



Brain MRI Segmentation Using Autoencoders

Kishan Jackpersad¹ and Mandlenkosi Gwetu² 

¹ University of KwaZulu-Natal, Private Bag X54001, Durban 4000, South Africa

² University of KwaZulu-Natal, Private Bag X01, Scottsville 3209, South Africa
gwetum@ukzn.ac.za

Abstract. Brain MRI segmentation is a popular area of research that has the potential to improve the efficiency and effectiveness of brain related diagnoses. In the past, experts in this field were required to manually segment brain MRIs. This grew to be a tedious, time consuming task that was prone to human error. Through technological advancements such as improved computational power and availability of libraries to manipulate MRI formats, automated segmentation became possible. This study investigates the effectiveness of a deep learning architecture called an autoencoder in the context of automated brain MRI segmentation. Focus is centred on two types of autoencoders: convolutional autoencoders and denoising autoencoders. The models are trained on unfiltered, min, max, average and gaussian filtered MRI scans to investigate the effect of these filtering schemes on segmentation. In addition, the MRI scans are passed in either as whole images or image patches, to determine the quantity of contextual image data that is necessary for effective segmentation. Ultimately the image patches obtained the best results when exposed to the convolutional autoencoder and gaussian filtered brain MRI scans, with a dice similarity coefficient of 64.18%. This finding demonstrates the importance of contextual information during MRI segmentation by deep learning and paves the way for the use of lightweight autoencoders with less computational overhead and the potential for parallel execution.

Keywords: Brain MRI · Segmentation · Autoencoders · Filters · Image patches

1 Introduction

Brain imaging techniques have come a long way from the time they have been discovered. From Electroencephalography (EEG) to Magnetic Resonance Imagery (MRI), each of these has shown advancement in the way we obtain information about the human brain. This allows us to analyze and process data more efficiently to get a better understanding of the problem at hand. Efficient and

accurate segmentation of regions of interest within the brain aid neurologists in identifying brain diseases and disorders.

Manual segmentation has become a thing of the past. It is tedious and time-consuming. Hence automated segmentation has been introduced to enhance this process. This will improve diagnosis, and appropriate testing can take place within shorter spaces of time.

Brain MRI had taken the world by storm and is now the safest and most suggested brain imaging technique. It does not use radiation and can detect tumors or brain fractures that would not be picked up by an X-ray scan. Images from MRI scans are more detailed, and one can observe a greater range of soft-tissue structures. Segmentation of brain MRI scans are used to determine structural changes within the anatomy of the brain. These structural changes are used to detect brain diseases and disorders such as Alzheimer's. Automated segmentation methods have been widely explored and are still gaining popularity as new approaches in Deep Learning are allowing for advancement and more accurate segmentation.

Deep Learning methods have overcome the shortcomings of traditional Machine Learning techniques. They can generalize better by extracting a hierarchy of features from complex data due to their self-learning ability. The application of filters in image processing has also proved to play a significant role in obtaining better results, and hence, the effect of filtered slices for segmentation will be observed. These filtered slices could improve image data by making certain anatomical regions more visible and increase segmentation accuracy, or pixel values could be altered to a point where they affect the structure of brain anatomy and thus result in inaccurate or ineffective brain MRI segmentation.

The primary research question that this study will answer is: "Do filtered slices improve the effectiveness of brain MRI segmentation based on whole images and image patches?". Experiments will be conducted to investigate the effect that filtered slices have on segmentation, and further more, it will be determined if whole images or image patches improve the effectiveness of brain MRI segmentation.

This paper is structured in the following manner: Sect. 2 consists of the Literature Review which entails research on how Deep Learning models are being used for brain MRI segmentation. Section 3 is the Methodology that explains the implementation of this study. Section 4 is the Results and Discussion where the outcomes of this study are presented. Lastly, Sect. 5 is the Conclusion which summarizes the research findings and outlines future work.

2 Literature Review

Previously, Machine Learning algorithms were the go-to for many problems. These algorithms were applied to various scenarios and they asserted their dominance by proving to be very efficient. With the introduction of Deep Learning, things were taken a step further. Their application showed that there were new possibilities to obtain better results. When applied to brain MRIs, Deep Learning had opened numerous doors by achieving improved results in segmentation and classification tasks.

2.1 Convolutional Neural Networks

In terms of image processing, Convolutional Neural Networks (CNNs) work very well, as they are able to learn a hierarchy of complex features that can be used for image recognition, classification and are now being applied for image segmentation.

A journal article by Dolz et al. [9] reported the use of an ensemble of CNNs to segment infant T1 and T2-weighted brain MRIs. They found that segmentation of infant brain MRIs would pose a challenge as these scans have close contrast between white matter (WM) and grey matter (GM) tissue. The MICCAI iSEG-2017 Challenges public dataset was used to train the networks. Their main aim was to show how effective semi-dense 3D fully CNN, mixed with an ensemble learning strategy, could be and at the same time provide more robustness in infant brain MRI segmentation. Ensemble learning is the use of multiple models that are trained using different instances to improve performance accuracy and reduce variance for a problem. A set of models are used for prediction, and the outputs from these models are combined into a single prediction, usually generated by taking the most popular prediction of all the models. The basis behind using ensemble learning was to reduce random errors and increase generalization. The models used in ensemble learning varied by training data, where subsets from the overall training dataset were randomly chosen to train 10 CNNs. Majority voting was then used to choose the final prediction. For the activation function within the hidden layers, the authors used Parametric Rectified Linear Unit (PReLU), which increased model accuracy at a minor additional computation cost. Upon doing research, the authors found that using multiple input sources benefited the network and altered their model to allow for multi-sequence images (T1-weighted, T2-weighted, and Fractional Anisotropy - A type of brain MRI scan) as input. The CNN architecture composed of 13 layers made up of convolutional layers, fully-connected layers, and a classification layer. Batch normalization and a PReLU activation function is used before the convolutional filters. Input images were sent in as patches to reduce training time and increase the number of training examples. The average Dice Similarity Coefficient (DSC) for WM was 90%, GM was 92%, and cerebrospinal fluid (CSF) was 96%. The proposed architecture achieved excellent results as their metrics were able to rank first or second in most cases against twenty-one teams in the MICCAI iSEG-2017 Challenge.

A patch-based CNN approach was implemented by Yang et al. [8] which outperformed the two Artificial Neural Networks (ANNs) and three CNNs it was compared against. The proposed patchbased CNN contained seven main layers, made up of convolutional layers, two max-pooling layers, and a fully-connected layer. The architectures that it was compared against had various structures and differed by the number of feature maps and the input patch size. The proposed CNN employed a larger number of feature maps, which resulted in it having a higher performance than the other networks. The use of a larger patch size (32×32 compared to a patch size of 13×13) proved advantageous as the smaller patch size carried less information and made it harder for label

learning. The proposed architecture achieved an average DSC of 95.19% for the segmentation of cerebral white matter, the lateral ventricle, and the thalamus proper. The dataset used in this article is the same as the one used in this study; the Schizbull2008 dataset from CANDI neuroimaging. The only pre-processing done in the article consisted of what was implemented in the original dataset, and the authors further implemented the generation of patches. This article shows that the use of patches used less memory and thus reducing the models' training time. The use of patches also mends the problem of having a limited dataset.

Label propagation entails assigning labels to unlabelled data points by propagating through the dataset. Liang et al. [19] implemented this method using a Deep Convolutional Neural Network (DCNN) to classify voxels into their respective labels. The model was trained to reduce the DSC loss between the final segmentation map and a one-hot encoded ground truth. Input images were fed into the network as mini-batches of $128 \times 128 \times 128$ patches. A combination of the ISBR and MICCAI 2012 dataset was used, and in total, 28 images were used as training, and 25 images were used for testing. Based on the discussion of the results, the implemented method obtained a DSC score of 84.5%.

2.2 U-Net Models

Lee et al. [18] implemented a U-net model that follows a patch-wise segmentation approach to segment brain MRIs into WM, GM, and CSF. A U-net architecture is similar to an autoencoder in the sense that it has an encoder and decoder layer and is an extension of a Fully Convolutional Network (FCN). A U-net architecture has a decoder, which is similar to the encoder to give it that U-shape. The authors used a combination of two datasets in their implementation: An Open Access Series of Imaging Studies (OASIS) and the International Segmentation Brain Repository (ISBR). They compared a patch-wise U-net architecture to a conventional U-net and a SegNet model. The results show that the proposed architecture achieved greater segmentation accuracy in terms of DSC and Jaccard Index (JI) values. An average DSC score of 93 was achieved, which beats the conventional U-net by 3% and the SegNet by 10%. Stochastic Gradient Descent (SGD) was used as the optimizer, and categorical cross-entropy was used as the loss function. An investigation of how patch sizes affect the DSC was conducted, and it was found that smaller patch sizes (32×32 was better than 128×128 and 64×64) resulted in a better DSC.

Zhao et al. [32] implemented convolutional layers in a U-net architecture to segment brain MRIs. Their implementation of a learning-based method for data augmentation and applied it to oneshot medical image segmentation. The segmentation labels were obtained from FreeSurfer, and the dataset was put together from 8 different databases. The brain images were of size $256 \times 256 \times 256$ and were then cropped to $160 \times 192 \times 224$. No intensity corrections were applied, and only skull stripping was applied as pre-processing. The model was trained to segment the brain images into 16 different labels, including WM, GM, and CSF. The implementation was evaluated against other learning-based methods

and outperformed them. A DSC of 81.5% was achieved compared to the second-highest of 79.5% achieved from a model that used random augmentation.

2.3 Autoencoders

Atlason et al. [3] implemented a Convolutional Autoencoder to segment brain MRIs into brain lesions, WM, GM, and CSF in an unsupervised manner. The Convolutional Autoencoder architecture was created using fully convolutional layers. Noise was added to the inputs to ensure the model learns important features. The dataset used was obtained from the AGES-Reykjavik study. The Convolutional Autoencoder predictions were compared against a supervised method, a FreeSurfer segmentation, a patch-based Subject Specific Sparse Dictionary Learning (S3DL) segmentation, and a manual segmentation. The Convolutional Autoencoder performed the best against all these methods. The segmentation results for WM, GM, and CSF were not reported, although lesion segmentation achieved a DSC of 76.6%. The authors found that, at times, their model over-segmented due to image artifacts.

In 2018, a data scientist, Jigar Bandaria [5], implemented a Stacked Denoising Autoencoder (SDAE) to segment brain MRIs into WM, GM, and CSF. An Area Under the Curve evaluation of 83.16% was achieved on the OASIS dataset.

2.4 Review Summary

Brain MRI segmentation has come a long way and various Deep Learning methods (not limited to the ones mentioned above) have been applied to this problem in attempt to better previous state-of-the-art results. These attempts do indeed beat previous results, be it in performance or metric. However, many of these attempts use different datasets (OASIS, ISBR, MICCAI, etc.) that have different properties, labels, etc. This does not create a level playing field and poses a challenge to actually determine which attempt is relatively better than the other. Certain attempts also focus on the entire brain while others focus on specific parts of the brain (WM, GM, CSF, tumors, etc.). However, they all appear to use DSC or JI as a metric to determine the accuracy of the model. Convolutional layers and batch normalization play an important role in obtaining good results and determining the right activation function for the architecture is key. Another common thing to note is that majority of the attempts that used image patches (size 32×32) showed to require less computational resources and at the same time, obtain as good or even better results. Even though the autoencoder implementations to segment brain MRIs did not achieve as high accuracy's as the other mentioned models as they can be lossy due to compression and decompression, the U-net model that uses patch-wise segmentation achieved results that are close to conventional CNNs.

3 Methodology

Autoencoders have been around for many years, but in the past, they were mainly used for dimensionality reduction and information retrieval as they did

not produce better results than Principal Component Analysis (PCA). Since it was found that autoencoders can learn linear and nonlinear transformations, they proved to provide a more robust architecture [13] which makes them better at generalizing to problems.

Autoencoders are feed-forward neural networks that use self supervised or un-supervised learning to determine a hierarchy of important features within data. The aim is to learn an efficient encoding (representation) for a dataset and generate an output using that encoding. Using back-propagation to adjust weights within the network, an autoencoder will minimize reconstruction loss based on an output and a target. For the reconstruction of original data, the input becomes the target; otherwise, in terms of segmentation, the ground truth is used as the target.

There are three main components of an autoencoder: the encoder, the bottleneck (encoding), and the decoder. The encoders purpose is to reduce the dimensions of the data and train the network to learn meaningful representations along the way. The input into the encoder is the image to be segmented and the output from the encoder is fed into the bottleneck. The bottleneck is where the lowest point of the autoencoder lies. At this point, the dimensions of the data are at their lowest and contain very little information. The encoding is then fed into the decoder to decompress and generate an output (the segmented image). The bigger the size of the bottleneck, the more features are stored, and the more the reconstructed image will look like the original input. One should ensure not to keep the hidden layers too large, as the network would just learn to copy the data from input to output. The main idea of an autoencoder is to compress and decompress data. For this reason, autoencoders are known as lossy. The generated outputs often appear degraded when compared to the original inputs. More often than not, this is helpful when the transfer of data is more important than integrity or accuracy, e.g., live streaming concerts, game-plays, etc. Autoencoders are data specific and work best on data that they have been trained against. For example, an autoencoder that has been trained on compressing dogs would not be able to do a good job of compressing and reconstructing people.

Autoencoders have been implemented to segment images as in [30] for scene segmentation and in [25] for indoor and road scene segmentation. This goes to show that it is possible for an autoencoder to take in an image, compress it and then decompress it to generate a segmentation of the input. The process for segmenting brain MRIs is depicted in Fig. 1. The brain MRI scan is fed into the encoder to be compressed and is then fed into the bottleneck. The bottleneck then feeds the compressed image into the decoder to be decompressed, and as this occurs, the segmented image is being generated. The model output (segmented image) is then taken along with the corresponding ground truth, and a loss value is calculated. The network aims to minimize this loss value as it trains.

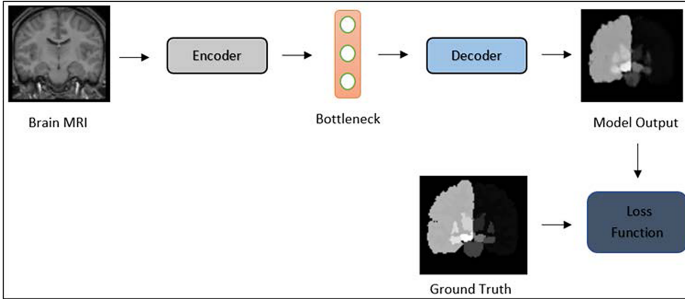


Fig. 1. Autoencoder image segmentation process.

3.1 Autoencoder Types

This study focuses on using two autoencoders for brain MRI segmentation: a Convolutional Autoencoder and a Denoising Autoencoder.

Convolutional Autoencoder. A Convolutional Autoencoder utilizes convolutional, pooling, and upsampling or transposed layers to compress and decompress data in an autoencoder architecture. The convolutional layers are used to extract features by applying filters to inputs and creating a feature map representing present information. This is where most of the network parameters are tweaked to obtain improved results. These parameters include the activation function, the number of kernels, the size of kernels, etc. Pooling layers perform the same task as convolutional layers by applying filters, although their primary purpose is to reduce data dimensionality within the network or, in other words, compress data [21]. Typical pooling layers consist of max-pooling (where the maximum pixel value within the neighborhood is taken) and average pooling (where the average of pixel values within a neighborhood is taken). Upsampling or transposed layers are then used for decompression. The final layer typically includes the use of a specific activation function based on the problem type. Convolutional Autoencoders are trained end-to-end to learn optimal filters to extract features that would help minimize the reconstruction error.

Convolutional Autoencoders have been used in areas such as segmentation of digital rock images [15], anomaly detection [28], and image classification paired with Particle Swarm Optimization (PSO) [27]. Figure 2 depicts the Convolutional Autoencoder architecture used in this study. The outcome of this model is the segmented brain MRI which needs to be as close to the ground truth as possible. The Convolutional Autoencoder contains convolutional layers (with LeakyRelu activation with $\alpha = 0.2$), batch normalization layers, maxpooling layers, upsampling layers, a layer normalization layer, and a final layer with a sigmoid activation.

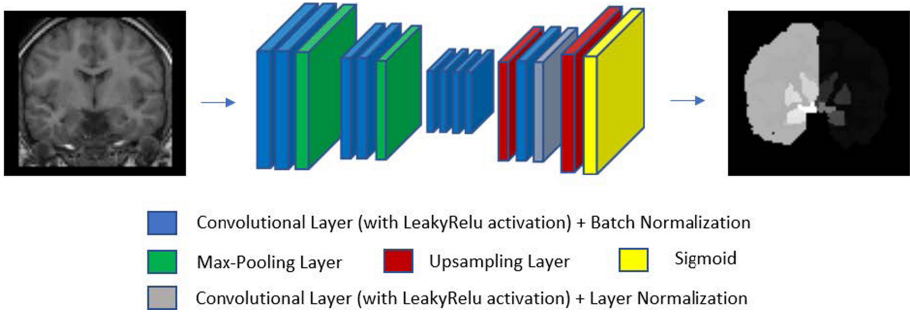


Fig. 2. Convolutional Autoencoder architecture.

Denoising Autoencoder. Denoising Autoencoders are standard autoencoders that have been trained to denoise data or reconstruct corrupted data. Denoising autoencoders help reduce the problem of having the autoencoder merely copy the data from input to output throughout the network. It ensures that the purpose of the autoencoder is not rendered useless by reducing the risk of the network learning the “Identity Function” or “Null Function.” Input data is stochastically corrupted on purpose and fed into the network. The loss function would then compare the output with the original, uncorrupted input. This would enhance the feature selection process and ensure useful features are extracted to help the network learn improved representations [29]. Typical noise that can be added to corrupt data include gaussian noise or salt-and-pepper noise.

Denoising autoencoders have been applied for speech enhancement [20], medical image denoising [12], fake twitter followers’ detection [6]. The Denoising Autoencoder architecture used in this study has the same layers as the Convolutional Autoencoder, although the input is much noisier. This should allow the Denoising Autoencoder to learn important features that would improve segmentation accuracy.

3.2 Dataset

The dataset used for this study is obtained from the CANDIShare website, which was developed by The Child and Adolescent NeuroDevelopment Initiative (CANDI) at the University of Massachusetts Medical School. This dataset is more commonly known as Schizbull 2008 [17], the dataset contains linear, non-linear, and processed brain MRI scans, along with their corresponding ground truth segmentations. The ground truth images were manually generated by Jean A, Frazier et al. [17].

There are 103 subjects in the dataset that are between the ages of 4 and 17 years. Both male and female subjects are included and are from one of four diagnostic groups: Bipolar Disorder with Psychosis, Bipolar Disorder without Psychosis, Schizophrenia Spectrum, and Healthy Control. Considering all the

ground truth images, there are 39 classes in total; however, the labels of these classes are unknown.

Each brain MRI scan and ground truth image has a size of $256 \times 256 \times 128$ (height, width, and the number of slices) and only contain a single channel. The brain MRI scans come in the NIFTI (.nii) format, which is one of the standard formats for multi-dimensional data in medical imaging. The Schizbull 2008 data had undergone bias field correction, and the images have been put into a standard orientation (image registration) by the authors [17] as pre-processing.

3.3 Pre-processing

Pre-processing refers to a technique that takes raw data and cleans or organizes it to allow the models to learn better. It makes the data more interpretable it is now easier for the model to parse. After obtaining brain MRIs, it is essential to pre-process these images to allow for a more efficient and accurate segmentation. Typical preprocessing methods in brain MRI segmentation include removing non-brain tissue, image registration [17, 22], bias field correction [1, 11, 14, 24] intensity normalization, and noise reduction. This study focused on the normalization and noise filtering preprocessing methods.

Intensity Normalization refers to the process of reducing intensity values to a standard or reference scale, e.g. [0, 1]. Intensity variations are caused when different scanners or different parameters were used to obtain MRI scans for the same or different patient at various times. One of the popular methods of intensity normalization includes the computing of z-scores. Yang et al. [26] used a histogram-based normalization technique as a pre-processing method, and they found that this greatly improved performance in image analyses and they were able to generate higher quality brain templates. Another popular method is the min-max normalization.

Min-max normalization was used as a pre-processing method in this implementation. Min-max normalization transforms the smallest value to zero and the largest value to one. When data points are very close together, normalization aids by making these datapoints sparse, this helps the model learn better and reduces training time. The formula for min-max normalization is given below.

$$norm(Img(i, j)) = \frac{Img(i, j) - min}{max - min} \quad (1)$$

In this case, $Img(i, j)$ is the brain MRI image pixel we wish to normalize, min refers to the minimum value found in the dataset, and max refers to the maximum value within the dataset.

Batch Normalization (BN) is a fundamental normalization technique that has shown significant improvement in the performance of a CNN [31]. BN solves the problem of preventing gradients from exploding. Since changing one weight

affects successive layers (Internal Covariate Shift), BN reparametrizes the network to control the mean and magnitude of activations. This makes optimization easier. However, BN does have its kryptonite. Small batch sizes cause the estimates to be very noisy, which has a negative effect on training. Advantages of BN include: It accelerates the training of deep neural networks, implements regularization within the network, hence eliminating the need for dropout layers. It allows the use of much higher learning rates [2] and. Learning becomes more efficient with BN, and since it introduces regularization within the network, it reduces the chances of the model overfitting the data. BN works well with small training scales. As training scales increase, an evolution of BN called Synchronized Batch Normalization would be needed. BN was implemented after every convolution layer (as this is where the activation function lies) as [16] showed that BN performs and yields better results when placed after the activation function.

Layer Normalization (LN) works by normalizing across input features that are independent of each other, unlike Batch Normalization where normalization occurs across the batch. It is basically the transpose of Batch Normalization. LN helps speed up the training time of a network and address the drawbacks of Batch Normalization. [4] introduced the LN technique and found that this technique worked well with recurrent networks, but more research and tests would need to be conducted to make LN work well with Convolutional Networks.

3.4 Activation Function

Activation functions are a necessity in neural networks as nonlinear activation functions make back-propagation possible. It helps determine which features are important within the data, and they reduce the chances of the network learning irrelevant features. Basically put, activation functions classify data into useful and notso- useful. The activation function introduces non-linearity into the output of the neuron by determining whether or not a neuron should be activated. This is done by activation could better these results. This was done, and ultimately Leaky ReLU produced the best results. Leaky ReLU addresses the problems of ReLU as it rectifies the dying ReLU problem [10]. This allows the model to essentially continue learning and perform better. Unlike the traditional ReLU activation function where all values less than 0 are set to 0, Leaky ReLU sets these values to values that are slightly less than 0; hence the slight descend on the left of the graph. This is done by the following equation:

$$f(x) = \begin{cases} 0.01x, & \text{if } x < 0 \\ x, & \text{otherwise} \end{cases} \quad (2)$$

From the above equation, we can see that values less than zero are set much close to 0 (0.01x). If the values are not less than zero, they are not changed. This helps minimize the sensitivity to the dying ReLU problem.

The values in the dataset have been normalized to a range of 0 and 1. The network would need to predict a value within this range for each pixel. Hence, a Sigmoid activation function was used in the final layer of the network.

3.5 Loss Function

Mean Squared Error (MSE) is used as the loss function. This loss shows the squared difference between the ground truth and the output of the model. MSE is calculated as follows:

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad (3)$$

Where N is the number of pixels, y_i is the ground truth image, and \hat{y}_i is the output from the model. The result of MSE is always positive, and when the ground truth and model output are identical, a value of zero is obtained. The models loss starts at a high value and converges to much smaller values that are close to zero. When the model makes big mistakes, the squaring penalizes the model. Smaller mistakes are not as heavily penalized.

Other common loss functions for image segmentation that can be applied are Cross-Entropy (CE) loss as in [29]. The authors in [29] tried both CE and MSE and found that the network learned different features depending on which loss function was implemented. Dice Loss is another loss function that can be implemented as applied in [19]. It is possible to create a custom loss function based on a combination of loss functions. These will depend on what features you wish the model to learn.

4 Results and Discussion

4.1 Evaluation Metrics

During medical imaging, in real-time, the ground truth for patients is not available, and only the segmentation is generated given a brain MRI. Hence, it is important to determine a metric to validate the loss of the model or algorithm that is going to be used to generate these segmentations. The evaluation can be based on unseen data that has the corresponding ground truth. These metrics will tell us how well the model should perform in real-time.

Dice Similarity Coefficient. One of the most common and well-known metrics to evaluate image segmentation is the Dice Similarity Coefficient (DSC) [7]. The DSC tells us how similar two images are. It calculates area of overlap between two images and divides that by the total size. This metric is calculated as follows:

$$DSC = \frac{2|X \cap Y|}{|X| + |Y|} \quad (4)$$

X is the predicted segmentation from the model, and Y is the corresponding ground truth. $|X|$ denotes the number of pixels in the predicted segmentation, and $|Y|$ denotes the number of pixels the corresponding ground truth. $|X \cap Y|$ is known as the area of overlap. This metric calculates the number of identical pixels at a corresponding point within two images. The DSC is within the range of 0 and 1. A DSC of 1 is achieved when both X and Y are identical, and a DSC of 0 is achieved when there is no overlap between the two images. This would mean that values closer to 1 are desired.

Jaccard Index. Another standard metric for image segmentation evaluation is the Jaccard Index (JI), also known as the Tanimoto Coefficient [7]. This metric also calculates the similarity between two images, just like the DSC. Although, it penalizes bad classifications. This metric is calculated as follows:

$$JI = \frac{|X \cap Y|}{|X| + |Y| - |X \cap Y|} \quad (5)$$

X is the predicted segmentation from the model, and Y is the corresponding ground truth. $|X|$ denotes the number of pixels in the predicted segmentation, and $|Y|$ denotes the number of pixels the corresponding ground truth. $|X \cap Y|$ is known as the area of overlap. The JI calculates the intersection divided by the union of two images. This is metric is also within the range of 0 and 1. A value of 0 means that the two images are entirely different, and a value of 1 means that the two images are identical. This would mean that a Jaccard Index closer to 1 would be desired.

4.2 Implementation

In this study, the use of filters on two autoencoder models are investigated. These filters include a min filter, a max filter, an average filter, and a gaussian filter. The autoencoder architectures implemented are a Convolutional Autoencoder and a Denoising Autoencoder. These architectures will take in whole images and images patches to determine which of these work best. Tests were run on these two autoencoders with the previously mentioned filters and the original unfiltered dataset. Their results will be discussed in the next section.

After the brain MRI scans were loaded, the filters are applied. For the min, max and average filtered scans, a combined image was created by taking 3 consecutive slices and the respective min, max or average filter was applied. For gaussian filtered scans, a gaussian filter with a sigma value of one was used. This gave the MRI scan a slight blur. The unfiltered MRI scans remained as so. The brain MRI scans and their corresponding ground truth are then reduced from 256×256 to 128×128 . The reduction in size was done to decrease computational costs. Furthermore, the scans are cropped to 80×88 to reduce the large amount of background pixels.

The middle twenty slices from each brain MRI scan was used for training models on whole images. These slices were used as they contained less non-brain tissue. This gave us a total of 2060 scans, along with their corresponding

ground truth images. For the image patches, three slices from the middle were used and overlapping patches of 40×44 were generated from each of the three slices. In total 7725 patches were generated from 309 scans, thus 25 overlapping patches from each images was generated. A batch size of 32 was used for all models, and they were trained for 250 epochs. RMSprop was used as the optimizer with the default learning rate of 0.001. The loss function implemented was Mean Squared Error. Early stopping with a patience of 10 was implemented to reduce the chances of the model overfitting. A random seed of 42 was used for reproducibility.

The images were split into training, validation, and testing sets. The dataset was split as follows: 60% of the data was used for training, 20% was used for validation, and the remaining 20% was used for testing. For the Denoising Autoencoder, gaussian noise was added to images before training.

The min, max, and average filters were applied by taking three consecutive slices, and their min, max, or average was computed to obtain a single image. The gaussian filter (with a sigma value of one) was applied to all unfiltered training images, which gave them a slight blur. Figure 3 depicts the original image and alongside it is what the image looks like with the filters applied. From left to right, the images are: the unfiltered image, the min filtered image, the max filtered image, the average filtered image, and the gaussian filtered image. These images were used as input to train the Convolutional Autoencoder.

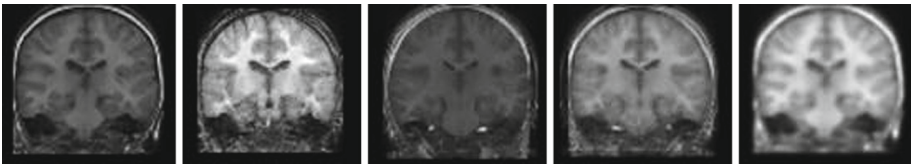


Fig. 3. Brain MRI scans with the applied filters.

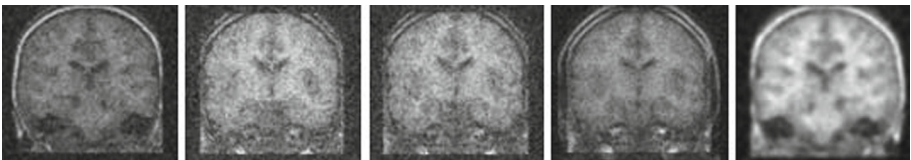


Fig. 4. Noisy brain MRI scans with the applied filters.

Figure 4 depicts the noisy images and alongside it is what the image looks like with the filters applied. From left to right, the images are: the unfiltered image, the min filtered image, the max filtered image, the average filtered image, and the

gaussian filtered image. These images were used as input to train the Denoising Autoencoder.

It is apparent that compared to the unfiltered images, the other images vary in contrast. Certain regions of the brain appear in higher contrast to others. The min filter gives the center tissues of the brain a higher contrast, whereas the max filter makes it appear darker. The average filter image is basically a combination of the min and max filter. With the gaussian filter applied, the unfiltered image is now blurry and the center region of the brain is more defined.

4.3 Results

The results obtained from experimentation of various filters for each autoencoder on whole images and image patches are depicted below. These results are obtained from using the trained models to predict on the unseen test data. The filter that allowed an autoencoder model to obtain the highest result, for each table, is in bold. Table 1 shows the results obtained when various filters were applied to the whole images before being trained on the Convolutional Autoencoder. Based on the Dice Similarity Coefficient and Jaccard Index, the images that had an average filter applied to it has obtained the best results.

Table 1. Test results for Convolutional Autoencoder on whole images.

	Convolutional Autoencoder				
	Unfiltered	Min	Max	Average	Gaussian
DSC	44.47%	45.42%	45.07%	45.86%	45.30%
JI	29.42%	29.38%	29.09%	29.75%	29.33%

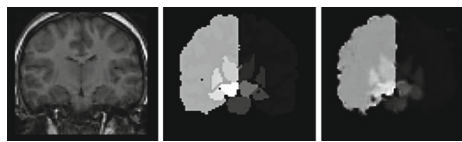


Fig. 5. Convolutional Autoencoder whole image segmentation.

Figure 5 depicts the segmentation of a brain MRI scan using the Convolutional Autoencoder. The first image shows the scan that is used as the input to be segmented. This is the average filtered image. The second image is the corresponding ground truth and this is what the predicted image from the Convolutional Autoencoder should look like. The third image is the predicted segmentation of the brain MRI scan. Table 2 shows the results obtained when various filters were applied to image data before being trained on the Denoising Autoencoder. The unfiltered whole images that were used as input obtained the best results.

Table 2. Test results for Denoising Autoencoder on whole images.

	Denoising Autoencoder				
	Unfiltered	Min	Max	Average	Gaussian
DSC	44.59%	44.27%	44.54%	44.28%	44.23%
JI	28.67%	28.43%	28.65%	28.43%	28.39%

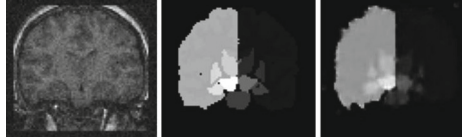
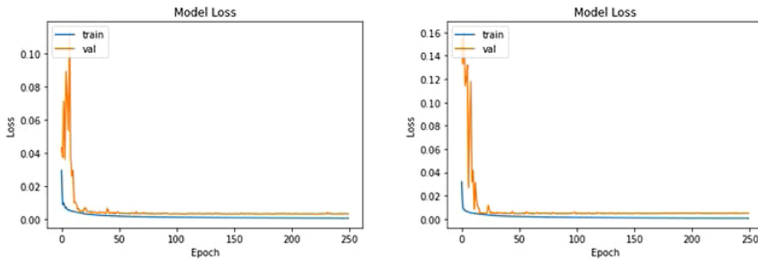


Fig. 6. Denoising Autoencoder whole image segmentation.

Figure 6 depicts the segmentation of a brain MRI scan using the Denoising Autoencoder. The first image shows the scan that is used as the input to be segmented. This is the unfiltered noisy image. The second image is the corresponding ground truth and this is what the predicted image from the Denoising Autoencoder should look like. The third image is the segmented brain MRI scan.



(a) Average Filtered Data : Convolutional Autoencoder.

(b) Unfiltered Data : Denoising Autoencoder.

Fig. 7. Loss graphs for the best performing models on whole images.

The graphs in Fig. 7 depict the loss curves for the models that were trained on average filtered whole images and unfiltered whole images respectively. These are best performing models for the Convolutional Autoencoder and Denoising Autoencoder. Table 3 shows the results obtained when various filters were applied to the input patches before being trained on the Convolutional Autoencoder. The gaussian filtered image patches that were used as input achieved the best results. Figure 8 depicts the segmentation of a brain MRI patch using the Convolutional Autoencoder. The first image shows the patch that is used as the input to be segmented. This is the gaussian filtered image patch. The second image is

Table 3. Test results for Convolutional Autoencoder on image patches.

	Convolutional Autoencoder				
	Unfiltered	Min	Max	Average	Gaussian
DSC	63.23%	63.84%	64.13%	63.44%	64.18%
JI	46.23%	46.88%	47.20%	46.45%	47.26%

**Fig. 8.** Convolutional Autoencoder image patch segmentation.

the corresponding ground truth and this is what the predicted patch from the Convolutional Autoencoder should look like. The third image is the predicted segmented patch from the model.

Table 4. Test results for Denoising Autoencoder on image patches.

	Denoising Autoencoder				
	Unfiltered	Min	Max	Average	Gaussian
DSC	62.49%	62.87%	62.80%	63.41%	63.25%
JI	45.45%	45.85%	45.77%	46.42%	46.25%

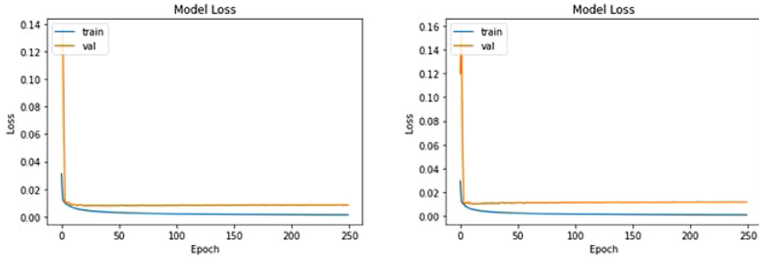
Table 4 shows the results obtained when various filters were applied to the input patches before being trained on the denoising autoencoder. The model that used average filtered image patches as input achieved best results. Figure 9 depicts the segmentation of a brain MRI patch using the Denoising Autoencoder. The first image is a noisy patch of an MRI scan that is used as the input to be segmented. This is an average filtered noisy image patch. The second image is the corresponding ground truth patch and this is what the prediction of the Denoising Autoencoder should look like. The third image is the predicted segmented patch from the model. The graphs in Fig. 10 depict the loss curves for the models that were trained on gaussian filtered image patches and average filtered image patches for the Convolutional Autoencoder and Denoising Autoencoder respectively.

4.4 Discussion

The results show that a simple autoencoder architecture can be used to segment brain MRIs. In two out of four cases (one being for the Convolutional Autoencoder on whole images and the other for the Denoising Autoencoder on image



Fig. 9. Denoising Autoencoder image patch segmentation.



(a) Gaussian Filtered Data : Convolutional Autoencoder. (b) Average Filtered Data : Denoising Autoencoder.

Fig. 10. Loss graphs for the best performing models on image patches.

patches), the average filtered data generated the best predictions and achieved the highest DSC and JI scores. The Convolutional Autoencoder achieved a DSC of 45.86% and JI of 29.75%, while the Denoising Autoencoder achieved a DSC of 63.41% and a JI of 46.42%. The model that achieved the best results on the Denoising Autoencoder using whole images had unfiltered MRI scans that were used as an input during training. It achieved a DSC of 44.59% and a JI of 28.67%. For the ConvolutionalAutoencoder that was trained on image patches, the gaussian filtered MRI scans achieved the best DSC and JI scores, this being 64.18% and 47.26% respectively. From all the experiments done, it is evident that the Convolutional Autoencoder which used gaussian filtered image patches as input achieved the best results.

The Convolutional Autoencoder achieved higher results than the Denoising Autoencoder when whole images and image patches were used for most filters applied. Although, the difference between their results is not very large. At most, there is a 2% difference. This shows that the autoencoder can work almost as well if given noisy data. The signal noise is reduced in the encoder as compression takes place. This will be advantages when noisy MRI scans are generated as the network is being trained to ignore this noise. This is main benefit of using a Denoising Autoencoder.

The use of filters has shown to increase segmentation accuracy in almost all models. This shows that their presence helps the network learn better. In three out of four cases, the min filter proved to be better than the max filter. When the filters were applied to image patches to train the model, they showed that

they all had a greater effect as they achieved better results than the model that was trained on unfiltered image patches.

The use of image patches allowed the models to perform much better in every case as compared to those models that were trained on whole images. This showed a large $\pm 20\%$ improvement in both DSC and JI scores for the Convolutional Autoencoder and the Denoising Autoencoder. The models were able to focus more on the local features of the training data, which contributed to them performing better. The smaller the patch size, the more focus is given to local features. However, due to the autoencoder architecture, one cannot make the patch size too small, as this would lead to a lot of information loss when the network compresses the data.

The lower the loss, the better a model is. The calculated loss is a representation of the errors made in the training and validation sets. Figure 7 depicts the loss graphs obtained from training the models on whole images. These models seem to be fitting the data very well as the training and validation loss curves are very close to each other. Both the curves have spikes in their early stages of training, but as training continues, the curves appear much smoother. Figure 10 depicts the loss graphs obtained from training the models on image patches. The graphs show that both models seem to be slightly overfitting as the training loss is decreasing while the validation loss seems to be increasing very slightly as each training epoch increases. If the overfitting begins to get worse, the early stopping will kick in.

Based on the graphs, the models seem to have learnt fairly well. At very early stages in training, the training and validation loss curves were closest to each other. In all four graphs, the validation loss either starts very high or goes very high for a small period of time and then decreases. The training curve starts off relatively low and continues decreasing as the model training occurs. Throughout training the loss curve is smooth and has no spikes.

Due to the large number of classes in the ground truth, the chances of a pixel being classified into one of the thirty-nine classes is slim; unlike other research experiments where pixels are classified into one of four classes (background, WM, GM, or CSF).

5 Conclusion

This study researched the application of autoencoders in image segmentation and whether or not filters improve its effectiveness. Based on the results and in response to the posed research question, it can be concluded that gaussian filtered images do improve segmentation quality. The use of whole images and image patches for training was also investigated. This investigation showed that image patches work much better than whole images. With a simple autoencoder architecture, we were able to generate the segmentation of a brain MRI scan with fairly good results. However, the lossy nature of the autoencoder may prove to be a bit of a problem. More experiments need to be conducted to try and reduce this lossy nature.

Future work will focus on applying other types of autoencoders such as Stacked Autoencoders that worked well for brain MRI segmentation as in [5]. Transfer learning is another promising area of research that can prove beneficial to autoencoders as demonstrated in [23].

References

1. N4 bias field correction. https://simpleitk.readthedocs.io/en/master/link-N4BiasFieldCorrection_docs.html. Accessed 05 Sept 2021
2. Adaloglou, N.: In-layer normalization techniques for training very deep neural networks (2020). <https://theaisummer.com/normalization/>. Accessed 05 Sept 2021
3. Atlason, H.E., Love, A., Sigurdsson, S., Gudnason, V., Ellingsen, L.M.: Unsupervised brain lesion segmentation from MRI using a convolutional autoencoder. In: Medical Imaging 2019: Image Processing, vol. 10949, p. 109491H. International Society for Optics and Photonics (2019)
4. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint [arXiv:1607.06450](https://arxiv.org/abs/1607.06450) (2016)
5. Bandaria, J.: Brain MRI image segmentation using stacked denoising autoencoders. <https://bit.ly/3dE0KFs> (2017). Accessed 05 Sept 2021
6. Castellini, J., Poggioni, V., Sorbi, G.: Fake twitter followers detection by denoising autoencoder. In: Proceedings of the International Conference on Web Intelligence, pp. 195–202 (2017)
7. Crum, W.R., Camara, O., Hill, D.L.: Generalized overlap measures for evaluation and validation in medical image analysis. *IEEE Trans. Med. Imaging* **25**(11), 1451–1461 (2006)
8. Cui, Z., Yang, J., Qiao, Y.: Brain MRI segmentation with patch-based CNN approach. In: 2016 35th Chinese Control Conference (CCC), pp. 7026–7031. IEEE (2016)
9. Dolz, J., Desrosiers, C., Wang, L., Yuan, J., Shen, D., Ayed, I.B.: Deep CNN ensembles and suggestive annotations for infant brain MRI segmentation. *Comput. Med. Imaging Graph.* **79**, 101660 (2020)
10. Dubey, A.K., Jain, V.: Comparative study of convolution neural network’s Relu and Leaky-Relu activation functions. In: Mishra, S., Sood, Y.R., Tomar, A. (eds.) Applications of Computing, Automation and Wireless Systems in Electrical Engineering. LNEE, vol. 553, pp. 873–880. Springer, Singapore (2019). https://doi.org/10.1007/978-981-13-6772-4_76
11. Fluck, O., Vetter, C., Wein, W., Kamen, A., Preim, B., Westermann, R.: A survey of medical image registration on graphics hardware. *Comput. Methods Programs Biomed.* **104**(3), e45–e57 (2011)
12. Gondara, L.: Medical image denoising using convolutional denoising autoencoders. In: 2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW), pp. 241–246. IEEE (2016)
13. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016)
14. Ivanovska, T., Wang, L., Laqua, R., Hegenscheid, K., Völzke, H., Liebscher, V.: A fast global variational bias field correction method for MR images. In: 2013 8th International Symposium on Image and Signal Processing and Analysis (ISPA), pp. 667–671. IEEE (2013)

15. Karimpouli, S., Tahmasebi, P.: Segmentation of digital rock images using deep convolutional autoencoder networks. *Comput. Geosci.* **126**, 142–150 (2019)
16. Kathuria, A.: Intro to optimization in deep learning: busting the myth about batch normalization (2018). <https://bit.ly/2KXTA63>. Accessed 05 Sept 2021
17. Kennedy, D.N., et al.: CANDIShare: a resource for pediatric neuroimaging data. *Neuroinformatics* **10**, 319–322 (2012)
18. Lee, B., Yamanakkanavar, N., Choi, J.Y.: Automatic segmentation of brain MRI using a novel patch-wise U-Net deep architecture. *PLoS ONE* **15**(8), e0236493 (2020)
19. Liang, Y., Song, W., Dym, J.P., Wang, K., He, L.: CompareNet: anatomical segmentation network with deep non-local label fusion. In: Shen, D., et al. (eds.) *MICCAI 2019. LNCS*, vol. 11766, pp. 292–300. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32248-9_33
20. Lu, X., Tsao, Y., Matsuda, S., Hori, C.: Speech enhancement based on deep denoising autoencoder. In: *Interspeech*, vol. 2013, pp. 436–440 (2013)
21. O’Shea, K., Nash, R.: An introduction to convolutional neural networks. arXiv preprint [arXiv:1511.08458](https://arxiv.org/abs/1511.08458) (2015)
22. Subramanian, P., Faizal Leerar, K., Hafiz Ahammed, K.P., Sarun, K., Mohammed, Z.: Image registration methods. *Int. J. Chem. Sci* **14**, 825–828 (2016)
23. Rane, R.: Efficient pretraining techniques for brain-MRI datasets (2019). <https://doi.org/10.13140/RG.2.2.11782.11843>
24. Song, J., Zhang, Z.: Brain tissue segmentation and bias field correction of MR image based on spatially coherent FCM with nonlocal constraints. *Comput. Math. Methods Med.* **2019** (2019)
25. Spolti, A., et al.: Application of u-net and auto-encoder to the road/non-road classification of aerial imagery in urban environments. In: *VISIGRAPP (4: VISAPP)*, pp. 607–614 (2020)
26. Sun, X., et al.: Histogram-based normalization technique on human brain magnetic resonance images from different acquisitions. *Biomed. Eng. Online* **14**(1), 1–17 (2015)
27. Sun, Y., Xue, B., Zhang, M., Yen, G.G.: A particle swarm optimization-based flexible convolutional autoencoder for image classification. *IEEE Trans. Neural Netw. Learn. Syst.* **30**(8), 2295–2309 (2018)
28. Tran, H.T., Hogg, D.: Anomaly detection using a convolutional winner-take-all autoencoder. In: *Proceedings of the British Machine Vision Conference 2017*. British Machine Vision Association (2017)
29. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.A., Bottou, L.: Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.* **11**(12) (2010)
30. Yusiong, J.P.T., Naval, P.C.: Multi-scale autoencoders in autoencoder for semantic image segmentation. In: Nguyen, N.T., Gaol, F.L., Hong, T.-P., Trawiński, B. (eds.) *ACIIDS 2019. LNCS (LNAI)*, vol. 11431, pp. 587–599. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-14799-0_51
31. Zhang, Q.: An overview of normalization methods in deep learning. <https://zhangtemplar.github.io/normalization/>. Accessed 05 Sept 2021
32. Zhao, A., Balakrishnan, G., Durand, F., Gutttag, J.V., Dalca, A.V.: Data augmentation using learned transformations for one-shot medical image segmentation. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8543–8553 (2019)