37). Although the use of standard-valued dropout causes a smaller decrease in the average values of SAR, there is a larger degradation of SDR averages, when compared with the standard-valued normalisation model (Table 57).

Changes to the dropout layers, similarly to the other experimental model, increases the degree of model overfitting, with even more of an impact observed in the real-valued dropout model. The primary objective of dropout layers in neural networks is to prevent model overfitting. Thus, the larger degree of model overfitting - caused by the use of real-valued dropout

- follows the idea that altering the algebra of layers within a model would have the impact of decreasing the intended effects of those layers. Similarly (Table 58)

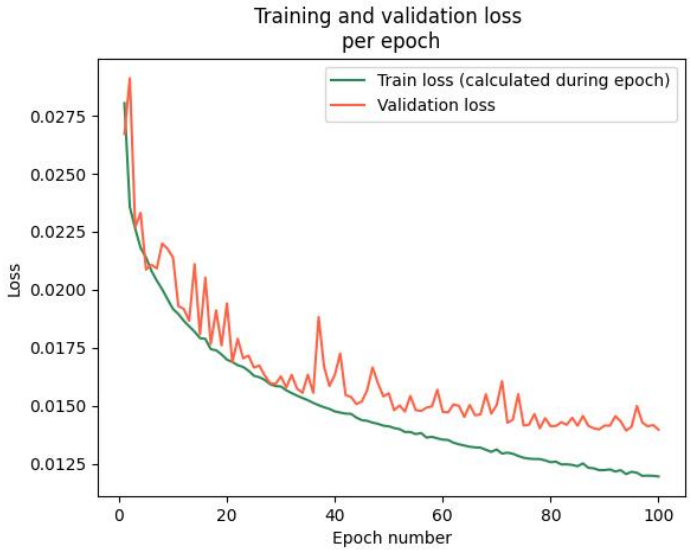

Figure 6: Quaternion U-Net (Real-Valued Dropout) validation and training loss: L1 loss, lr=0.001

Analysing the training graphs for both models (Figures 3 and 6), there is an increase in the rate of fluctuation of the validation loss for the real-valued dropout model, however the peaks are much closer to the training loss values for the first half of the training epochs. Training of the experimental quaternion model also occurs over more than 30 extra epochs when compared with the fully-quaternion model. While the validation loss is closer to the training loss during the early epochs, the training and validation loss values diverge from each other - in the later epochs - to a greater extent than the fully-quaternion model.

The results indicate that the use of real-valued dropout negatively impacts efforts to reduce model overfitting for 64-channel vocal separation quaternion U-Nets, as well as decreasing SDR averages.

5.4 Brief Comparison with the State of the Art

Comparison of the highest-performing models trained over the course of this project with the current model achieving state of the art performance on the MUSDB18-HQ dataset [Rouard et al., 2022], the state of the art model shows improved median SDR of approximately 3dB for vocals, 6dB for bass, 5dB for drums and 4dB for other. The state of the art model is not trained to separate accompaniment, so no comparison can be made for this source.

6 Evaluation, Reflections, and Conclusions

Despite the apparent negative effects of the use of quaternion-valued U-Nets in the context of music source separation, it is recommended that further experimentation is carried out within this context. This is due to the positive results achieved by quaternion U-Nets in contexts that utilise similar types of models [Parcollet et al., 2018a; Parcollet et al., 2018b; Zhu et al., 2019; Guizzo et al., 2022].

6.1 Review of Objectives

Upon reviewing the adjusted objectives, listed in section 1.1, the project was a success. The MUSDB18-HQ dataset was selected for this project based on its widespread use as a benchmark dataset in the training of other models within this context , fulfilling Objective 1.

The desire for a suitable input structure - that could be utilised in both real and quaternion-valued networks - was satisfied by the use of the Complex as Channel spectrogram format, satisfying Objective 2. This allowed for real and quaternion models to be trained using data containing identical information so that the focus of comparison would be related only to the introduction of quaternion algebra.

Initially, a single-layer U-Net architecture was defined before the complexity of the general architecture was increased to a sufficient degree that Objective 3 can be considered complete.

Quaternion implementations of convolutional layers were acquired after contact with another academic - Eric Guizzo - who has recently experimented with quaternion CNNs [Guizzo et al., 2022]. This satisfied the test for Objective 4, however, more layers were acquired.

Quaternion batch normalisation implementation was acquired from the official GitHub repo associated with the Quaternion Generative Adversarial Networks paper [Grassucci et al., 2021]. A Quaternion dropout definition was developed and implemented by the author, which is novel to the best of my knowledge.

In order to fulfill Objective 5, the acquired quaternion layers were then substituted into the general U-Net architecture, allowing for the building of real-valued and quaternion-valued models with equivalent architectures.

Once the preliminary model training had been run, dummy models of different sizes were constructed. Scaling the models, through the scaling of the number of real-valued output channels of the first encoder layer, provided insight into the relationship between this model hyperparameter and the number of trainable parameters in each type of model (Table 15). The exploration of this relationship allowed for the training of real-valued and quaternion-valued U-Nets constructed to have approximately the same number of trainable hyperparameters, resulting in the completion of Objective 6.

The results section primarily analyses the performance metric values for real and quaternion-

valued models. Model memory size, and the number of trainable parameters are both examined in the preliminary results alongside differing model sizes (Section 4.1). All model architectures - excluding experimental ones - were incorporated within the analysis provided in the preliminary results. Analysis of the effects of quaternion-valued layers on training time is conducted briefly in the preliminary results (Section 4.1) and more extensively in the Discussion section (Section 5), completing the set of requirements for Objective 7.

6.2 Review of Planning and Processes

The largest setback encountered over the course of this project came as a result of an overly ambitious initial objective of creating a quaternion-valued adaptation the state of the art architecture at the time of planning [Défossez, 2021]. The original objective was then replaced with the objectives of defining a very basic real-valued U-Net architecture to then be adapted to use quaternion-valued layers. Once training had been successfully carried out on the initial single-layer models, the complexity of the general U-Net architecture was increased. This process of increasing model complexity was then repeated until training had been executed on sufficiently complex models.

The training procedure demonstrated issues relating to early stopping. Due to erratic validation loss values, training was terminated too early during particular experiments. These issues suggest that the validation set was undersized, although a new train/val split was not executed due to limitations in time for experimentation and the availability of hardware for model training. Initial issues with early stopping were tackled via an increase in the minimum number of epochs as well as the patience used for early stopping.

Model overfitting was a common problem with large levels of overfitting displayed by most of the trained models. Attempts to reduce overfitting could have been implemented via a lowering of the maximum number of training epochs or the introduction of regularisation. More advanced methods of reducing overfitting includes the creation of more training data through data augmentation - a method employed in a number of previous studies. Techniques include multiplying the stems by different factors before combining them to form a new mixture, through the use of addition [Stoller et al., 2018 p. 4]. Other examples include pitch-shifting, time-stretching [Cohen-Hadria et al., 2019 p. 3], adding noise and dynamic range compression [Salamon and Bello, 2017 p. 3]. These techniques can be used separately, or even in conjunction, in order to offer a solution to overfitting caused by data scarcity.

6.3 Potential Extensions

The most appropriate primary focus of extensions to this project would be further formation and exploration of different quaternion input structures. Such extensions would have the capability

to ascertain whether the underperformance of quaternion-valued models in comparison to real-valued models is due to a poor choice of quaternion input data structure. Possible 3D quaternion vector configurations for stereophonic music audio could include the use of a spectrogram's real components for both audio channels whilst using the imaginary values obtained from a monophonic version of the audio data. Further investigation into quaternion input structures for music audio data could entail an in-depth exploration of the potential to obtain quaternion structures from monophonic audio data.

More obvious extensions include the development of a more sophisticated general U-Net structure from which both real and quaternion-valued models are obtained. This could be done via the incorporation of commonplace neural network layers not employed in this project. Other methods could include altering the kernel sizes, strides and padding of different convolutional layers.

In previous studies in the field of music source separation, specialised weight initialisation methods are employed in order to improve training and, in the most successful cases, remove the need for batch normalisation [Défossez et al., 2021 p. 7]. During experimentation for this project, the volatility of the validation loss in relation to weight initialisation is demonstrated through the training graphs of identical models (Figures 2 and 4). This suggests that further exploration into quaternion weight initialisation would produce potential useful insights and the further development of methods utilised in the field.

Papers encountered during research also experimented with novel loss functions [Choi et al., 2019]. One example includes a loss function proposed in order to incorporate phase estimation into early U-Net models for music source separation[Staines et al., 2019 p. 5]. The formulation of loss functions, that take into consideration the mathematical laws of both real and imaginary components, could allow models to be trained using CaC spectrogram data input while the loss is calculated between output and target spectrograms as opposed to audio.

Most recent high-performance models, in the context task of source separation, are designed to acquire the isolated audio for multiple sources at a time. Extending the models architectures used in this project to the task of multi-source separation is then a natural development of the ideas implemented in this project. Code was written for multi-source separation data set classes, termed '2src' for models constructed to isolate vocals and accompaniment and '4src' for models tasked with separating vocals, bass, drums and other. This code, however, remains unused due to constraints in experimentation time.

6.4 Personal Conclusions

Being the first large-scale data science project I have worked on, most of the lessons picked up related to the planning of such projects. Lack of a mitigation plan, in the event of the initial objectives being unachievable, led to over-committing to the initial objectives, wasting a significant amount of time. The approach to determining whether these initial objectives

were achievable was also flawed. Due to the repeated complications surrounding the utilisation of packages with little to no documentation online - such as Facebook's Hydra package used in the Hybrid Demucs code - approaches to similar situations, in the future, will begin with establishing what packages are included in the code and whether these packages have sufficient documentation.

References

[Cheong Took and Mandic, 2011] Cheong Took, C. and Mandic, D. P. (2011). Augmented second-order statistics of quaternion random signals. *Signal Processing*, 91(2):214–224.

[Choi et al., 2019] Choi, H.-S., Kim, J.-H., Huh, J., Kim, A., Ha, J.-W., and Lee, K. (2019). PHASE-AWARE SPEECH ENHANCEMENT WITH DEEP COMPLEX U-NET. page 20.

[Choi et al., 2020] Choi, W., Kim, M., Chung, J., Lee, D., and Jung, S. (2020). Investigating U-Nets with various Intermediate Blocks for Spectrogram-based Singing Voice Separation. Technical Report arXiv:1912.02591, arXiv. arXiv:1912.02591 [cs, eess, stat] type: article.

[Cohen-Hadria et al., 2019] Cohen-Hadria, A., Roebel, A., and Peeters, G. (2019). Improving singing voice separation using Deep U-Net and Wave-U-Net with data augmentation. Technical Report arXiv:1903.01415, arXiv. arXiv:1903.01415 [cs, eess] type: article.

[Défossez, 2021] Défossez, A. (2021). Hybrid Spectrogram and Waveform Source Separation. Technical Report arXiv:2111.03600, arXiv. arXiv:2111.03600 [cs, eess, stat] version: 1 type: article.

[Défossez et al., 2021] Défossez, A., Usunier, N., Bottou, L., and Bach, F. (2021). Music Source Separation in the Waveform Domain. Technical Report arXiv:1911.13254, arXiv. arXiv:1911.13254 [cs, eess, stat] version: 2 type: article.

[Fu et al., 2017] Fu, S.-W., Hu, T.-y., Tsao, Y., and Lu, X. (2017). Complex spectrogram enhancement by convolutional neural network with multi-metrics learning. In *2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, Tokyo. IEEE.

[Gaudet and Maida, 2017] Gaudet, C. and Maida, A. (2017). Deep Quaternion Networks.

[Grassucci et al., 2021] Grassucci, E., Cicero, E., and Comminiello, D. (2021). Quaternion Generative Adversarial Networks. arXiv:2104.09630 [cs, eess].

[Guizzo et al., 2022] Guizzo, E., Weyde, T., Scardapane, S., and Comminiello, D. (2022). Learning Speech Emotion Representations in the Quaternion Domain. Number: arXiv:2204.02385 arXiv:2204.02385 [cs, eess].

[Jansson et al., 2017] Jansson, A., Humphrey, E., Montecchio, N., Bittner, R., Kumar, A., and Weyde, T. (2017). SINGING VOICE SEPARATION WITH DEEP U-NET CONVOLUTIONAL NETWORKS. page 7.

[Kadandale et al., 2020] Kadandale, V. S., Montesinos, J. F., Haro, G., and Gómez, E. (2020). Multi-channel U-Net for Music Source Separation. arXiv:2003.10414 [cs].

[Kingma and Ba, 2017] Kingma, D. P. and Ba, J. (2017). Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs].

[Krizhevsky et al., 2012] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

[Liu et al., 2021] Liu, H., Kong, Q., and Liu, J. (2021). CWS-PResUNet: Music Source Separation with Channel-wise Subband Phase-aware ResUNet. Technical Report arXiv:2112.04685, arXiv. arXiv:2112.04685 [cs, eess] type: article.

[Liu et al., 2020] Liu, H., Xie, L., Wu, J., and Yang, G. (2020). Channel-wise Subband Input for Better Voice and Accompaniment Separation on High Resolution Music. Technical Report arXiv:2008.05216, arXiv. arXiv:2008.05216 [cs, eess] type: article.

[Luo and Mesgarani, 2019] Luo, Y. and Mesgarani, N. (2019). Conv-TasNet: Surpassing Ideal Time-Frequency Magnitude Masking for Speech Separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(8):1256–1266. arXiv:1809.07454 [cs, eess].

[Mitsufuji et al., 2022] Mitsufuji, Y., Fabbro, G., Uhlich, S., Stöter, F.-R., Défossez, A., Kim, M., Choi, W., Yu, C.-Y., and Cheuk, K.-W. (2022). Music Demixing Challenge 2021. arXiv:2108.13559 [cs, eess].

[Parcollet et al., 2018a] Parcollet, T., Ravanelli, M., Morchid, M., Linarès, G., and De Mori, R. (2018a). Speech recognition with quaternion neural networks.

[Parcollet et al., 2018b] Parcollet, T., Zhang, Y., Morchid, M., Trabelsi, C., Linarès, G., De Mori, R., and Bengio, Y. (2018b). Quaternion Convolutional Neural Networks for End-to-End Automatic Speech Recognition. Technical Report arXiv:1806.07789, arXiv. arXiv:1806.07789 [cs, eess, stat] type: article.

[Perez-Lapillo et al., 2019] Perez-Lapillo, J., Galkin, O., and Weyde, T. (2019). Improving singing voice separation with the Wave-U-Net using Minimum Hyperspherical Energy. Technical Report arXiv:1910.10071, arXiv. arXiv:1910.10071 [cs, eess, stat] type: article.

[Qiu et al., 2020] Qiu, X., Parcollet, T., Ravanelli, M., Lane, N. D., and Morchid, M. (2020). Quaternion Neural Networks for Multi-Channel Distant Speech Recognition. In *Interspeech 2020*, pages 329–333. ISCA.

[Rafii et al., 2017] Rafii, Z., Liutkus, A., Stöter, F.-R., Mimilakis, S. I., and Bittner, R. (2017). MUSDB18 - a corpus for music separation. Version Number: 1.0.0 Type: dataset.

[Rafii et al., 2019] Rafii, Z., Liutkus, A., Stöter, F.-R., Mimilakis, S. I., and Bittner, R. (2019). MUSDB18-HQ - an uncompressed version of MUSDB18. Type: dataset.

[Ronneberger et al., 2015] Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. Technical Report arXiv:1505.04597, arXiv. arXiv:1505.04597 [cs] type: article.

[Rouard et al., 2022] Rouard, S., Massa, F., and Défossez, A. (2022). Hybrid Transformers for Music Source Separation. arXiv:2211.08553 [cs, eess] version: 1.

[Salamon and Bello, 2017] Salamon, J. and Bello, J. P. (2017). Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification. *IEEE Signal Processing Letters*, 24(3):279–283. arXiv:1608.04363 [cs].

[Staines et al., 2019] Staines, T., Weyde, T., and Galkin, O. (2019). Monaural speech separation with deep learning using phase modelling and capsule networks. *2019 27th European Signal Processing Conference (EUSIPCO)*, 2019-S. ISBN: 9789082797039.

[Stoller et al., 2018] Stoller, D., Ewert, S., and Dixon, S. (2018). Wave-U-Net: A Multi-Scale Neural Network for End-to-End Audio Source Separation. Technical Report arXiv:1806.03185, arXiv. arXiv:1806.03185 [cs, eess, stat] type: article.

[Stöter et al., 2018] Stöter, F.-R., Liutkus, A., and Ito, N. (2018). The 2018 Signal Separation Evaluation Campaign. arXiv:1804.06267 [cs, eess].

[Stöter et al., 2019] Stöter, F.-R., Uhlich, S., Liutkus, A., and Mitsufuji, Y. (2019). Open-Unmix - A Reference Implementation for Music Source Separation. *Journal of Open Source Software*, 4(41):1667.

[Takahashi and Mitsufuji, 2021] Takahashi, N. and Mitsufuji, Y. (2021). D3Net: Densely connected multidilated DenseNet for music source separation. Technical Report arXiv:2010.01733, arXiv. arXiv:2010.01733 [cs, eess] version: 4 type: article.

[Trabelsi et al., 2018] Trabelsi, C., Bilaniuk, O., Zhang, Y., Serdyuk, D., Subramanian, S., Santos, J. F., Mehri, S., Rostamzadeh, N., Bengio, Y., and Pal, C. J. (2018). Deep Complex Networks. arXiv:1705.09792 [cs].

[Vincent et al., 2006] Vincent, E., Gribonval, R., and Fevotte, C. (2006). Performance measurement in blind audio source separation. *IEEE Transactions on Audio, Speech and Language Processing*, 14(4):1462–1469.

[Zhu et al., 2019] Zhu, X., Xu, Y., Xu, H., and Chen, C. (2019). Quaternion Convolutional Neural Networks. Number: arXiv:1903.00658 arXiv:1903.00658 [cs].

A Vocal Separation Results Tables

A.1 32-Channel

	Mean	STD	Median	MAD	Max	Min
SDR	6.19119	3.759793	5.770601	2.195375	21.85032	-4.49228
SAR	6.113813	6.022945	6.757933	2.915052	22.74002	-21.9677

Table 60: U-Net validation set results: L1 loss, lr=0.001

	Mean	STD	Median	MAD	Max	Min
SDR	4.847673	3.73603	4.757054	2.128856	14.46027	-13.1162
SAR	4.334817	6.621251	5.194556	2.805971	17.12395	-38.1964

Table 61: Quaternion U-Net validation set results: L1 loss, lr=0.001

	Mean	STD	Median	MAD	Max	Min
SDR	4.822133	3.64792	4.829934	1.945237	15.54083	-8.92318
SAR	4.631993	5.953938	5.368241	2.563938	16.68324	-20.4578

Table 62: Quaternion U-Net validation set results: MSE loss, lr=0.001

	Mean	STD	Median	MAD	Max	Min
SDR	1.513435	3.131302	2.298227	0.9644	5.30182	-19.8598
SAR	-1.33992	5.364302	-0.64455	2.751186	14.09963	-24.6223

Table 63: Quaternion U-Net validation set results: MSE loss, lr=0.0001

A.2 64-Channel

	Mean	STD	Median	MAD	Max	Min
SDR	7.348379	4.069739	7.099396	2.453199	23.04883	-8.72985
SAR	7.710783	5.781062	8.000015	2.737189	23.74459	-40.1242

Table 64: U-Net validation set results: L1 loss, lr=0.001

	Mean	STD	Median	MAD	Max	Min
SDR	5.617242	3.616579	5.370573	2.124341	18.98443	-4.78683
SAR	5.621715	6.04678	6.372746	2.927007	19.99186	-20.8777

Table 65: Quaternion U-Net validation set results: L1 loss, lr=0.001

	Mean	STD	Median	MAD	Max	Min
SDR	1.873303	3.922892	2.526586	1.538662	9.324635	-20.9058
SAR	-0.47547	5.426164	0.366978	2.783046	11.85639	-19.9629

Table 66: Quaternion U-Net validation set results: MSE loss, lr=0.001, First Version

	Mean	STD	Median	MAD	Max	Min
SDR	6.418749	3.84413	6.340735	2.09953	20.58893	-9.47972
SAR	6.486014	5.506334	7.035128	2.611432	21.51019	-17.944

Table 67: Quaternion U-Net validation set results: MSE loss, lr=0.001, Second Version

	Mean	STD	Median	MAD	Max	Min
SDR	2.062853	3.677087	2.624072	1.520993	12.62446	-26.8041
SAR	-0.19517	5.069244	0.468242	2.866032	13.94753	-27.2194

Table 68: Quaternion U-Net train set results: MSE loss, lr=0.001, First Version

A.3 256-Channel

	Mean	STD	Median	MAD	Max	Min
SDR	7.795813	4.042357	7.34913	2.619782	24.04594	-5.37899
SAR	8.213988	5.336794	8.303707	2.981466	24.53576	-18.7496

Table 69: U-Net validation set results: L1 loss, lr=0.001

	Mean	STD	Median	MAD	Max	Min
SDR	2.368563	4.032882	3.063784	1.544645	10.80498	-19.8979
SAR	0.392915	5.65122	1.565794	2.637217	12.95696	-20.2286

Table 70: Quaternion U-Net validation set results: MSE loss, lr=0.001

	Mean	STD	Median	MAD	Max	Min
SDR	5.04753	3.530189	4.763783	2.057137	18.94548	-8.63355
SAR	5.092231	6.68226	5.993171	2.716378	20.13384	-24.9088

Table 71: Quaternion U-Net validation set results: L1 loss, lr=0.001