A DEGRADATION-ROBUST DEEP LEARNING FRAMEWORK FOR MRI COMPUTER-AIDED BRAIN TUMOR DIAGNOSIS

RICARDO BAUCHSPIESS

DISSERTAÇÃO DE MESTRADO
EM ENGENHARIA ELÉTRICA

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA

Universidade de Brasília
Faculdade de Tecnologia
Departamento de Engenharia Elétrica

A Degradation-Robust Deep Learning Framework for MRI Computer-Aided Brain Tumor Diagnosis

Ricardo Bauchspiess

DISSERTAÇÃO DE MESTRADO SUBMETIDA AO PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA DA UNIVERSIDADE DE BRASÍLIA COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE.

APROVADA POR:

Mylene Christine Queiroz de Farias, Dr. (UnB)
(Orientadora)

Daniel Guerreiro e Silva, Dr. (UnB)
(Presidente)

Cristiano Jacques Miosso Rodrigues Mendes, Dr. (UnB-Gama)
(Examinador Interno)

Fátima Nelsizeuma Sombra de Medeiros, Dr. (UFC)
(Examinadora Externa)

Francisco Assis de Oliveira Nascimento, Dr. (UnB)
(Suplente)

Brasília/DF, Outubro de 2024.

FICHA CATALOGRÁFICA

BAUCHSPIESS, RICARDO

A Degradation-Robust Deep Learning Framework for MRI Computer-Aided Brain Tumor Diagnosis. [Brasília/DF] 2024.

xxx, nnnp., 210 x 297 mm (ENE/FT/UnB, Mestre, Dissertação de Mestrado, 2024).

Universidade de Brasília, Faculdade de Tecnologia, Departamento de Engenharia Elétrica.

Departamento de Engenharia Elétrica

1. Keyword	2. Keyword
3. Keyword	4. Keyword
5. Keyword	6. Keyword
7. Keyword	8. Keyword
I. ENE/FT/UnB	II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

BAUCHSPIESS, RICARDO (2024). A Degradation-Robust Deep Learning Framework for MRI Computer-Aided Brain Tumor Diagnosis. Dissertação de Mestrado, Publicação PPGEE.DM 821/2024, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, xxxxp.

CESSÃO DE DIREITOS

Ricardo Bauchspiess

Universidade de Brasília (UnB)

Campus Darcy Ribeiro

Faculdade de Tecnologia - FT

Departamento de Engenharia Elétrica(ENE)

Brasília - DF CEP 70919-970

ACKNOWLEDGEMENTS

I'd like to first thank my advisor, Mylene Farias, whose constant guidance enabled the completion of this dissertation. I'm thankful to my parents, Adolfo and Enir, who raised me to be curious and value edutation. I'd also like to thank my siblings, Carolina, Daniel and Cristina, who gave me the initial push to enroll in my masters program.

ABSTRACT

Brain tumors are abnormal cell growths that affect the brain or central nervous system (CNS) and are estimated to be the cause of death for more than 200,000 individuals each year. However, the prognosis for patients with brain tumors can be improved by early diagnosis and treatment. The gold standard test for brain tumor diagnosis is the cranial Magnetic Resonance Imaging (MRI) and artificial intelligence (AI) methods have shown potential to improve radiologist efficiency and reduce human errors by predicting brain tumor diagnosis from Magnetic Resonance (MR) images. Like other imaging methods, MRI is susceptible to image distortions or degradations that impair the clinical utility of such images and negatively impact AI methods used to analyze brain MRI images.

To obtain degradation-robust brain diagnosis from 2D MRI images, we propose a two-stage diagnosis method. First, we use a U-shaped transformer model, called Uformer, to perform image restoration, removing image distortions, and then we use Convolutional Neural Network (CNN) models, namely EfficientNet, to obtain diagnosis predictions from the restored images. To train the restoration model, we propose an artifact generation function that trains the restoration model to restore images affected by an arbitrary number of degradation types.

We evaluated degraded images from various datasets, observing that images are often affected by more than one artifact type and that different artifacts affect each other, changing the characteristics of individual artifacts. We also observed that different artifact types have different degrees of impact in computer-aided diagnostic accuracy, with noise having the most significant impact and ghosting having the least impact. Our proposed artifact generation function takes into account these characteristics to generate realistic artificial artifacts.

Our experiments also show that image quality also has a negative impact during the training of deep learning models. Artifacts in the training images can lower the accuracy of models or cause overfitting that makes the models unreliable. We compared various image restoration and classification models, different methods to generate the training data, and multiple datasets to train the models. We tested our methods with artificial artifacts to evaluate the impact of

specific artifacts and intensities, but also tested the solution on five different datasets, including cross-validation between datasets, which included real cases of degraded images, showing the potential of our approach for real-world applications. The method is shown to improve image quality, computer-aided diagnostic accuracy, and model generalization.

Keywords: Brain tumor, MRI, CNN, image artifacts, image restoration, image diagnosis.

RESUMO

Título: Um Método de Aprendizado Profundo Robusto À Degradação para o Diagnóstico de Tumores Cerebrais Auxiliado por Computador em Imagens de Ressonância Magnética

Tumores cerebrais são crescimentos anormais de células do cérebro ou do sistema nervoso central, os quais são responsáveis por uma estimativa de mais de 200.000 mortes por ano no mundo todo. O prognóstico para pacientes com câncer cerebral pode, porém, ser melhorado com diagnóstico e tratamento precoce. O teste mais recomendado para o diagnóstico de tumores cerebrais é a ressonância magnética (RM) cranial e métodos de inteligência artificial (IA) demonstraram potentcial em aumentar a eficiência de radiologistas e reduzir o erro humano no diagnóstico de tumores do cérebro a partir de imagens de ressonância magnética. Assim como em outras modalidades de imagens, ressonâncias magnéticas podem ser afetadas por distorções ou degradações que pioram a utilidade clínica dessas imagens e prejudicam métodos automáticos de diagnóstico por imagem.

Para obter um diagnóstico de tumor cerebral a partir de imagens de ressonância magnética 2D de forma robusta contra degradações de imagens, nós propomos uma solução em duas etapas. Na primeira etapa nós utilizamos um modelo transformer em forma de U, chamado Uformer, para remover distorções e restaurar as imagens e em seguida utilizamos uma rede neural convolucional (RNC), particularmente a EfficientNet, para classificar a imagem restaurada e obter um diagnóstico quanto a tumores cerebrais. Para treinar o modelo de restauração de imagens, nós definimos uma função para gerar um número arbitrário de tipos de distorções em imagens, assim treinando o modelo para remover combinações arbitrárias de degradações em imagens de ressonância magnética do cérebro.

Nós realizamos uma avaliação de imagens de ressonância magnética contendo degradações, identificando os tipos mais comuns de artefatos de imagem em RM, sendo elencados os artefatos de ruído, borrão, artefato de Gibbs, mal contraste e artefatos de compressão JPEG. Também bservamos que imagens são tipicamente afetadas por mais de um tipo de artefato e que os próprios artefatos são por outros artefatos, alterando suas características individuais. A distri-

buição de ruído, por exemplo, pode ser afetada por mal contraste, borrão, artefato de Gibbs e de compressão JPEG. Apesar de todos os tipos de artefatos observados terem significativo impacto na qualidade das imagens, esses artefatos não tem impacto equivalente na facilidade de diagnóstico de imagens degradadas, com ruído resultando em grandes reduções na acurácia de diagnóstico, enquanto artefatos de contraste e de ghosting tem impacto mais reduzido. Levando essas características em consideração, nossa função geradora de artefatos foi definida para gerar artefatos de forma realista.

Artefatos de imagem reduzem a acurácia de modelos no momento de gerar a predição de diagnóstico, porém um risco talvez ainda maior é no momento de treinar modelos com imagens de baixa qualidade. A presença de artefatos nas imagens de treinamento pode fazer com que modelos aprendam a correlacionar esses artefatos com diagnósticos específicos, um tipo de sobreajuste que pode resultar em elevadas acurácias no momento de teste do modelos, mas gerando modelos não confiáveis para a aplicação em casos reais.

A solução foi avaliada por meio da qualidade das imagens restaurada, medidas pela métrica MSSIM e por análises qualitativas; e pela acurácia na predição do diagnóstico das imagens. Artefatos artificiais foram adicionados a imagens de boa qualidade para testar a qualidade do método para artefatos específicos e diferentes intensidades de degradações. A solução também foi testado com 5 bases de dados de classificação de tumores cerebrais a partir de imagens de ressonância magnética. Essas bases incluem casos reais de imagens degradadas, demonstrando assim a efetividade da solução proposta para situações reais de imagens com degradações. Com as várias bases de dados também foram realizados testes cruzados, em que uma base de dados era utilizada para treinar o modelo e outras bases eram utilizadas para testes. Esses testes cruzados demonstraram o quanto a inclusão da etapa de restauração melhora a capacidade de generalização da solução.

Testes extensivos foram realizados para avaliar os diferentes aspectos da solução. O modelo Uformer foi comparado com outros modelos de restauração de imagem como UNet e autoencoders. O modelo EfficientNet foi comparado com outras redes neurais para classificação de imagem, incluindo transformers para visão (ViT), redes residuais e ConvNeXts. Bases de dados variadas foram utilizadas para treinar os modelos, observando-se o impacto que bases de dados diferentes têm na acurácia, qualidade de imagem e capacidade de generalização da solução. Métodos variados para gerar artefatos artificiais foram testados para treinar o modelo de

restauração, demonstrando quais aspectos são relevantes na geração de degradações artificiais. O solução também foi comparada com um método alternativo ou complementar de treinar o modelo de classificação em imagens com degradações artificiais, o qual ainda demonstra que a inclusão da etapa de restauração continua sendo efetiva. Os testes extensivos corroboram que a solução tem bons impactos na melhora de qualidade de imagens de ressonância magnética, aumento da acurácia na predição de diagnóstico e melhora na generalização dos modelos de classificação de imagens.

Palavras-chave: Tumor cerebral, RM, RNC, artefatos de imagem, restauração de imagens, diagnóstico por imagem.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

GLOSSARY

AE	Autoencoder
BN	Batch Normalization
CNN	Convolutional Neural Network
CNS	Central Nervous System
CT	Computed Tomography
DAE	Denoising Autoencoder
DL	Deep Learning
DNN	Deep Neural Network
ELU	Exponential Linear Unit
GAP	Global Average Pooling
GM	Gray Matter
GN	Group Normalization
IN	Instance Normalization
LN	Layer Normalization
MLP	Multi Layer Perceptron
MR	Magnetic Ressonance
MRI	Magnetic Ressonance Imaging
MSA	Multi-Head Self Attention
MSSIM	Mean Structural Similarity Index Measure
NN	Neural Network

PReLU	Parameterized Rectifier Linear Unit
PSNR	Peak signal to Noise Ratio
ReLU	Rectifier Linear Unit
RF	Radio Frequency
SNR	Signal to Noise Ratio
SSIM	Structural Similarity Index Measure
WM	White Matter
WMSA	Window Multi-Head Self Attention

CHAPTER 1

INTRODUCTION

Brain tumors are abnormal cell growths that affect the brain or central nervous system (CNS) (DEANGELIS, 2001; ISLAM *et al.*, 2024) and are estimated to affect between 7 and 11 individuals per 100,000 persons-year worldwide, leading to an estimated 200,000 deaths each year (MAHDAVI *et al.*, 2023; ILIC; ILIC, 2023). The 5-year survival rate of patients diagnosed with brain cancer can be as low as a disconcerting 6.9%, for the particular case of glioblastoma tumor(KHALIGHI *et al.*, 2024).

However, the prognosis for people with brain tumors can be improved by early detection and treatment, even allowing surgical resection of early stage tumors (MAHDAVI *et al.*, 2023; ISLAM *et al.*, 2024). The gold standard test for brain tumor diagnosis is cranial magnetic resonance imaging (MRI) (DEANGELIS, 2001; CHANDARANA *et al.*, 2018). MRI is a medical non-invasive imaging modality that employs radio frequency waves in the presence of carefully controlled magnetic fields to produce high fidelity images of internal anatomical structures (KATTI *et al.*, 2011; HYUN *et al.*, 2018; JHAMB *et al.*, 2015). Compared to other methods such as X-ray and computed tomography (CT), the advantages of MRI include the fact that it does not expose patients to potential hazardous radiation and produces high-resolution images that can even detect structural lesions or nonenhancing tumors that could be missed by other methods(KATTI *et al.*, 2011; DEANGELIS, 2001; BRENNER; HALL, 2007). Its main disadvantage is the fact that it requires a long exposure time to produce the images.

In order to assist in faster tumor detection, artificial intelligence (AI)-based diagnosis methods have shown potential to improve radiologists' efficiency and reduce human error (KHALIGHI *et al.*, 2024), with several deep learning (DL)-based methods proposed for automatic brain tumor diagnosis (REHMAN *et al.*, 2020; BADŽA; BARJAKTAROVIĆ, 2020; TUMMALA *et al.*, 2022; YAZDAN *et al.*, 2022; ISLAM *et al.*, 2024). As is the case for other imaging methods, MRI can be subject to image distortions that may be introduced during

acquisition, processing or compression (OKSUZ, 2021). Not only do these distortions affect the perceived quality of these images, they can also negatively affect the clinical utility of those images (RODRIGUES *et al.*, 2022). Machine learning diagnosis methods are specially affected by such image degradations, which may result in lower prediction confidence, lower accuracy and even incorrect predictions with high confidence (OKSUZ, 2021; DODGE; KARAM, 2016; FARIAS *et al.*, 2022; BAUCHSPIESS; FARIAS, 2024; REBUFFI *et al.*, 2021). In addition to lower inference performance, image distortions during DL model training can cause "noise learning," resulting in overfitting (YING, 2019; JABBAR; KHAN, 2015; POTHUGANTI, 2018). Correcting such image artifacts and distortions is therefore crucial for effective brain tumor analysis (KHALIGHI *et al.*, 2024).

Several works have been proposed to remove or alleviate image distortions in MRI and other modalities of medical images. Such works were used to restore images with noise (EL-SHAFAI *et al.*, 2022a; ZHONG *et al.*, 2020a), poor contrast (MZOUGHI *et al.*, 2019), blurring (YIM *et al.*, 2020; LIM *et al.*, 2020), Gibbs ringing (MUCKLEY *et al.*, 2021) and ghosting(LEE *et al.*, 2016b; JUREK *et al.*, 2023), among others. Most image restoration works use methods that focus on only one specific artifact type, but several images affected by a combination of multiple artifact types can be recognized when evaluating MRI images from popular brain tumor datasets, which are shown in Figure 1.1. Taken from the (NICKPARVAR, 2021) dataset, Fig. 1.1 (a) shows a scan with ghosting, noise and blurring artifacts, Fig. 1.1(b) shows a scan with contrast and JPEG compression artifacts and Fig. 1.1(c) shows a scan affected by ghosting, noise, blurring and poor contrast. Taken from the (SIAR; TESHNEHLAB, 2022) dataset, Fig. 1.1(d) shows a MRI scan affected by noise and Gibbs ringing artifacts.

We introduce a two-phase approach that aims at first restoring brain MRI scans, followed by classifying the presence and type of tumor in the enhanced image, offering a diagnosis prediction resistant to degradation. This method is tailored to work with images impacted by various types of artifacts, including those previously illustrated, ensuring that the image restoration system can handle images with mixed types of artifacts. Illustrated in Fig. 1.2, a Uformer model (WANG *et al.*, 2022) is used to restore the images, and an EfficientNet model (TAN; LE, 2019) is used to classify the restored images.

To train the restoration model, pairs of degraded and nondegraded images are required.

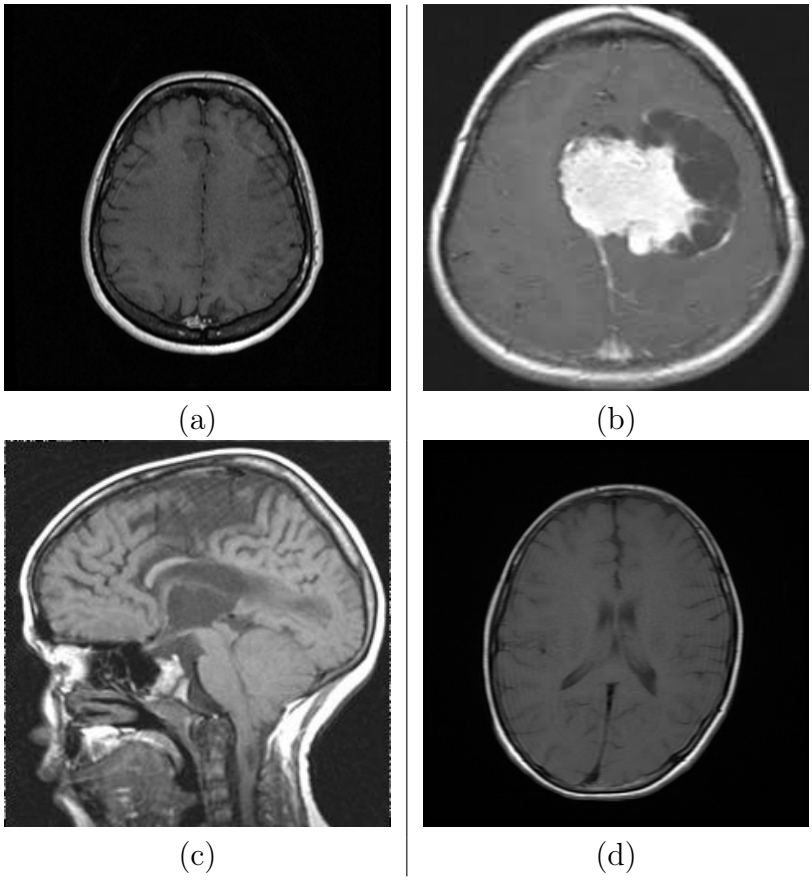


Figure 1.1: Various MRI scans with different artifacts: (a) ghosting, noise, and blurring; (b) contrast and JPEG compression; (c) ghosting, noise, blur, and contrast; (d) noise and Gibbs ringing.

We use good quality images as the non-degraded images and artificially add degradations to generate the corresponding degraded image. To make the restoration model more effective in real world scenarios, artificial degradations should resemble those of real cases. We evaluated real cases of degraded brain MRI to identify the most common artifact types, estimate the magnitude distribution of these artifacts, and understand the impact of multiple artifacts simultaneously affecting a single image. Common artifacts identified were Rician noise, blurring, Gibbs ringing, poor contrast, Nyquist ghost, and JPEG compression artifact. One relevant observation is that different types of artifact affect each other, and the image restoration model has to account for that to restore images more effectively. Based on this study, we defined an artifact generation function that generates realistic degraded images, accounting for artifact types, intensities, and how images are affected by multiple artifacts simultaneously.

Our method was tested on five different datasets with real cases of image degradation and also on artificially degraded images. The results show that the inclusion of the image resto-

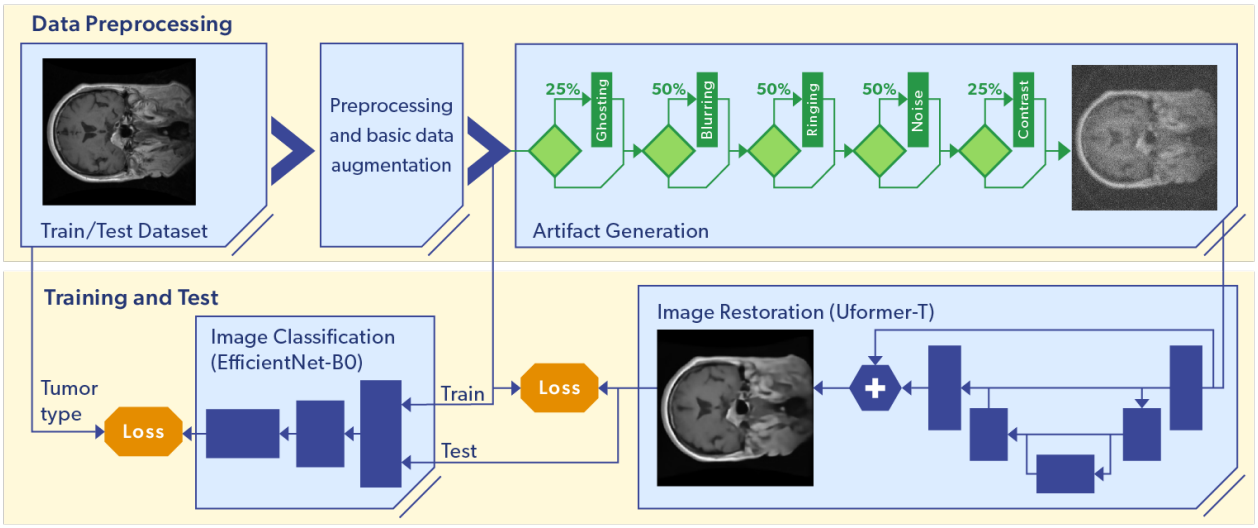


Figure 1.2: Proposed framework split into five modules: Dataset, preprocessing, artifact generator, restoration model and classification model. Modified from (BAUCHSPIESS; FARIAS, 2024).

ration step leads to more accurate diagnosis predictions, showing that the restoration models trained on artificially degraded images generalize well to real cases. By cross-validating between different datasets, our results show that our two-stage approach significantly improves the generalization of the computer-aided diagnostic method. In our experiments, we also observed how image quality affects the generalization capabilities of image classification models, with poor quality images leading to significant model overfitting.

We conducted extensive experiments to evaluate the different aspects of the proposed solution. Experiments included changing various aspects of the artifact generation function, comparing different image restoration models and various image classification models, and changing the datasets used to train the models. The solution was evaluated qualitatively and quantitatively, using accuracy and MSSIM metrics and confusion matrices.

The organization of the text is as follows. Chapter 2 presents a review of various techniques for image restoration, detailing the types of images and artifacts considered and the methods for diagnosing brain pathology by MRI. In Chapter 3, we provide an overview of brain tumors, including the subtypes examined in this study, and the basic concepts of magnetic resonance imaging. This chapter further outlines the artifact types and evaluation metrics utilized, along with a discussion on the layer types and architectures used to develop neural network models and their training methodologies. The chapter concludes with an outline of the baseline mo-

dels and datasets used for testing, including qualitative dataset analysis. In Chapter 4, the Methods section elaborates on the suggested solution in more detail, examining the definition of the artifact generation function, as well as the approaches for training and assessing the models. Chapter 5, the Experiments section, outlines a series of findings that demonstrate the performance of the solution and analyze its various components. These experiments investigate the correlation between image quality and accuracy, the effect of differing artifact types and intensities, the results of diverse artifact generation techniques, the efficacy of multiple image restoration models, and the performance of various image classification models. The dissertation is ultimately wrapped up in Chapter 6, the Conclusion section.

CHAPTER 2

STATE OF THE ART

In this chapter, we review the state-of-the-art in image restoration, with a focus on medical images and methods to predict daignosis based on medical images.

2.1 IMAGE RESTORATION

Mzoughi *et al.* stating that reduced MRI acquisition time results in degraded image contrast and SNR, which reduces the precision of clinical and computer-aided diagnosis (MZOUGHI *et al.*, 2019). Their work proposes the use of a bilinear filter to reduce noise and a contrast stretching function to adjust contrast. The contrast stretching function multiplies each voxel by a parameter that depends on image mean and variance and if the tumor is high-grade or low-grade, with low-grade tumors requiring an increase in contrast to better distinguish the tumor. This contrast enhancement method may be used to improve the accuracy of tumor classification as high-grade or low-grade (MZOUGHI *et al.*, 2020).

El-Shafai *et al.* trained an autoencoder to denoise CT and X-Ray torax images with artificially added Gaussian, salt-and-pepper, and speckle noise and used Convolutional Neural Networks (CNNs) to classify those images (EL-SHAFAI *et al.*, 2022a). Their results show that using the autoencoder denoising as a pre-processing step increased the classification CNN accuracy.

Zhong *et al.* proposed to train a residual CNN to denoise CT images of multiple parts of the body (ZHONG *et al.*, 2020a). This CNN model estimates the difference between the clean and noisy image, which is subtracted from the input image to obtain the estimated clean image. Their work argues that pre-training the model to denoise natural images before training on CT images results in improved model performance.

Yim *et al.* proposed to train two autoencoders, one for denoising and one for deblurring,

and then concatenate these models with a convolutional layer and retrain this final model, applied to lung CT images (YIM *et al.*, 2020). Results indicate that this models obtain better quality image than a simple CNN in terms of Peak Signal to Noise Ratio (PSNR), Mean Square Error (MSE) and Stuctural Similarity Index Measure (SSIM). Muckley et al. proposed using a residual model based on the UNet family to remove MRI noise and Gibbs ringing artifacts (MUCKLEY *et al.*, 2021).

Wang *et al.* proposed a residual UNet model for natural image restoration called Uformer (WANG *et al.*, 2022). The model architecture is based on the vision transformer architecture, to which depth-wise convolutions were added to the feed forward block. The degradations considered were various types of noise, blur, and rain artifacts. Color images taken from smartphones and handheld cameras were considered.

Zamir *et al.* proposed another residual UNet model based on the addition of convolutions to the transformer architecture, with the aim of restoration of natural images (ZAMIR *et al.*, 2022). The model adds depth-wise convolution to both the self-attention and the feedforward blocks of the transformer architecture and adds a gating mechanism to the feedforward block. Rain, noise, and blur artifacts, including motion blur and defocus blur, were considered.

Lee *et al.* proposed to restore MRI images affected by Nyquist ghost artifact without reference, using the ALOHA algorithm (LEE *et al.*, 2016a; LEE *et al.*, 2016b). The method is shown to perform even better than reference-based models, without as many requirements. Lim *et al.* trained a residual CNN to spatially vary blur correction in real time for MRI (LIM *et al.*, 2020). The model is evaluated both quantitatively with quality metrics, including PSNR and SSIM, and qualitatively through visual inspection.

Jurek *et al.* revert images to the k-space and introduce Nyquist ghost and ramp sampling effects in this order and then add noise in the image domain (JUREK *et al.*, 2023). The noise is obtained from an air scan. The model trained with all effects resulted in loss of edges as well as noise removal; for this reason the model was then trained only with phase effects and noise, with better results. This work also considered the blurring function of imaging systems to be a Point Spread Function that approximates the Gaussian function, with standard deviation around 0.65, and when applicable adds this blurring before noise. The model used to denoise the image was a residual CNN, so the CNN estimates the noise to be subtracted from the

Table 2.1: Comparison of different image restoration methods, based on artifact type, image type and algorithm used to restore the images.

Paper	Artifacts	Image type	Method
(LEE *et al.*, 2016b)	Ghosting	MRI	ALOHA
(MZOUGHI *et al.*, 2019)	Rician noise, contrast	Brain MRI	Bilinear filter, contrast stretching
(YIM *et al.*, 2020)	Gaussian noise, Gaussian blur	Lung CT	Autoencoder
(ZHONG *et al.*, 2020a)	Gaussian noise	CT	Residual CNN
(LIM *et al.*, 2020)	Blur	MRI	Residual CNN
(MUCKLEY *et al.*, 2021)	Gibbs ringing, Gaussian noise	MRI, natural images	Residual UNet
(EL-SHAFAI *et al.*, 2022a)	Gaussian, salt and pepper and speckle noise	Torax CT, X-Ray	Autoencoder
(WANG *et al.*, 2022)	Noise, blur	Natural images	Conv+transformer residual UNet
(ZAMIR *et al.*, 2022)	Noise, blur	Natural images	Conv+transformer residual UNet
(JUREK *et al.*, 2023)	Rician noise, ramp-sampling effects, ghosting, Gaussian blur	Brain MRI	Residual CNN
Proposed	Rician noise, ghosting, Gibbs ringing, Gaussian blur, Poor Contrast, JPEG compression	Brain MRI	Conv+transformer residual UNet

image. The work proposes first applying the Nyquist ghost correction and then applying the denoising method, arguing that the reverse order yielded worse results. Two reference methods were considered: averaging of repeated scans and the blockwise Non-Local Means (NLM).

We selected the Uformer model introduced by (WANG *et al.*, 2022) as our baseline model. The study used 2D brain magnetic resonance images and evaluated artifact types including Rician noise, Gaussian blur, Gibbs ringing, ghosting, low contrast, and JPEG compression artifacts. Table 2.1 presents a comparison among various works, highlighting the artifact types, image types, and restoration techniques used.

2.2 BRAIN IMAGE CLASSIFICATION

Korolev *et al.* proposed to use 3D CNNs with and without residual to classify 3D MRI images into Alzheimer's disease, late mild cognitive impairment, early mild cognitive impairment and normal cohort (KOROLEV *et al.*, 2017). Their results indicate that CNNs with and without residuals perform similarly. Rehman *et al.* evaluated a range of convolutional neural network models alongside an optimization algorithm to categorize two-dimensional MRI data into three distinct types of brain tumors: meningioma, glioma, and pituitary (REHMAN *et al.*, 2020). The method included a contrast stretching pre-processing step.

Badza and Barjaktarovic proposed their own CNN model to classify 2D MRI as meningioma, glioma, and pituitary (BADŽA; BARJAKTAROVIĆ, 2020). They observed that the dataset tested (CHENG *et al.*, 2016) included multiple scans of the same subjects. They separated the images into train sets and test sets in two manners. In the first manner, scans were separated by subject, i.e. images belonging to the same subject were all put in the train set or all put in the test set. In the second manner, the scans were separated individually, so that one scan from a subject could be in the train set and another scan from the same subject could be in the test set. They observed that this second manner led to higher test accuracy, indicating a kind of overfitting in which the model recognized the subject rather than the tumor type.

Tummala *et al.* proposed to use an ensemble of vision transformers (ViT) (DOSOVITSKIY *et al.*, 2020) to classify 2D MRI into meningioma, glioma, and pituitary (TUMMALA *et al.*, 2022). They increased performance by increasing image resolution and combining the outputs of multiple models. Yazdan et al. proposed a multiscale CNN to classify 2D MRI into four classes: meningioma, glioma, pituitary and non-tumor (YAZDAN *et al.*, 2022). They included a pre-processing step to remove Rician noise, using a Fuzzy Similarity-based Non-Local Means (FSNLM) filter to improve accuracy. The pre-processing step was shown to improve accuracy both on the dataset original data as well as on the dataset with synthetic noise added. Islam et al. proposed to use EfficientNet (TAN; LE, 2019) CNN architecture to classify 2D MRIs into meningioma, glioma, and pituitary tumor classes (ISLAM *et al.*, 2024). They argue that the EfficientNet architecture achieves higher accuracy with fewer computational resources than other CNN architectures.

Table 2.2: Comparison of different image classification methods, based on the target classes, type of image and the type of model used to predict the output.

Paper	Classes	Image type	Method
(KOROLEV *et al.*, 2017)	Alzheimer disease	3D MRI	3D CNN
(REHMAN *et al.*, 2020)	Brain tumor	2D MRI	CNN
(BADŽA; BARJAKTAROVIĆ, 2020)	Brain tumor	2D MRI	CNN
(TUMMALA *et al.*, 2022)	Brain tumor	2D MRI	ViT
(YAZDAN *et al.*, 2022)	Brain tumor	2D MRI	CNN
(ISLAM *et al.*, 2024)	Brain tumor	2D MRI	EfficientNet
Proposed	Brain tumor	2D MRI	EfficientNet

For our work, we used EfficientNet models to classify 2D MRI images in different brain tumor categories. Five different brain tumor datasets were considered, which included the categories glioma, meningioma, pituitary, non-tumor and generic tumor classes; with different combinations depending on the dataset. Multiple models were trained depending on the classes included in each dataset. Tests were performed on datasets with undegraded and distorted images, as well as images with synthetically added artifacts, and we evaluate the accuracy improvement when an image restoration step is included. Table 2.2 summarizes the methods, based on the type of pathology predicted, the data type used and the method or neural network architecture used for classification.

CHAPTER 3

FUNDAMENTALS

In this chapter, we review concepts and fundamentals required for the research.

3.1 BRAIN TUMOR

Brain tumors encompass various neoplasms, each characterized by distinct biology, prognosis, and treatment approaches; however, a more precise term is intracranial neoplasms because some tumors, such as meningiomas, are not related to brain tissue (DEANGELIS, 2001). These tumors can cause focal or generalized neurological symptoms such as headaches, nausea, vomiting, sixth-nerve palsy, hemiparesis, aphasia, and seizures. Brain tumors vary widely in type. High-grade gliomas and meningiomas are the predominant forms of primary brain tumors in adults (BONDY *et al.*, 2008). Another fairly prevalent tumor is the pituitary tumor, which makes up approximately 15% of intracranial neoplasms (CHATZELLIS *et al.*, 2015). The preferred diagnostic method for brain tumors is magnetic resonance imaging with gadolinium enhancement (DEANGELIS, 2001). In this section a concise overview is provided, accompanied by MRI scans of these three types of tumors.

3.1.1 Glioma

The term glioma encompasses all forms of tumors believed to originate from glial cells (SCHWARTZBAUM *et al.*, 2006), and can also be known as glial tumors (DEANGELIS, 2001). Gliomas exhibit a heterogeneous appearance and have poorly defined boundaries. The margins of active tumors do not align closely with contrast enhancement features, and pathological contrast enhancement is generally associated with more aggressive tumors (UPADHYAY; WALDMAN, 2011). Figure 3.1 presents several slices of MRI showing glioma tumors (CHENG *et al.*, 2016).

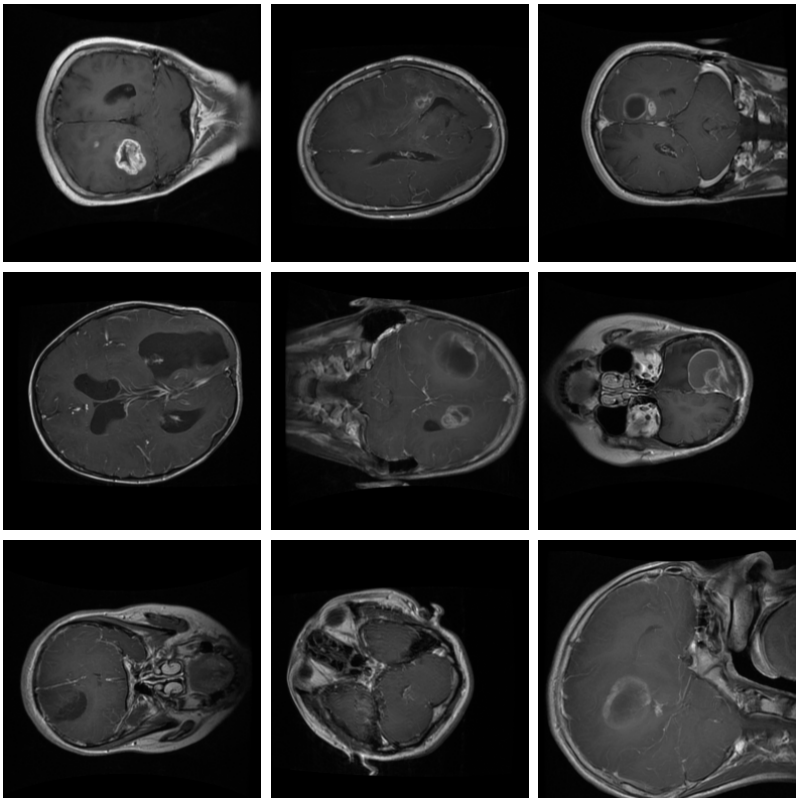


Figure 3.1: Glioma samples taken from a brain tumor dataset (CHENG *et al.*, 2016).

3.1.2 Meningioma

Meningioma tumors arise from meningothelial cells in the outer membrane of the brain and are found primarily at the base of the skull, over cerebral convexities, and within the parasellar areas (DEANGELIS, 2001). These tumors are often asymptomatic and are often discovered accidentally during autopsies. On MRI scans, meningiomas are usually observed near the bones. Figure 3.2 presents several slices of MRI showing meningioma tumors, sourced from (CHENG *et al.*, 2016).

3.1.3 Pituitary tumors

Pituitary tumors are marked by excessive growth of cells in the anterior pituitary and dysregulated overproduction of specific hormones(MELMED, 2015; DAI *et al.*, 2021a). These tumors are predominantly benign and are known as pituitary adenomas, with a small number classified as pituitary carcinomas. The range of symptoms, such as hypertension, psychological problems, headaches, and soft tissue swelling, varies depending on the cell type and hormones

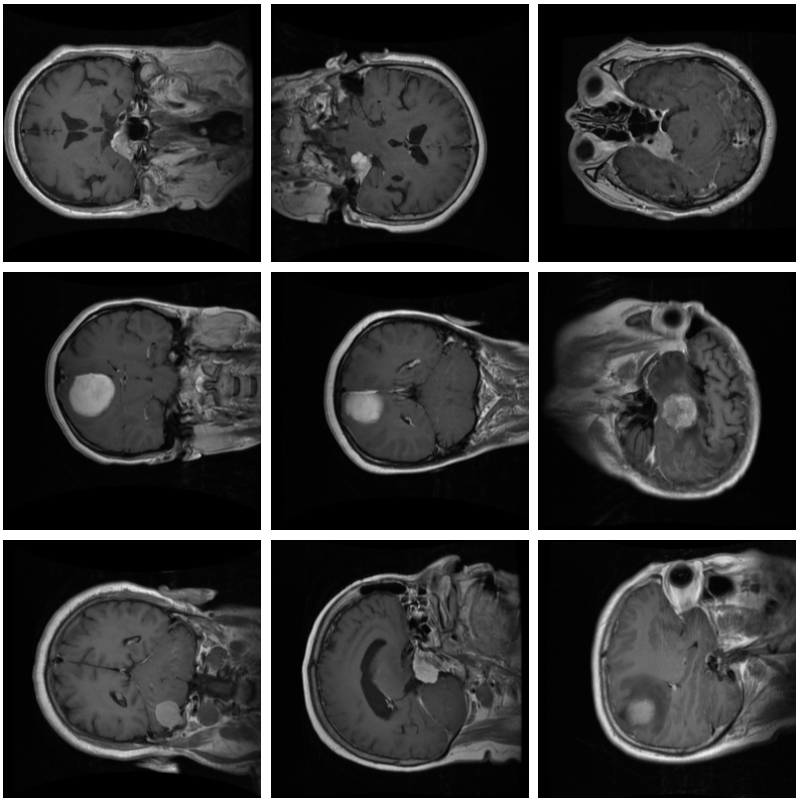


Figure 3.2: Meningioma samples taken from a brain tumor dataset (CHENG *et al.*, 2016).

involved. Fig. 3.3 illustrates example MRI scan slices of pituitary tumors (CHENG *et al.*, 2016).

3.2 MAGNETIC RESONANCE IMAGING - MRI

Magnetic Resonance Imaging (MRI) is a non-invasive medical imaging modality for mapping internal structures of the body. It uses non-ionizing electromagnetic radio frequency (RF) radiation in the presence of carefully controlled magnetic fields to produce high-quality cross-sectional images of the body in any plane (KATTI *et al.*, 2011). MRI machines produce relatively strong magnetic fields that cause the nuclei of atoms in the body, including hydrogen, to align with it. When energy in the form of RF electromagnetic waves is directed at it, protons in tissue that have Larmor frequency matching that of the RF wave absorb energy and rotate away from the direction induced by the magnetic field; the longer the exposition, the larger the rotation. When radio waves are turned off, the radiation energy is released to surrounding molecules and the protons realign with the magnetic field, a process called T1 recovery, and the energy loss is detected as a signal (KATTI *et al.*, 2011; WESTBROOK, 2016).

MRI has some different modalities that lead to images with different characteristics. Two

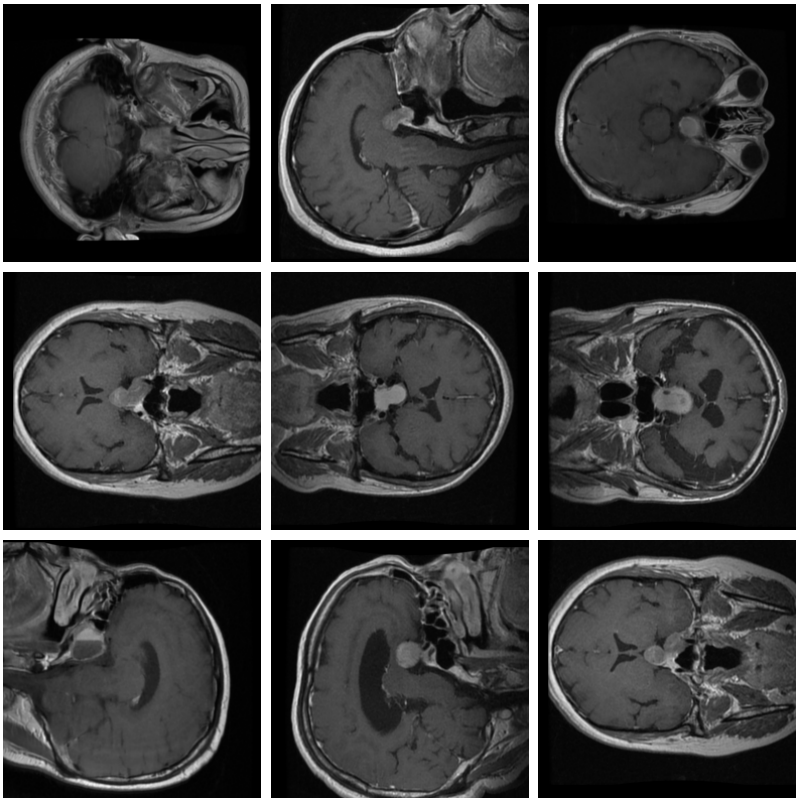


Figure 3.3: Pituitary tumor samples taken from a brain tumor dataset (CHENG *et al.*, 2016).

common modalities are the T1 weighted (T1) and the T2 weighted (T2). For brain tumor diagnosis, T1 is considered good for tumor segmentation, and T2 makes the fluid around the tumor visible (DAIMARY *et al.*, 2020). In T1 brain scans white matter (WM) is lighter than gray matter (GM), while in T2 scans white matter is darker than the gray matter [1] Fig.3.4 exemplifies the T1-CE and T2 MRI modalities for brain scans with glioma tumors.

The raw data obtained from MRI scanning devices are spatial frequency information of an object, usually referred to as K-space data (MEZRICH, 1995; MORATAL *et al.*, 2008; JHAMB *et al.*, 2015). The K-space data contains the density of nuclear magnetic resonance signals generated from the body. To obtain a conventional image or spatial volume from k-space data, the inverse Fourier transform is applied (GALLAGHER *et al.*, 2008; CÁRDENAS-BLANCO *et al.*, 2008). The signal obtained from the inverse Fourier transform is a complex signal, with real and imaginary parts, but it is common practice in MR to obtain the magnitude of this signal, resulting in real-value magnitude images.

Brain MRI scans are usually 3D volumes that can be sliced into 2D images, typically in 3 orientation planes: axial, sagittal, and coronal (PADMANABAN *et al.*, 2020; DAIMARY *et*

[1] https://radiopaedia.org/articles/mri-sequences-overview

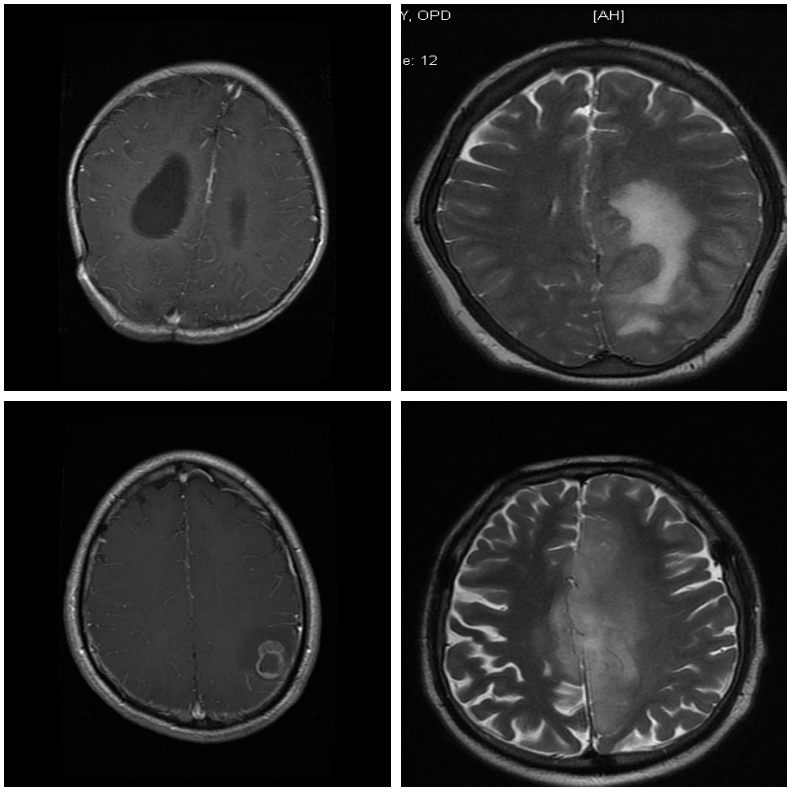


Figure 3.4: Brain MRI scan modalities. Left: T1-CE brain scans taken from (CHENG *et al.*, 2016). Right: T2 brain scans taken from (QADRI *et al.*, 2022).

al., 2020). The axial plane goes from the top to the bottom of the head, the coronal plane goes from front to back, and the sagittal plane goes from side to side. Fig. 3.5 illustrates the 3 planes.

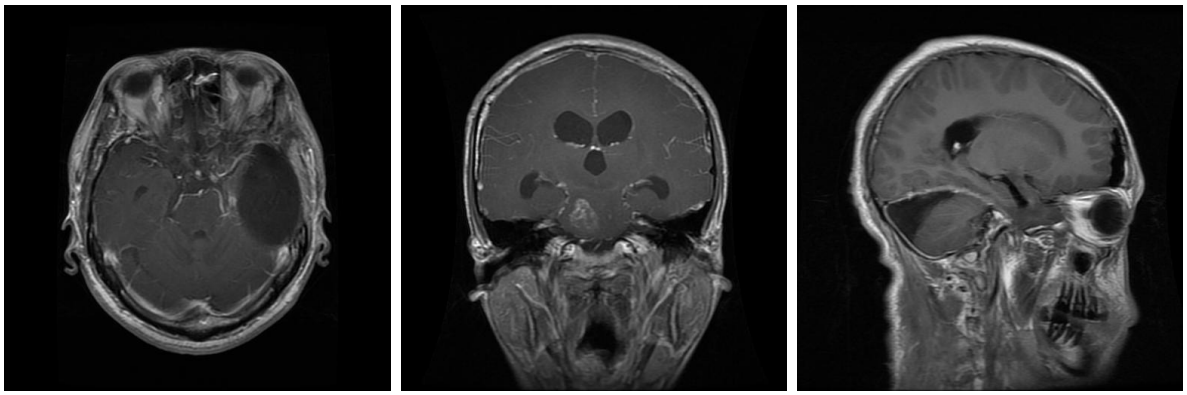


Figure 3.5: Brain MRI planes. From left to right: Axial, coronal and sagittal.Images taken from the Figshare dataset (CHENG *et al.*, 2016).

3.3 MRI ARTIFACTS

MRI artifacts are degradations that affect the quality of the image and affect the classification performance of neural networks (FARIAS *et al.*, 2022). Types of MRI artifacts include noise, Gibbs ringing, blurring, ghosting, and poor contrast. Image compression can also generate artifacts such as JPEG compression artifacts. In this section, we review these six types of image artifacts that can affect MRI scans.

3.3.1 Noise

Noise in MRI images is described as complex additive signals, in which its real and imaginary components are independent, identically distributed, zero mean Gaussian distributions (CÁRDENAS-BLANCO *et al.*, 2008). A noisy k-space MR signal is then defined as

$$S = S_R + \mathcal{N}(0, \sigma^2) + j \cdot (S_I + \mathcal{N}(0, \sigma^2)), \tag{3.1}$$

where S_R and S_I are, respectively, the real and imaginary parts of the signal and $\mathcal{N}(0, \sigma^2)$ is the noise drawn from a normal (Gaussian) distribution with mean value 0 and standard deviation σ. Note that the noise has a real and an imaginary part, both drawn from the same distribution. Those values are, however, independent, i.e. the actual value of the real part does not influence the value of the imaginary part and vice versa. Fig. 3.6 illustrates the signal in the k-space with added noise.

The spatial MRI signal is obtained through the inverse Fourier transform, which keeps the additive properties of the noise:

$$\mathcal{F}^{-1}(S) = \mathcal{F}^{-1}(S_R) + \mathcal{F}^{-1}(\mathcal{N}(0, \sigma^2)) + j \cdot (\mathcal{F}^{-1}(S_I) + \mathcal{F}^{-1}(\mathcal{N}(0, \sigma^2))). \tag{3.2}$$

However, when the magnitude MRI image is obtained from these complex signals,

$$Magnitude = \sqrt{(\mathcal{F}^{-1}(S_R) + \mathcal{F}^{-1}(\mathcal{N}(0, \sigma^2)))^2 + (\mathcal{F}^{-1}(S_I) + \mathcal{F}^{-1}(\mathcal{N}(0, \sigma^2)))^2}, \tag{3.3}$$

there is a change to this distribution, due to the non-linear nature of this transform.

The probability distribution of magnitude MRIs in the presence of noise is characterized by the Rician distribution (RICE, 1944; GUDBJARTSSON; PATZ, 1995), defined by:

$$P_M(M) = \frac{M}{\sigma^2} e^{-(M^2+A^2)/2\sigma^2} I_0\left(\frac{A \cdot M}{\sigma^2}\right), \tag{3.4}$$

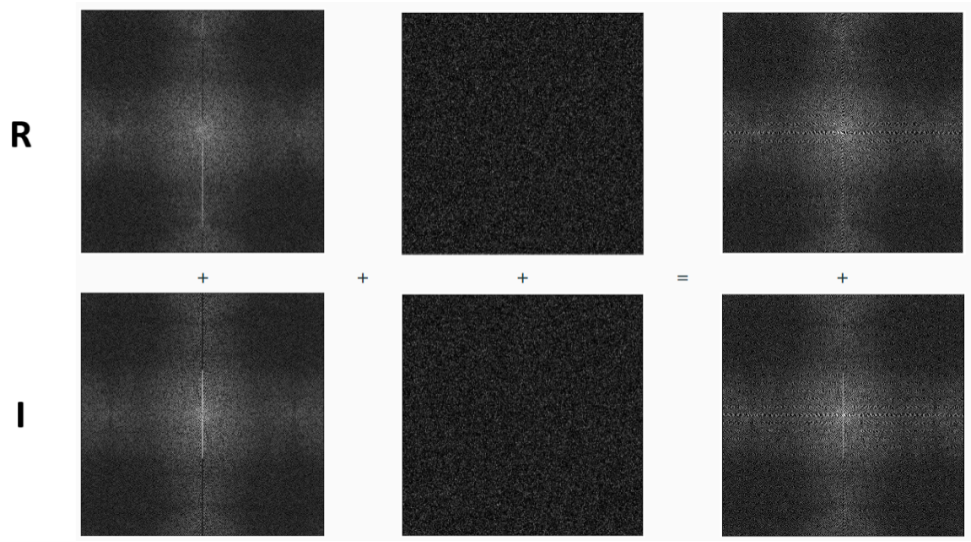


Figure 3.6: Frequency signal with added noise, with real (R) and imaginary (I) parts.

where M is the measured pixel intensity, A is the pixel intensity without noise, I_0 is the zeroth order Bessel function, and σ is the standard deviation of the Gaussian noise in the complex image. It should be noted that this distribution is influenced by the signal-to-noise ratio (SNR, A/σ) and is not the noise distribution on its own (CÁRDENAS-BLANCO *et al.*, 2008), although several works refer to noise in MRI as Rician noise (COUPÉ *et al.*, 2010; BASU *et al.*, 2006).

Gudbjartsson and Patz point out that for $A = 0$, the Rician distribution is equivalent to the Rayleigh distribution (RAYLEIGH, 1896; BECKMANN, 1964), while for SNR $(A/\sigma) > 2$ the distribution approximates the Gaussian distribution as seen in Fig. 3.7(GUDBJARTSSON; PATZ, 1995). Like the Rayleigh distribution, the Rician distribution is exclusively positive. Another observation is that the image noise adds a bias shift in the image mean, such that the mean image value may be defined approximately by:

$$\bar{M} = \sqrt{A^2 + \sigma^2}. \tag{3.5}$$

To generate artificial Rician noise in magnitude images we apply

$$\text{RicianNoise}(I, \sigma) = \text{magnitude}\left(I + \mathcal{N}(0,\sigma) + j \cdot (\mathcal{N}(0,\sigma))\right). \tag{3.6}$$

where j is the imaginary unit, I is the clean image and $\mathcal{N}(0,\sigma)$ is a function to generate noise from a Gaussian distribution with mean zero and standard deviation σ. We defined the standard deviation of the noise function based on an *intensity* parameter as

$$\sigma = 0.025 \cdot intensity, \tag{3.7}$$

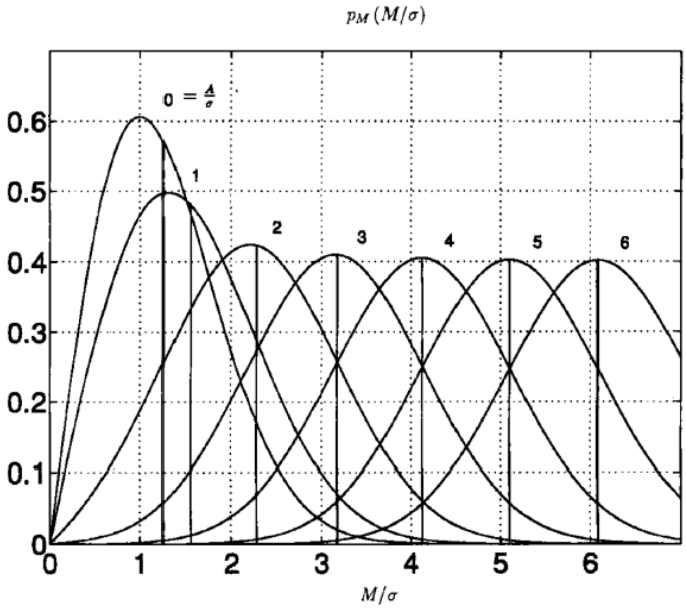


Figure 3.7: Rician probability distribution for different SNR. Extracted from(GUDBJARTSSON; PATZ, 1995).

with which we generated Rician noise examples with intensities ranging from 1 to 10, shown in Figure 3.8.

3.3.2 Gibbs-Ringing

The Gibbs artifact, also known as a ringing or truncation artifact, is characterized by spurious ringing near sharp edges of reconstructed images when the Fourier space is truncated at acquisition or compression, and thus lacks an adequate number of high-frequency terms (VERAART *et al.*, 2016; GALLAGHER *et al.*, 2008; WILBRAHAM, 1848, 1848). In Figure 3.9, this can be seen as a ripple effect associated with the sharp edges of the cranium in the MRI.

The most common method to suppress this artifact in images is to apply spatial smoothing (blurring); however, this method inherently lowers the image resolution and adds partial volume effects (VERAART *et al.*, 2016). Gibbs-ringing can be artificially generated by masking the frequency domain image, such that only lower frequencies are considered. This is the ideal low-pass filter. To illustrate this artifact, a circular mask was defined with radius according to

$$\text{radius} = (10 - \text{intensity})/9 \cdot 50 + 16, \tag{3.8}$$

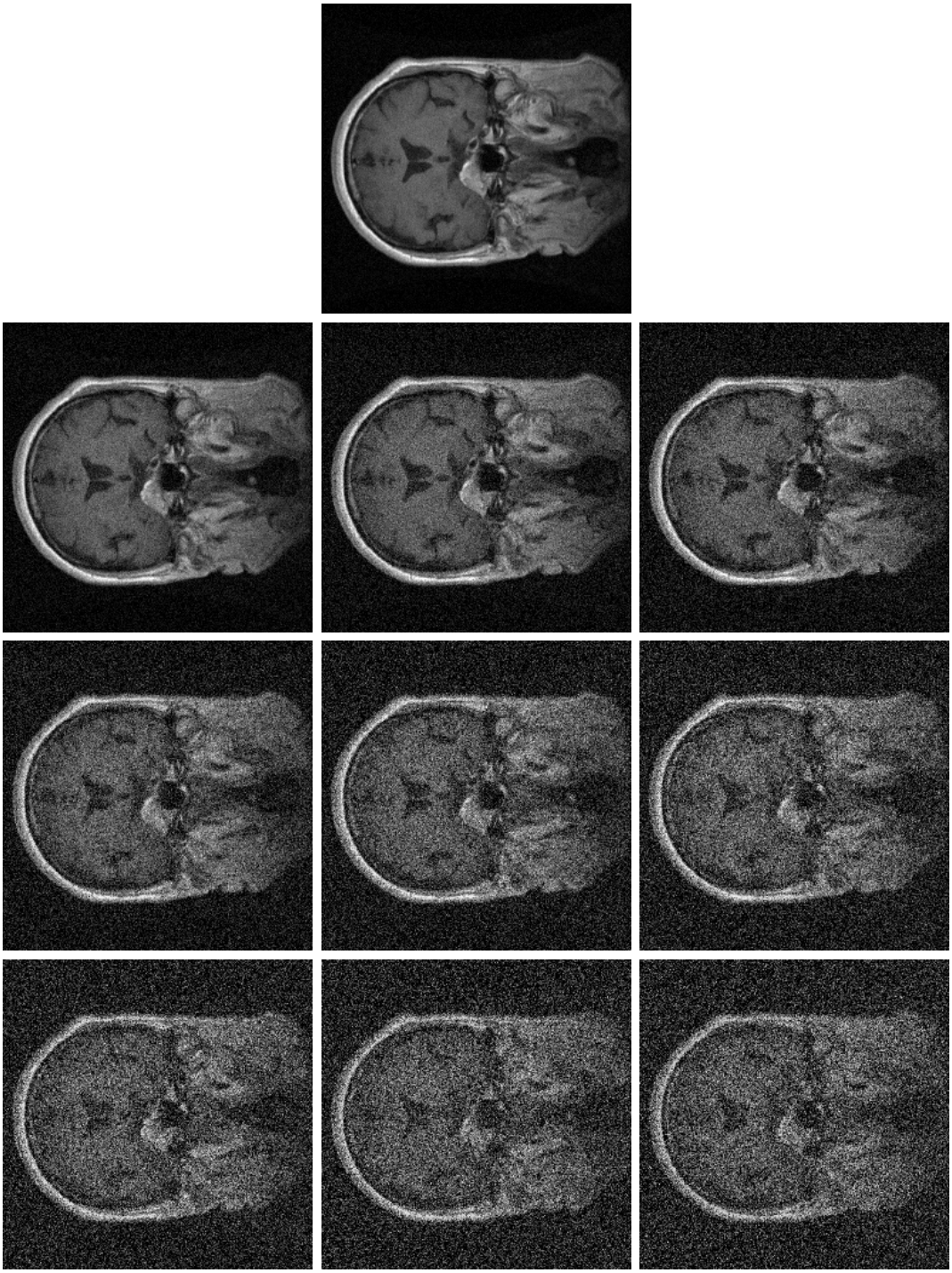


Figure 3.8: MRI with artificially applied Rician noise with levels from 1 to 10, increasing from left to right and top to bottom.

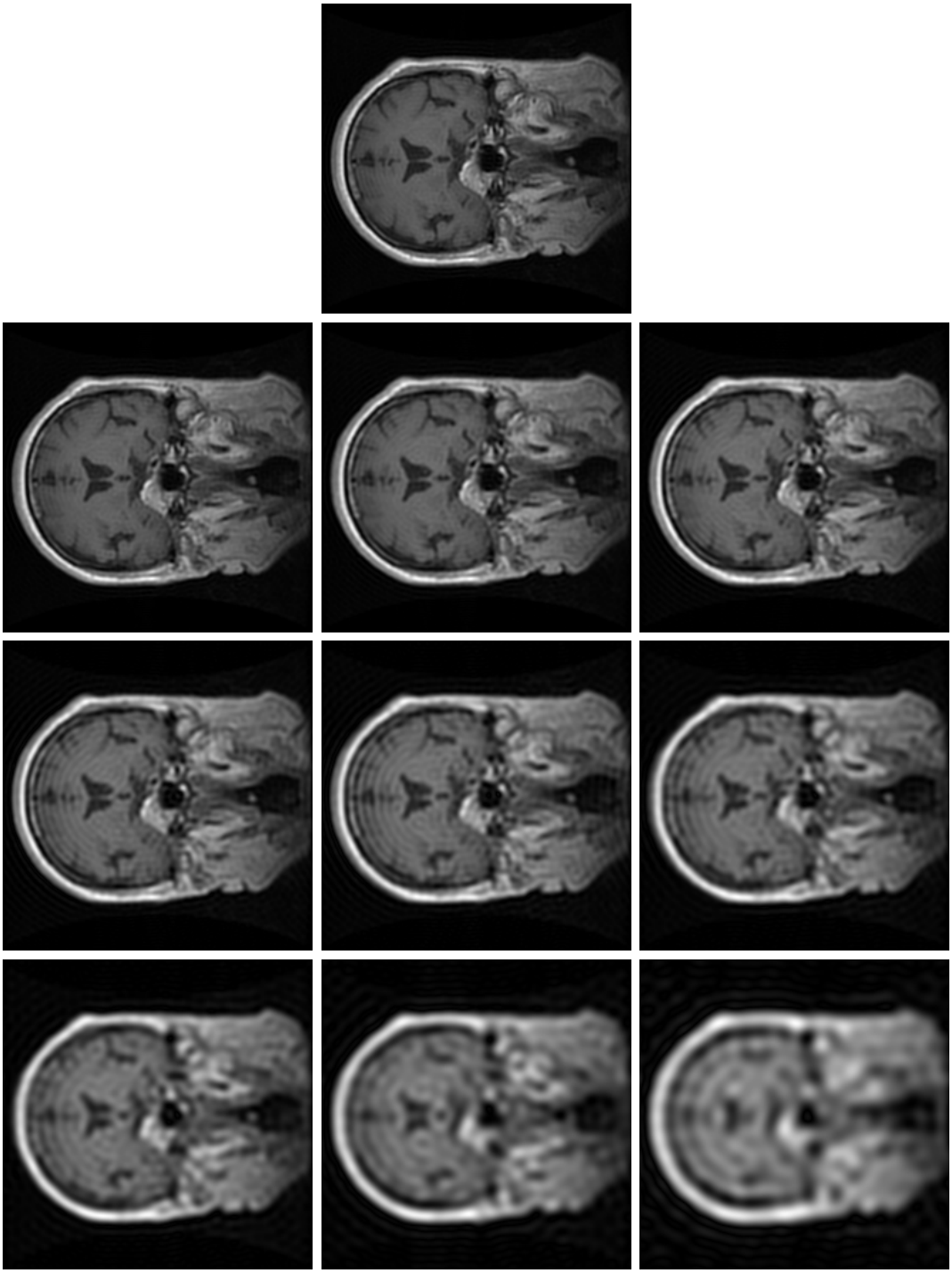


Figure 3.9: MRI with artificially applied Gibbs ringing artifact with levels from 1 to 10, odd values on the left and even values on the right increasing from top to bottom.

in which *intensity* is a control parameter. Figure 3.9 illustrates the effect of this filter for intensity values from 1 to 10. As can be seen in the figure, for higher intensities, the rings tend to be fewer and coarser, and there is loss of details in the image, as only lower frequencies are used to reconstruct the image.

3.3.3 Blurring

Blur might be defined as "to render indistinct". In the technical context of vision, it implies the smearing of an image, through some sort of low-pass filtering (WATSON; AHUMADA, 2011). This may result in loss of image detail, such as images with less clear borders and image regions. Image blurring has multiple sources and types. Blurring in magnetic resonance imaging may occur due to the presence of ferromagnetic sources or as a result of methods to remove other distortions in images (BUI *et al.*, 2000).

To generate blurring in the MRI scans, we applied a Gaussian blur average filter to these images. To define different intensities of the blurring artifact the standard deviation value of the Gaussian filter kernel was defined as

$$\sigma = 0.015 + 0.4985 \cdot intensity, \tag{3.9}$$

such that by increasing the standard deviation of the filter, more detail is lost in the scans, as shown in Fig. 3.10.

3.3.4 Contrast

The contrast of an image is represented as the disconnection between the brightest and darkest spots of an image(MAHMOOD *et al.*, 2019). Poor contrast in images is usually related to a low pixel intensity difference between darker and brighter regions of the image, making it harder to distinguish between different regions or edges in the image. Contrast enhancement is beneficial for many vision tasks, such as color segmentation, edge detection, image sharpening, and visualization (LI *et al.*, 2014) and for accurate segmentation of medical images(ZHAO *et al.*, 2024).

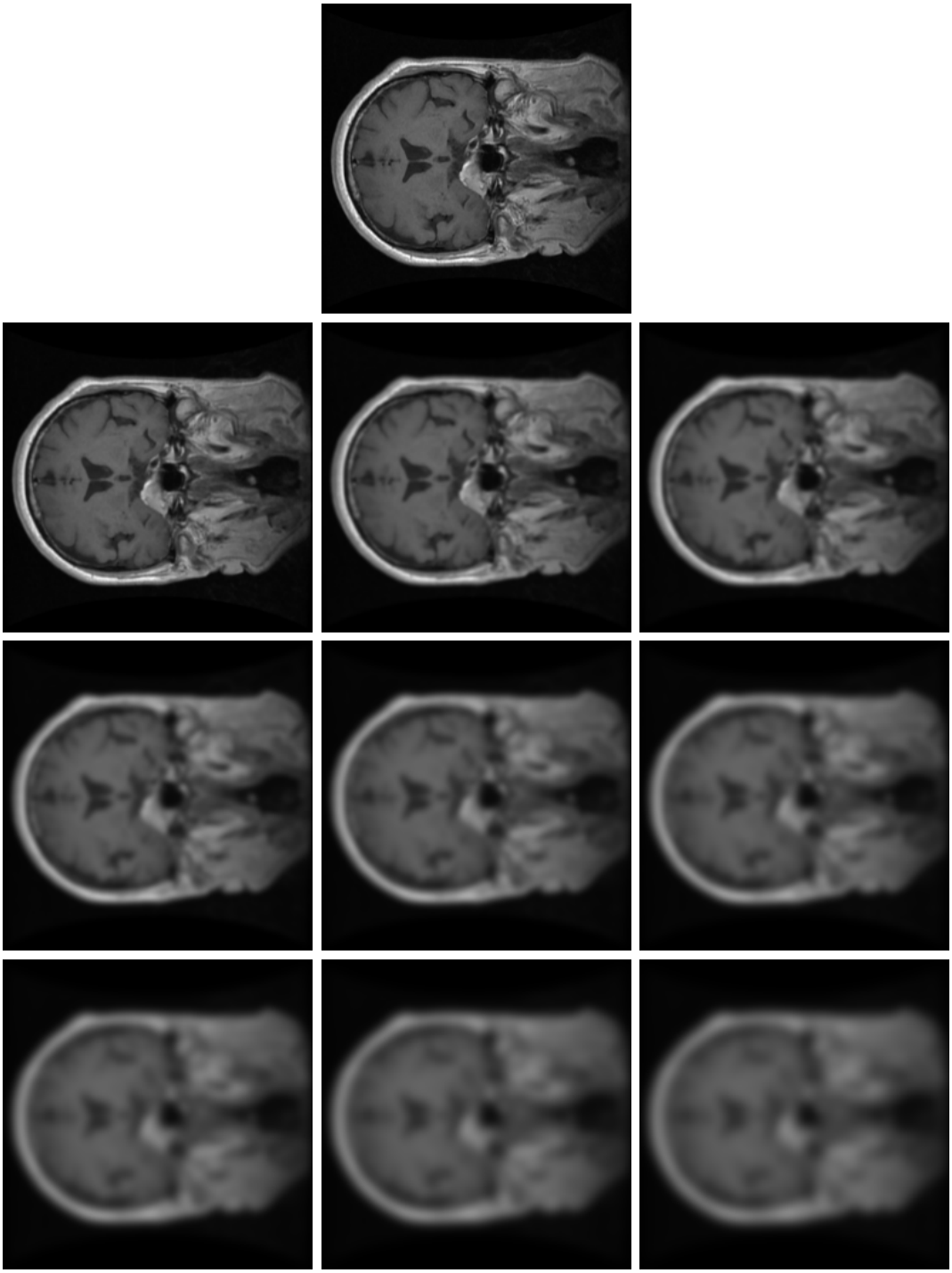


Figure 3.10: MRI with artificially applied blurring artifact with levels from 1 to 10, increasing from left to right and top to bottom.

We can introduce contrast artifacts in an image using the following equation:

$$I_d = I_o \times \alpha + \beta, \tag{3.10}$$

where I_d is the image with contrast artifacts and I_o is the original image, α is a contrast parameter and β is a brightness which we define based on an *intensity* variable as

$$\begin{aligned} \alpha &= (11 - intensity) \times 0.009 \\ \beta &= 255 \times (1 - \alpha) \end{aligned} \tag{3.11}$$

. Based on these equations, we generate 10 examples of low-contrast images with intensities 1 through 10, shown in Fig.3.11.

3.3.5 Ghosting

Gradient delay associated with alternating readout polarity leads to misalignment between positive and negative echoes during MRI data sampling, resulting in a Nyquist ghost artifact in the reconstructed images (LIU *et al.*, 2023). These artifacts appear as repeated versions of the main object, translated and with lower intensity.

To generate ghosting, we used a function provided by the TorchIO library (<torchio. readthedocs.io>), which allows us to set the number of ghosts, its intensity, and on which axis the ghosts are spread. We generated 10 ghosting examples at intensities 1 through 10 in Fig. 3.12.

3.3.6 JPEG compression

JPEG is a common image compression method and stands for Joint Photographic Experts Group, which is the name of the committee that created the standard(WALLACE, 1992). JPEG is a commonly used method of lossy compression for digital images, particularly for photographs. In JPEG, it is possible to set a compression quality level, and the lower the quality, the smaller the compressed file. Images compressed with lower levels of quality may have different types of artifacts, such as loss of details, quantization noise, blockiness, color inaccuracies, and false contours, among others.

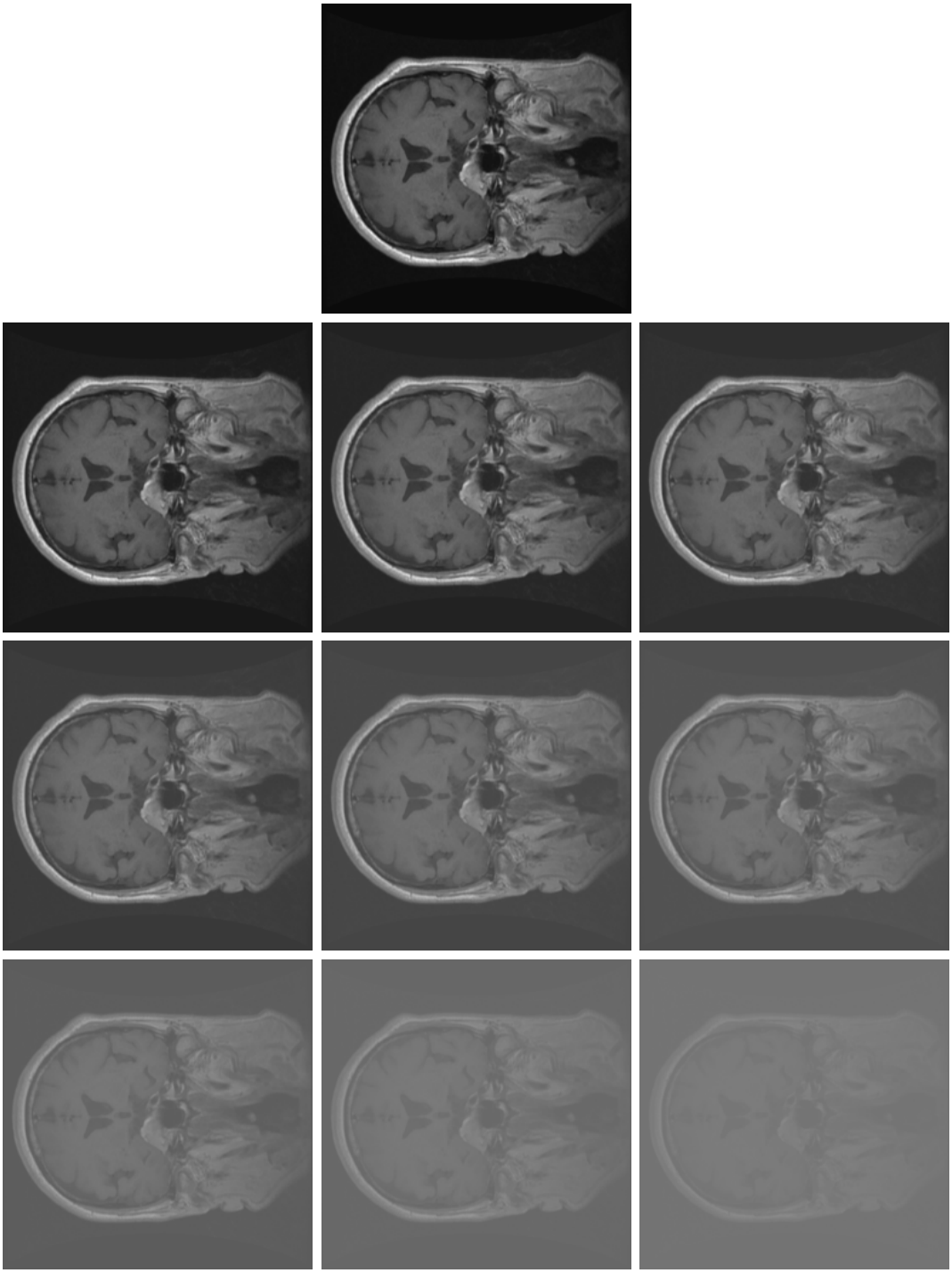


Figure 3.11: MRI with artificially applied contrast artifact with levels from 1 to 10, increasing from left to right and top to bottom.

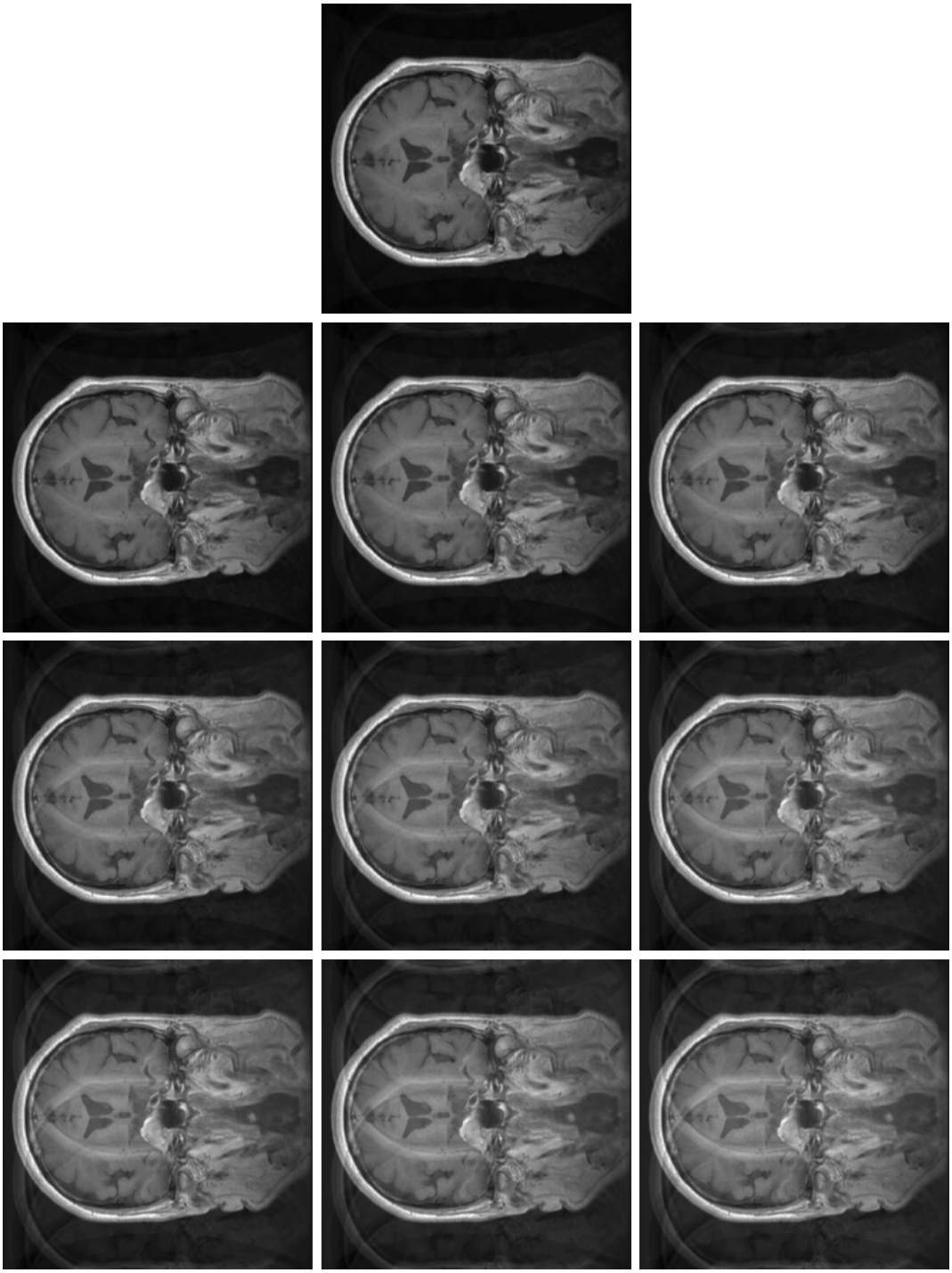


Figure 3.12: MRI with artificially applied ghosting artifact with levels from 1 to 10, increasing from left to right and top to bottom.

We generated JPEG compression artifacts using the Opencv imencode function, with which is possible to set the compression quality percentage. We set the intensity of the JPEG compression artifact as a function of the quality percentage, controlled by an *intensity* control parameter, as

$$\text{JPEGquality} = 100 - 9 \cdot intensity, \tag{3.12}$$

and generate examples of 10 compression intensities, shown in Fig. 3.13 , where the blockiness aspect can be identified, mostly in the bottom two images.

3.4 PERFORMANCE METRICS

Defining quality standards for medical content remains a non-trivial task, as the focus should be on the diagnostic value evaluated by the expert (RODRIGUES *et al.*, 2022). In most healthcare applications, end-user quality perception is likely to be strongly influenced by the clinical utility of the content, rather than strictly aesthetic criteria (RODRIGUES *et al.*, 2022). An indirect way to measure the diagnostic value of a set of images is to measure the accuracy of the machine learning models used to perform diagnostic predictions on those images.

The peak signal-to-noise ratio (PSNR) and the structural similarity index measure (SSIM) are the two most commonly used quality metrics for image restoration (ABDELHAMED *et al.*, 2018). Although both PSNR and mean SSIM (MSSIM) are good metrics for medical image quality, PSNR is not well correlated with human evaluations (RODRIGUES *et al.*, 2022; OSZUST, 2016). In this work, we focus on the MSSIM metric for image restoration quality measurement.

3.4.1 Structural Similarity Index Measure (SSIM)

The structural similarity index measure (SSIM), proposed by (WANG *et al.*, 2004), performs an image quality assessment based on the degradation of structural information by comparing two images. The method first computes local statistics in an 11x11 window, using a circular symmetric Gaussian weighting function so that pixels closer to the center of that window have a larger weight.

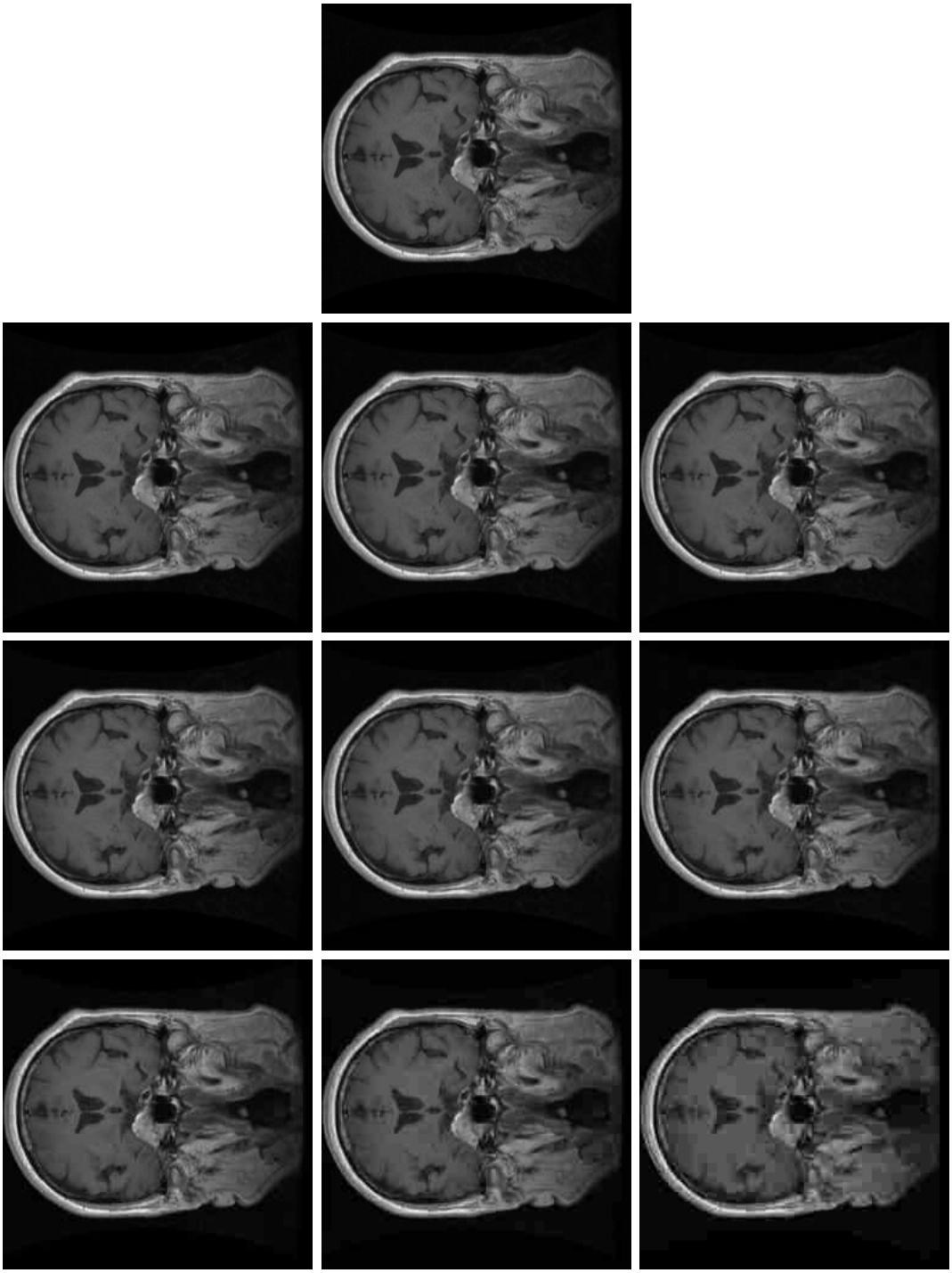


Figure 3.13: MRI with JPEG compression artifact with intensity levels from 1 to 10, increasing from left to right and top to bottom.

Given two non-negative image signals, x and y, the local statistics are the weighted average

$$\mu_x = \sum_i w_i x_i, \tag{3.13}$$

the weighted standard deviation

$$\sigma_x = \left(\sum_i w_i (x_i - \mu_x)^2 \right)^{1/2} \tag{3.14}$$

and the weighted covariance

$$\sigma_{xy} = \sum_i w_i (x_i - \mu_x)(y_i - \mu_y), \tag{3.15}$$

where w_i is the weight drawn from the circular symmetric Gaussian weighting function.

These statistics are used to perform three comparisons between the two signals. These three comparisons are: a luminance comparison defined as

$$l(x,y) = \frac{2\mu_x\mu_y + C_l}{\mu_x^2 + \mu_y^2 + C_l}, \tag{3.16}$$

a contrast comparison defined as

$$c(x,y) = \frac{2\sigma_x\sigma_y + C_c}{\sigma_x^2 + \sigma_y^2 + C_c}, \tag{3.17}$$

and the structure comparison defined as

$$s(x,y) = \frac{\sigma_{xy} + C_s}{\sigma_x\sigma_y + C_s}, \tag{3.18}$$

in which C_l, C_c and C_s are constants used to avoid denominators close to zero, and $\mu_x, \mu_y, \sigma_x, \sigma_y\sigma_{xy}$ are the statistic values calculated with equations 3.13, 3.14, 3.15 for signals x and y.

With an adjustment of the constant values and multiplying the three comparison values, the SSIM is obtained in a single function as

$$\text{SSIM}(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \tag{3.19}$$

in which C_1 and C_2 are constants used to avoid denominator too close to 0. C_1 is set to 0.01 and C_2 to 0.03.

To obtain the similarity of the entire images, a sliding-window method is applied, obtaining the SSIM value for every spatial position in the images. The final similarity between both images is obtained as the mean SSIM (MSSIM) value of the whole images as

$$\text{MSSIM}(im1,im2) = \frac{1}{M} \sum_i^M SSIM(im1[i],im2[i]), \tag{3.20}$$

in which $im1$ and $im2$ are the two images being compared, M is the total number of 11x11 windows in the images and $im1[i]$ and $im2[i]$ are the $i-th$ windows in images $im1$ and $im2$, respectively. In the survey by (RODRIGUES *et al.*, 2022), the SSIM metric had the best correlation with the evaluation of professionals in work using ultrasound and computed tomography (CT) images, in particular brain CT.

3.4.2 Accuracy

Accuracy may be defined as the ratio of correctly predictions out of all predictions as

$$Accuracy = \frac{CorrectPredictions}{Totalpredictions} \tag{3.21}$$

and is the most common performance metric for classification problems (ISLAM *et al.*, 2024). In this work, the accuracy metric is also used as an indirect performance metric for the restoration model. The better an image is restored, the better a classification model will perform on that image.

3.5 NEURAL NETWORK FUNDAMENTALS

In this section, we define the functions used in the design of neural networks that were used in the development of this work.

The basic function of a neural network is the perceptron(ROSENBLATT, 1958), with a perceptron layer being defined as

$$f(\mathbf{X}) = \varphi(\mathbf{W}^T\mathbf{X} + \mathbf{B}), \tag{3.22}$$

in which $\mathbf{X}$ is the input of the function, $\mathbf{W}$ are the layer weights, $\mathbf{B}$ are the bias terms and φ is a non-linear activation function. $\mathbf{W}$ and $\mathbf{B}$ are parameters that are adjusted during the training, or learning, procedure of the neural network, with an algorithm called the delta rule, in which the trainable parameters of the neural network are adjusted so that its output approaches the expected output from the training data.

A multilayer perceptron (MLP) can then be defined as a sequence of perceptrons as

$$f_n(\mathbf{X}) = \varphi(\mathbf{W}^T f_{n-1}(\mathbf{X}) + \mathbf{B}), \tag{3.23}$$

for a MLP with n perceptron layers, f_n being the n-th perceptron layer and other terms being equivalent to that of equation 3.22. The most common method of training neural networks with multiple layers is the backpropagation algorithm, in which the error propagates from the final layer of the network to earlier layers based on error gradient propagation (RUMELHART *et al.*, 1986a; RUMELHART *et al.*, 1986b). A common problem in training deep neural networks are the vanishing gradient, in which the scale of the gradients decreases at each layer, making model training particularly challenging for very deep neural networks (HOCHREITER, 1998).

One popular variation of the neural network for image tasks is the convolutional neural network (CNN)(FUKUSHIMA, 1980), which replaces the simple linear function of the perceptron for a convolution operation, as

$$f(I) = \sum_{c=0}^{C} \mathbf{w}_c * \mathbf{I}_c, \tag{3.24}$$

where $textbfI$ is the input with C channels, c is the channel index, $\mathbf{w}_c$ is the convolutional kernel for the c-th channel and f is the convolutional function. The convolutional layer is usually followed by a non-linear activation function.

While the perceptron layer generates sets of scalar outputs, the convolutional layer generates sets of n-dimensional arrays, usually two-dimensional for images. Since the convolution operator applies the same weights to every position of the input, the convolutional layer detects patterns with position invariance. A convolutional layer is usually defined by the size of the convolutional kernel (1x1, 3x3, etc.). Since the convolutional kernel is usually smaller than the input image, the layer usually only detects local patterns within the image, and several layers are required to recognize larger or global patterns.

3.5.1 Activation Functions

The main purpose of the activation functions in a neural network is to add nonlinearity to the model, enabling it to adapt to nonlinear problems. Activation functions have also been used to project values to constrained ranges, for example [0,1] or [-1,1].

3.5.1.1 Sigmoid and Hyperbolic Tangent

The sigmoid function (ROSENBLATT, 1958) is defined as

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{3.25}$$

which is a monotonic function, i.e. its values only increase, constrained to the range (0,1). The function may be seen as a smooth differentiable version of the step function and its output may be interpreted as a probability of a true value. The differentiable property of this function enables it to be used in a model trained with back-propagation.

An alternative, the hyperbolic tangent (LECUN *et al.*, 2015) function is defined as

$$tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1}, \tag{3.26}$$

having a similar S shape as the sigmoid function, but the $tanh$ is symmetric around the 0 value, being constrained to the range (-1,1) instead of (0,1).

3.5.1.2 Softmax

The softmax function was proposed as a normalized exponential function that turns a matrix of values into an array of probabilities (BRIDLE, 1990). The function defined by

$$softmax(x_k) = \frac{e^{x_k}}{\sum_i e^{x_i}}, \tag{3.27}$$

where return values are in the range 0-1 just like the sigmoid function. However, contrary to the sigmoid function, the output of each element in the Softmax function is dependent on the other values of the input array, with all the output elements adding up to 1. In this manner, the Softmax function treats each output as mutually exclusive, i.e. one and only one output is expected to be true and the Softmax function gives the probability of each input being the true value.

The Softmax function is often used at the output of networks in multiclass classification problems, in which only one class is expected to be the correct class. However, some works such as (VASWANI *et al.*, 2017) have used the function as an activation function in intermediate layers of deep neural networks, using it to generate a mask for a weighted average operation.

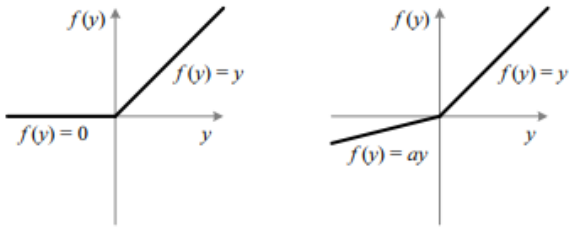


Figure 3.14: Graphical view of the ReLU (left) and leaky ReLU (right) functions, extracted from (HE *et al.*, 2015)

3.5.1.3 Rectifier Linear Activation Unit

The Rectifier Linear Activation Unit (ReLU) (NAIR; HINTON, 2010; JARRETT *et al.*, 2009; KRIZHEVSKY *et al.*, 2012; KRIZHEVSKY *et al.*, 2017) is an activation function defined as

$$f(x) = max(0,x). \tag{3.28}$$

In this function, only negative values are modified, set to zero. Since positive values are not affected by this function, it allows the gradient to propagate unchanged to earlier layers in the back-propagation algorithm. As a consequence, these functions make it easier and faster to train deeper neural networks.

One downside of the ReLU is that for negative values the gradient becomes zero. A generalization of the ReLU that deals with this problem is the leaky ReLU (MAAS *et al.*, 2013) defined as

$$f(x) = max(\alpha x,x), \tag{3.29}$$

where negative values are multiplied by a constant value α, usually set to 0.001, so that there is a non-zero derivative for negative values. For $\alpha = 0$, the function becomes equivalent to the ReLU function.

3.5.1.4 Non-Monotonic activation functions

While most previously mentioned functions are monotonic, i.e. have only increasing or decreasing values, more recent works have shown that non-monotonic activation functions achieve better network performance in various tasks, including natural language processing and computer vision.

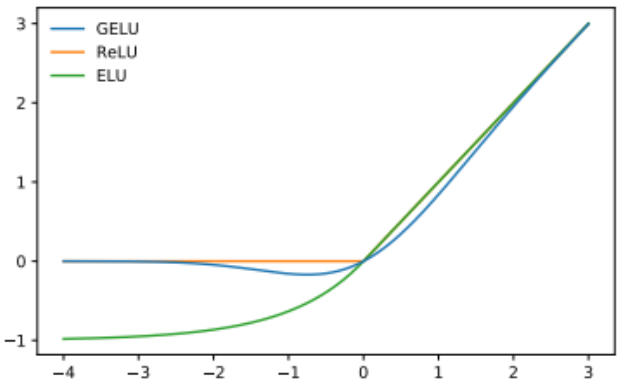


Figure 3.15: Visual comparison of the ReLU, ELU and GELU activation functions, extracted from (HENDRYCKS; GIMPEL, 2016).

The Gaussian Error Linear Units (GELUs)(HENDRYCKS; GIMPEL, 2016) is a non-saturating, non-monotonic activation function defined by

$$GELU(x) = x\Phi(x), \tag{3.30}$$

where $\Phi(x)$ is the standard Gaussian distribution function. The GELU activation function may be approximated by

$$GELU(x) = 0.5x(1 + tanh(\sqrt{2/\pi}(x + 0.044715x^3))) \tag{3.31}$$

and can be seen as a smoothed version of ReLU, having the same two asymptotes. A visual comparison of the ReLU and GELU functions can be seen in Figure 3.15.

Another popular non-monotonic, non-saturating activation function is the Sigmoid Linear Unit (SiLU), also referred to as swish (RAMACHANDRAN *et al.*, 2017) defined by

$$SiLU(x) = x\sigma(x) \tag{3.32}$$

where $\sigma(x)$ is the sigmoid function, though it is argued that this function has poorer performance compared to GELU(HENDRYCKS; GIMPEL, 2016). These non-monotonic activation functions were shown to have consistent improved performance over the variations of the Rectifier Linear Units and have been adopted in several modern neural networks.

3.5.2 Normalization

Normalization for neural networks usually refers to the adjustment of data to have 0 mean and unit standard deviation, that is, a standard deviation equal to 1. This is defined in equation

form as

$$x' = \frac{x - \mu}{\sigma}, \tag{3.33}$$

where x is the input data, μ is its mean value and σ is its standard deviation. It is common for images to be normalized before being passed as input to a neural network, taking into account the mean and variance of the entire dataset or popular datasets such as the ImageNet dataset (DENG *et al.*, 2009).

3.5.2.1 Batch Normalization

As normalization of the images helped improve image recognition, normalizing the output of intermediate layers of deep neural networks with a layer called batch normalization (BN) results in faster training and more accurate models (IOFFE; SZEGEDY, 2015). Batch normalization normalizes data following the equation

$$y = \frac{x - E[x]}{\sqrt{Var[x] + \epsilon}} \cdot \gamma + \beta, \tag{3.34}$$

where ϵ is a small value to avoid division by 0, γ and β are learnable parameters initialized with 1 and 0, respectively, $E[x]$ is the channels-wise mean value of all elements in a batch and $Var[x]$ is the variance of those values.

In this context, batch refers to the subset of data used in each training iteration. When this function is set to inference or evaluation mode, $E[x]$ and $Var[x]$ are replaced by the running mean and variance instead of the batch mean and variance, so it will work for individual images. Note that the subtraction of the mean value nullifies the effect that the bias would have in the previous linear layers. For this reason, it is standard practice to remove the bias term from linear layers when training with normalization layers, though the β term replaces the effect of bias.

3.5.2.2 Sample Normalization

One problem with batch normalization is its increased error when training with small batches, e.g., with fewer than 16 samples, as shown by (WU; HE, 2018). To avoid this problem, alternative normalization layers have been proposed in which the mean and standard deviation

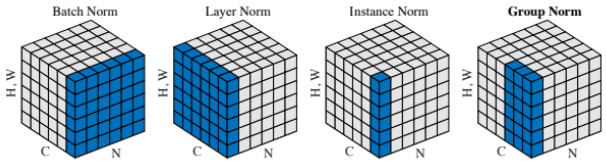


Figure 3.16: Visual comparison of different normalization layers based on the subset of data used. Taken from (WU; HE, 2018)

are calculated per sample instead of per batch, which work better for smaller batch sizes.

Like batch normalization, the instance normalization (IN) layer proposed by (ULYANOV *et al.*, 2016) calculates the mean and standard deviation per channel, but for each individual sample instead of the whole batch. The layer normalization (LN) function defined by (BA *et al.*, 2016) calculates the mean and standard deviation for all channels in a layer, separated by sample. As a generalization between these two approaches, (WU; HE, 2018) proposed the group normalization (GN) layer, in which the mean and standard deviation are calculated per channel group. Figure 3.16 provides a visual comparison between BN, IN, LN and GN.

One difference between batch normalization and sample normalization layers is the fact that sample normalization layers are the same at training and test time. No running mean and standard deviation are calculated during training to be used at test time, instead those values are calculated for each test sample.

3.5.3 Down-sampling and Up-sampling

Convolutional layers are responsible for processing images or feature maps with a spatial configuration denoted by W×H. The computational load of these layers is directly proportional to this configuration size, whereas the convolutional kernel processes only a limited portion of these matrices. The term down-sampling encompasses operations aimed at diminishing the spatial dimensions of these matrices, which can be generally expressed as

$$O_{x,y} = F(I_{k_1 x, k_2 y}) \tag{3.35}$$

, where $O_{x,y}$ represents the output at the spatial position (x,y), F denotes the downsampling function, I signifies the input plane, and (k_1,k_2) is the downsampling factor, commonly known as stride.

The simplest downsampling operation is the identity function, simply skipping data and outputting every k-th value (HE *et al.*, 2016). Other common operations are the average pooling (HUANG *et al.*, 2017) and Max Pooling (KRIZHEVSKY *et al.*, 2012), which, respectively, take the mean and maximum value in a region around the k-th value of the input. Another option, the stride convolution(HE *et al.*, 2016) is a convolutional layer that skips the input positions.

After downsampling, the receptive field of a convolutional kernel will be a larger percentage of the input plane and consequently process higher-level features, closer to the global features of an image. The reduced spatial size of downsampled planes also reduce the computational cost of following layers; for this reason, it is common to increase the number of planes after down-sampling, or rather down-sampling is applied before increasing the number of channels to avoid excessive computational cost.

Upsampling serves as the inverse operation to downsampling, functioning to augment the spatial dimensions of feature maps. This technique is employed when the production of a higher resolution output, characterized by more intricate details, is necessary. Techniques for upsampling encompass interpolation methods and transposed convolution, also known as the deconvolutional layer (ZEILER *et al.*, 2010).

3.5.4 Residual Connections

The residual function was proposed as the basic building block of deep neural networks by (HE *et al.*, 2016), defining the residual function as

$$f(x) = R(x) + x, \tag{3.36}$$

where $R(x)$ is a residual function, containing the linear layers and activation functions of the neural network, which is added to the input of the function. The addition of the input, x, is also referred to as skip connection or main branch, while the residual function may be referred to as residual branch.

One advantage of the residual function is that the skip connection allows for the gradient to flow more easily to earlier layers, allowing the training of even extremely deep neural networks with over a thousand layers, although the use of residual functions has also been shown to

improve the performance of models with fewer than 10 layers (ZAGORUYKO; KOMODAKIS, 2016).

When the residual branch down-samples the feature maps or changes the number of output channels, the residual function is changed to

$$f(x) = R(x) + I(x), \tag{3.37}$$

where $I(x)$ is a function, known as identity mapping, parameterized or not, that changes its input to the same shape as the output of the residual branch. The identity mapping has fewer layers and parameters than the residual branch, so that the gradient is still less affected by this branch than by the residual branch. Common identity mapping functions are average pooling with channel zero padding and strided convolution. However, some works have preferred to completely remove the skip connection when the output of the residual branch does not have the same shape as its input (TAN; LE, 2019).

3.5.5 Attention mechanisms

Attention may be understood as the mechanism in which part of the information is selected to be processed, while the rest of the information is ignored. In machine learning, particularly deep learning, this can be done by assigning weights in the range [0,1] to each input value. If an input is multiplied by a weight close to 0, that input has little to no effect on the output of the layer, so that this input is ignored and consequently the layer *pays attention* to the other input values that were assigned larger weights. Two attention mechanisms used in computer vision are the scaled dots product attention used by transformer models (VASWANI *et al.*, 2017) and the squeeze-and-excitation layer (HU *et al.*, 2018).

3.5.5.1 Squeeze-and-Excitation layer

Hu *et al.* introduced a channel self-attention mechanism referred to as a squeeze and excitation (SE) layer (HU *et al.*, 2018). This component, depicted in Figure 3.17, is characterized by the function

$$SE(x) = \sigma(ff(GAP(x))) \cdot x, \tag{3.38}$$

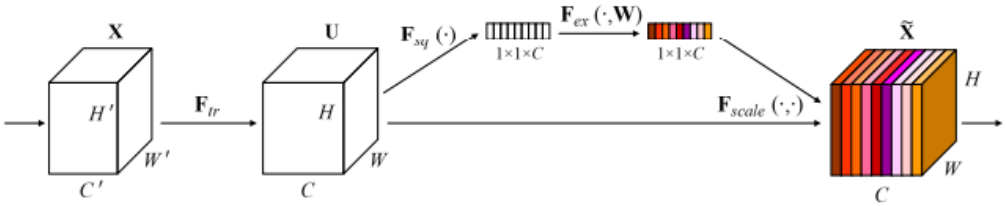


Figure 3.17: Squeeze-and-Excitation layer. Image taken from (HU *et al.*, 2018).

where GAP denotes the global average pooling function, which calculates the channel-wise average. The sigmoid function is denoted by σ, and ff represents a feed-forward neural network comprising two linear layers interposed by a ReLU activation function. A critical parameter for the SE layer is the number of hidden units within the feed-forward neural network, which is typically set to one-quarter of the input channels.

The sigmoid function makes it so that the soft weights applied to each channel are in the range [0,1]. Channels multiplied by values close to 0 are disabled, and therefore the network will pay attention to the features of the channels that were not disabled. Since the layer uses the average value of each channel, instead of the entire channel, this layer has significantly less computational cost than a convolutional layer. The GAP function also helps include global information in convolutional layers that only process local features.

3.5.5.2 Transformer

The transformer model proposed by (VASWANI *et al.*, 2017) has become a new standard in artificial intelligence. The model, initially proposed for language translation, is based on an attention mechanism that processes token vectors associated with each words. The model has since been adapted to other tasks, including image recognition and restoration.

The transformer attention function, known as scaled dot product attention, shown in Figure 3.18, takes as input three matrices identified as query (Q), key (K) and value (V), in which each matrix row is a vector associated with one token. Each row in the key and value matrices is associated with the same token. Each query vector may come from a different source than the key and value vectors, but if it comes from the same source, the attention function becomes a self-attention function.

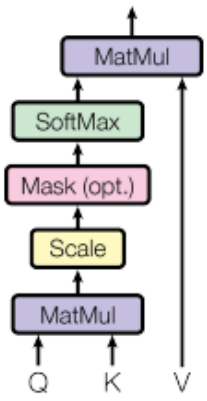


Figure 3.18: Scaled dot product attention. Image taken from (VASWANI *et al.*, 2017).

The scaled dot-product attention is defined by

$$e_{ij} = \frac{< q_i, k_j >}{\sqrt{d_k}}, \tag{3.39}$$

where q_i is the i-th query vector, k_j is the j-th key vector, d_k is the length of each vector and e_{ij} may be referred to as energy between query i and key j, following the nomenclature used in (BAHDANAU *et al.*, 2014). When the direction of the vectors q and k is more similar, the energy value will be greater. One may then interpret this as the key vector being a descriptor for each token, the query vector being an information being searched (queried) and that the energy indicates how much of a match the key token is to the queried information.

The Softmax function is then used to adjust the energy values to the [0,1] range, as

$$p_{ij} = \frac{exp(e_{ij})}{\sum_m exp(e_{im})}, \tag{3.40}$$

where p_{ij} could be interpreted as the relative level of importance that the j-th key token has to the i-th query token. The term d_k was added to the equation 3.39 to avoid large values that would result in extremely low gradients in the softmax function, thus improving training.

The final output for each query vector is then defined by

$$c_i = \sum_j p_{ij} \cdot v_j, \tag{3.41}$$

where c_j is a vector with the same length as the v_j vector, and is equivalent to a weighted average of the value vectors, giving larger weights to the tokens that have more importance to the i-th token.

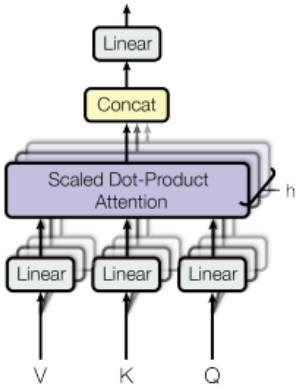


Figure 3.19: Multi-head self attention. Image taken from (VASWANI *et al.*, 2017).

This process may be implemented for all query-key pairs using optimized matrix operations as

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V. \tag{3.42}$$

Instead of performing a single attention function in each block, (VASWANI *et al.*, 2017) proposes to project linearly the h-times Q, K, V vectors and perform h-times attention functions in parallel, shown in Figure 3.19. Each parallel attention unit is referred to as a head and the output of each head is defined as

$$head_i = Attention(QW_{qi},KW_{ki},VW_{vi}). \tag{3.43}$$

The output of each head is then concatenated and linearly projected into a single output, such that the output of the multi-head attention unit is defined by

$$MultiHeadAttention(Q,K,V) = concat(head_1,..,head_h)W_o. \tag{3.44}$$

Subsequently, the multi-head attention mechanism is encompassed within a residual block, and each residual block is followed by a layer normalization operation. In addition to the attention component, the transformer architecture incorporates a feed-forward residual block. Consequently, the transformer encoder is composed of alternating feed-forward and multi-head self-attention blocks. The attention mechanism inherent in the transformer model processes all input tokens; therefore, a positional encoding must be appended to each token to ensure the model considers the positional context of each word within a sentence.

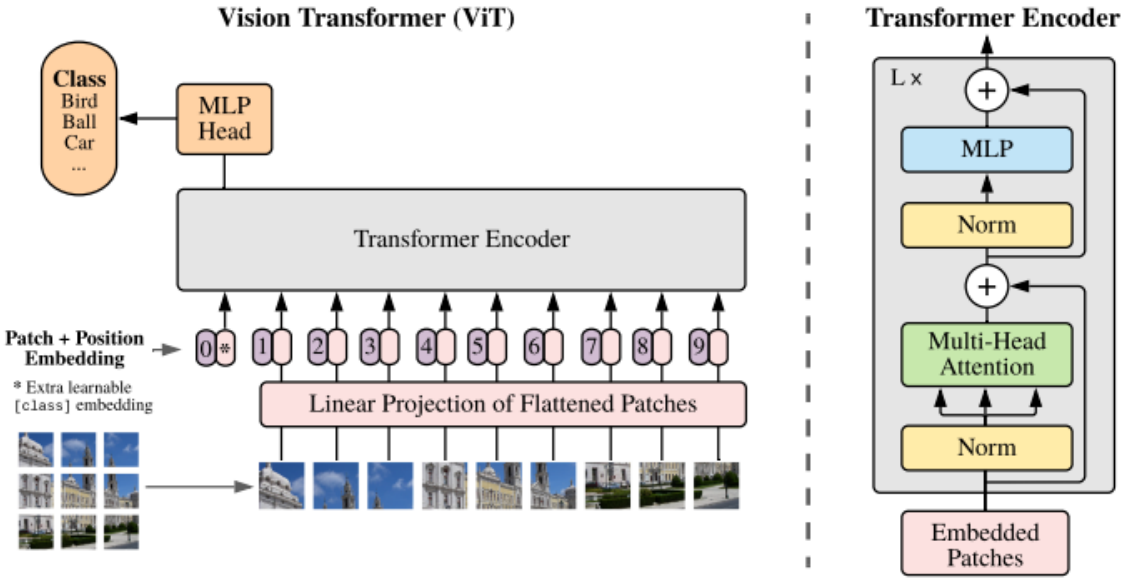


Figure 3.20: Vision transformer (ViT) architecture. Image taken from (DOSOVITSKIY *et al.*, 2020).

3.5.5.3 Vision Transformer

The original transformer model (VASWANI *et al.*, 2017) was proposed for language tasks, but (DOSOVITSKIY *et al.*, 2020) adapted the model for image classification, defining the Vision Transformer (ViT), shown in Figure 3.20, surpassing CNN performance and setting a new state of the art in the ImageNet dataset.

To adapt images to the transformer model, each image is split into 16x16 patches and each patch is then projected into a vector token. This can be done with a 16x16 convolution with stride of 16. In addition to these input tokens, one additional token is added, the classification token, with learnable parameters. After the transformer encoder processes all tokens, the classification token is then extracted, and used as input features in a MLP to perform the classification.

The vision transformer only performs self attention, meaning the query, key and value vectors are all projections from the input tokens. For image classification, the vision transformer only has the encoder part of the transformer. The transformer blocks were slightly adapted, mainly by the normalization layer placement. The layers in the vision transformer encoder are defined by two residual blocks

$$x'_i = x_{i-1} + MHSA(LN(x_{i-1})) \tag{3.45}$$

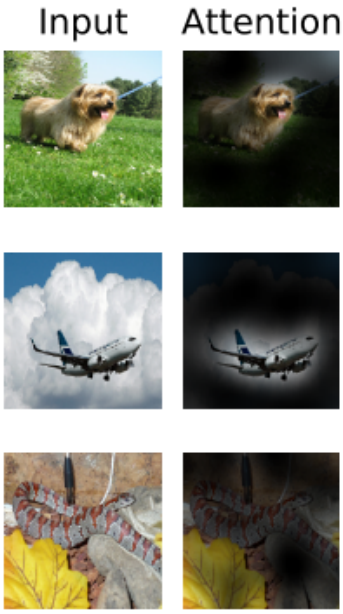


Figure 3.21: Sample representations of attention from the output token on the input space from a ViT model. Image taken from (DOSOVITSKIY *et al.*, 2020).

and

$$x_i = x'_i + MLP(LN(x'_i)), \tag{3.46}$$

where x_i are the tokens at the i-th residual block pair, $MHSA$ is the Multi-head self-attention function, MLP is a multi-layer perceptron and LN is the Layer Normalization function (BA *et al.*, 2016). As a standard, the linear projections increase the number of channels of x_i, C, to 4C within each residual function. It is possible to notice a similarity between the residual MHSA and the ResNext block (XIE *et al.*, 2017), in which spatial operations are performed in groups operations, in the middle of two point-wise linear projections. The attention function may be projected on the input as a highlight of the relevant regions of the image, as shown in Figure 3.21.

Several works improved the vision transformer design by adopting common CNN design practices. One such adaptation is to divide the vision transformer into stages in which each stage reduces the resolution of the previous stage while simultaneously increasing the number of channels, similar to the hierarchical structure of CNNs (HEO *et al.*, 2021; WANG *et al.*, 2021; GRAHAM *et al.*, 2021), as shown in Figure 3.22. This adaptation is shown to obtain better model accuracy and throughput (images/second), compared to the single-stage transformer. (HEO *et al.*, 2021) argues that this hierarchical structure with stages with different scales

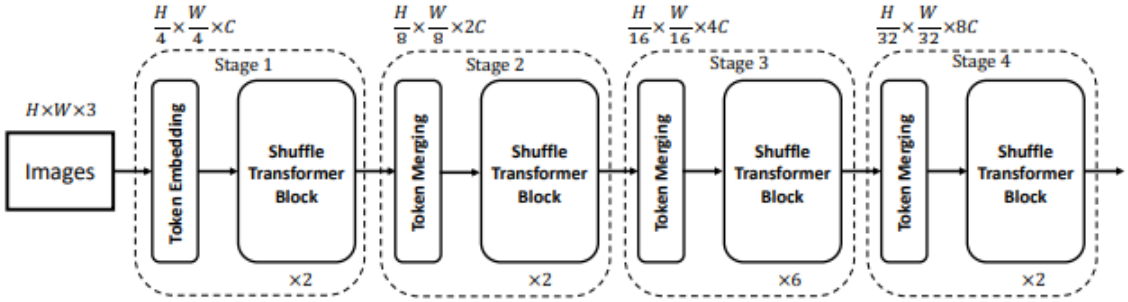


Figure 3.22: Hierarchical transformer architecture with shifted windows. Image taken from (HUANG *et al.*, 2021)

helps to improve the generalization of the model, as the original ViT does not improve the validation performance when training performance increases. (WANG *et al.*, 2021) shows that the multistage structure, enables the model to be trained task besides classification, such as object detection and image segmentation.

Another modification of ViT to make it more similar to CNNs is to remove the classification token and instead perform an average pool of tokens at the end of the encoder to generate the classification features (GRAHAM *et al.*, 2021). One processing bottleneck of the transformer architecture is the fact that matrix multiplications in the attention layers have computational cost quadratically proportionate to the number of input tokens (the input resolution), so that the computational cost of the multi-head self-attention function is

$$cost(MHSA) = C(WH)^2, \tag{3.47}$$

where C is the number of channels in the attention function, and (WH) are the number of tokens, associated with the input resolution, (W,H), in which W is the input width and H is the input height.

An approach to reduce this computational cost in the attention function is to split the input channels into windows and apply the attention function within each window, instead of the entire input (LIU *et al.*, 2021; CHU *et al.*, 2021), as shown in Figure 3.23. With this window splitting approach, the computational cost of the window multi-head self-attention becomes

$$cost(WMHSA) = C(win_w win_h)^2 \frac{W}{win_w} \frac{H}{win_h}, \tag{3.48}$$

where win_w and win_h are the windows width and height, respectively, and $\frac{WH}{win_w win_h}$ is the total number of windows in the input. In this case, when the resolution increases the number of

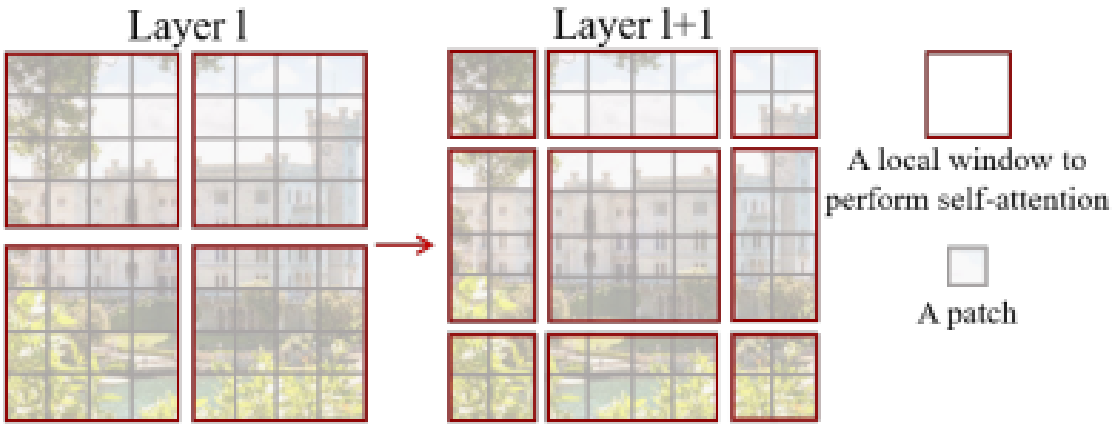


Figure 3.23: Feature maps split in windows for window-attention function. Image taken from (LIU *et al.*, 2021)

windows increases and the computational cost increases only linearly. With this adaptation, however, the self-attention function no longer performs global attention, instead local attention is performed.

An alternative approach to the vision transformer architecture involves the integration of convolutional layers within the transformer framework, resulting in a hybrid transformer-CNN architecture. Through a series of experimental evaluations, (DAI *et al.*, 2021b; RAMACHANDRAN *et al.*, 2019; GRAHAM *et al.*, 2021) demonstrated that deploying convolutional blocks in the initial stages of the network, followed by transformer blocks in the latter stages, provides superior performance compared to any configuration ranging from entirely convolutional to entirely transformer-based blocks. Unlike CNNs, vision transformers inherently lack an inductive bias to concentrate on spatially proximal elements; incorporating convolutions in the architecture introduces such a bias (GRAHAM *et al.*, 2021).

3.5.6 Autoencoder

An autoencoder (AE) is a neural network that is used to reconstruct input data (LI *et al.*, 2023). Since it doesn't require labels it is usually considered an unsupervised training model and method. An autoencoder is typically split into two parts, the encoder and the decoder. The encoder part learns the features of the input data and transfers them to another data space with a meaningful representation. The decoder part of the autoencoder restores the data from

this representative space to the original data space, restoring the image to its original form.

The denoising autoencoder (DAE) (VINCENT *et al.*, 2008) is a variation of the conventional autoencoder in which noise is added to the input data, but its target output is the original input data without the added noise. The learning objective of these models is to reconstruct the original input from the contaminated input.

3.5.7 U-shaped networks

CNNs used for classification tasks typically apply successive down-sampling layers, lowering feature resolution until the final label output. However, for several visual tasks, such as segmenting biomedical images, the output needs to have a resolution similar to that of the input image. One popular model that generates this output is the proposed UNet model (RONNEBERGER *et al.*, 2015) for biomedical image segmentation.

The UNet architecture consists of a contracting path and a corresponding expansive path. Each stage of the contracting path ends with a downsampling layer, while each stage of the expansive layers starts with an upsampling layer, so that each contracting stage has a corresponding expansive stage with similar resolution. In addition to those stages there is a bottleneck stage in between the contracting and expansive paths. The visual representation of this network resembles an U shape, as shown in Fig. 3.24.

When features undergo downsampling, subsequent convolutional layers achieve an increased receptive field relative to the input image, owing to the reduced resolution, thus enhancing contextual feature representation. However, this reduction in resolution compromises the precision of feature localization. To attain both comprehensive contextual representation and precise feature localization, each expansive stage initiates by concatenating the high-context upsampled feature with the high-resolution output of its corresponding contracting stage. Consequently, the expansive path benefits from precise feature localization derived from the contracting path, while preserving contextual richness from the upsampled features.

The design of UNet inspired the design of several other neural network architectures for tasks with high-resolution output, including biomedical image segmentation (ZHOU *et al.*, 2018; JHA *et al.*, 2019; TOMAR *et al.*, 2022) and image restoration (MUCKLEY *et al.*, 2021;

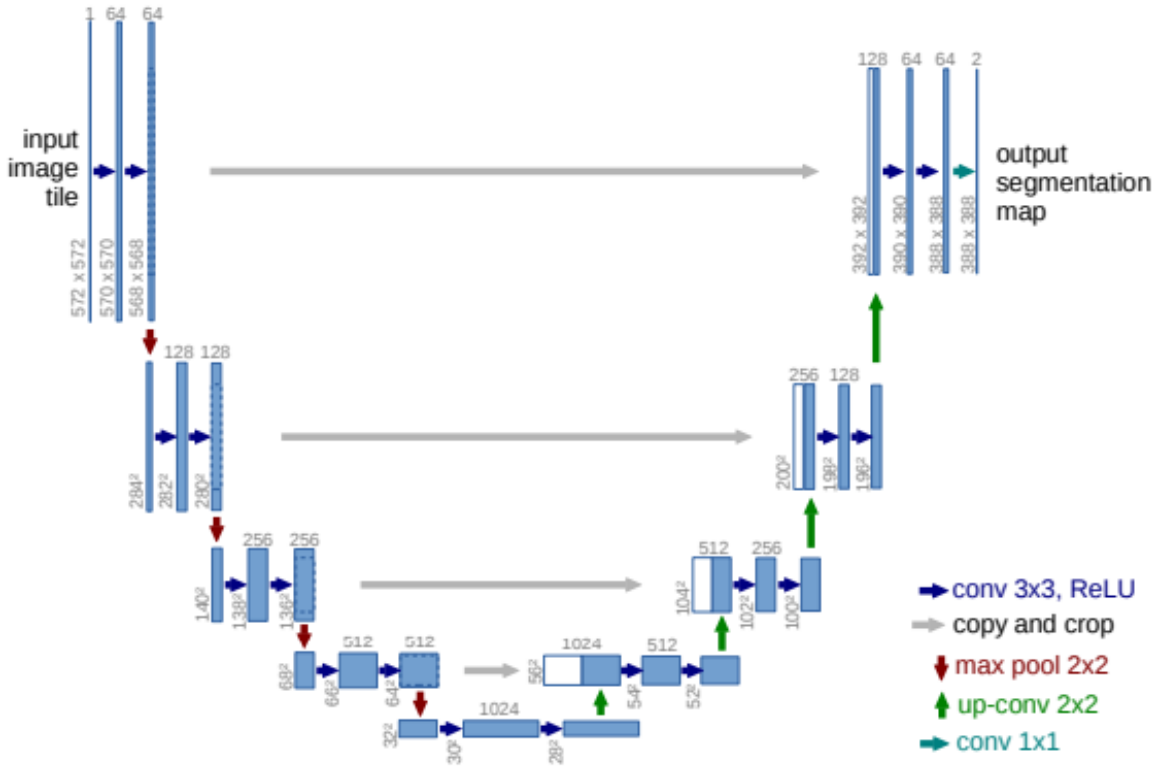


Figure 3.24: UNet architecture for biomedical image segmentation. Image taken from (RONNEBERGER *et al.*, 2015).

WANG *et al.*, 2022; ZAMIR *et al.*, 2022).

3.6 MODELS TRAINING

Machine learning models have two types of parameters, the model parameters that can be initialized and updated through the learning process, and the hyperparameters that have to be defined before the learning process and are used to configure the model or specify the learning algorithm (YANG; SHAMI, 2020). The model configuration hyperparameters are the ones that define the model architecture, such as the number, size, and type of layers, e.g., the layer types described in Section 3.5 and its dimensions. The other type of hyperparameter specifies the learning procedure, also known as model training or fitting, in which other aspects are defined such as the training method and the learning rate value. In this section, we present the fundamentals for training a deep learning model with the gradient descent algorithm (RUMELHART *et al.*, 1986b).

The training procedure of a neural network model with backpropagation typically can be

done as a sequence of steps:

1. Initialization of parameters: Initial weights of the model are defined.
2. Data sampling: A subset of the training data is selected to be used as input data for the model.
3. Data preprocessing: performs modifications to the input data and sometimes to the target expected output as well.
4. Forward pass: The model predicts the output for the input data given.
5. Loss estimation: the difference between the predicted and expected output is measured on the basis of a loss function.
6. Backpropagation: The loss is propagated to all layers of the model through the backpropagation algorithm and the gradients at each parameter are estimated.
7. Weight update: The network parameters are updated on the basis of the gradients and an optimization algorithm.
8. End condition: If the total number of training iterations or epochs, or another stop condition was reached, the training stops. Otherwise, the gradients at each layer are zeroed and the training procedure returns to step 2.

Common problems in training a neural network are underfitting, overfitting, exploding, and vanishing gradient problems. The problems of vanishing/exploding gradients are associated with unstable gradients during back-propagation in multilayer models, as described in Section 3.5. Underfitting refers to the problem when the proposed model is not capable of capturing the variability of the data or the training was not long enough to properly fit the model to the data (JABBAR; KHAN, 2015; POTHUGANTI, 2018).

Overfitting refers to the problem in machine learning when a model does not generalize well to unseen data, i.e. the model performance on new unseen data is worse than on data from the training set (YING, 2019; JABBAR; KHAN, 2015; POTHUGANTI, 2018). Common solutions to the overfitting problem include early stopping, network reduction, regularization, and dataset expansion. In early stopping, the training procedures stop once the validation accuracy stops

improving, before the total defined training iterations. Network reduction works by removing less relevant data from the model and reducing its complexity. Dataset expansion consists of increasing the number of training samples, which can be done by acquiring more data, creating more data, or creating modified versions of existing data, known as data augmentation. The larger the model, the more data that are required for that model to be effective (KOLESNIKOV *et al.*, 2020).

Common aspects of the training procedure that warrant some attention are: parameter initialization, data augmentation, regularization, optimizers or optimization algorithm, loss functions and learning rate scheduling.

3.6.1 Parameters initialization

The final parameters of a neural network at the end of training depend on the initial values of the neural network. Linear layers are usually initialized with random values; however, some studies have shown that specific distributions of these values can lead to more stable training and a more accurate final model. Another method of initializing the values of model parameters is by copying the parameters of another equivalent model trained on a different dataset, in a process known as transfer learning.

3.6.1.1 Normalized Initialization

Since the exploding and vanishing gradient problems (HOCHREITER, 1998) occur due to a change in the scale of gradients during back-propagation, these problems could be avoided if the gradient calculations in each layer did not change the scale of the error being back-propagated. (GLOROT; BENGIO, 2010) proposed to achieve this by initializing weights so that the output of each linear layer would have zero mean and unit variance, which could be referred to as normalized initialization. (HE *et al.*, 2015) adapted the initialization method to take into account the effect the ReLU activation function would have on the gradient, enabling stable training of deeper CNN models. Weights are initialized with random values sampled

from a normal distribution, $\mathcal{N}(0, \sigma^2)$ where

$$\sigma = \frac{af\_gain}{\sqrt{W/O}}, \tag{3.49}$$

where W is the number of weights in the layer, O is the number of output planes for that layer and af_gain is a gain related to activation function after that layer, usually $\sqrt{2}$ for the ReLU function. A similar process is used when initializing weights with different distributions, such as the uniform distribution and different activation functions.

3.6.1.2 Transfer Learning

Training deep learning models with strong performance requires a large amount of data and compute, which may be prohibitive for a specific deep learning task (KOLESNIKOV *et al.*, 2020). A solution to this problem, known as transfer learning, is to first train a model on a large generic dataset and then use its weights to initialize models for subsequent task, with reduced data and computation. With transfer learning, only task-specific layers of the network are replaced and initialized with a random distribution, e.g. the classification layer of a network is replaced with a layer with the correct number of target classes. Models pre-trained on large datasets are often available online and can be used as model initialization for new tasks.

3.6.2 Regularization

Regularization aims to limit the influence of useless features in the training process, usually by adding a penalty term to the training process. Such methods include regularization of the L2 norm and the dropout layer.

3.6.2.1 Weight Decay

The L2 norm regularization, also known as weight decay regularization in deep learning papers, uses the Euclidean distance as the penalty term (HANSON; PRATT, 1988; GOODFELLOW *et al.*, 2016; YING, 2019; ANDRIUSHCHENKO *et al.*, 2023) defined as

$$\Omega(\theta) = \|\theta\|_2 = \sqrt{\Sigma_i \theta_i^2}, \tag{3.50}$$

where θ are the model parameters. Based on the Pytorch implementation (PASZKE *et al.*, 2017), weight decay can be introduced in the weight update from equation **??** with a weight decay parameter λ as

$$\theta^k = \theta^{k-1} - \eta(\nabla_\theta J(\theta) + \lambda\Omega(\theta)). \tag{3.51}$$

3.6.2.2 Dropout

Dropout is a method to prevent overfitting in which units in the network are randomly dropped (SRIVASTAVA *et al.*, 2014), or alternatively, their value is set to 0 (zero). This method aims to prevent units from co-adapting. The dropout has a single parameter p that indicates the probability that a unit is dropped during the test time. Variations of the dropout include the spatial dropout (TOMPSON *et al.*, 2015), which drops an entire plane, and the drop-block (GHIASI *et al.*, 2018), which drops a region within the feature map.

3.6.3 Learning Rate Scheduling

The learning rate is usually considered the main hyperparameter in neural network training. Instead of defining a fixed learning rate, a more effective practice is to adjust the learning rate throughout the training procedure, usually starting at a large value and decreasing through the training. This process is referred to as the learning rate schedule.

Although standard practice starts with a high learning rate, deeper models might struggle to stabilize training at high learning rates. For this reason (HE *et al.*, 2016) proposed to linearly increase the learning rate from a very low value to the maximum value during initial iterations, helping stabilize training. This process is referred to as learning rate warm-up.

Although the standard practice was to decay the learning rate by multiplying it by a constant value (< 1), called step learning rate decay; (LOSHCHILOV; HUTTER, 2016) showed that decay of the learning rate according to the cosine function led to better performance. (HE *et al.*, 2019) proposed a simplified version of this cosine decay as

$$\eta_t = \eta_{min} + \frac{1}{2}\left(1 + cos\left(\frac{t\pi}{T}\right)\right)(\eta_{max} - \eta_{min}), \tag{3.52}$$

where t is the iteration number, T is the total number of iterations, η_{max} and η_{min} are the

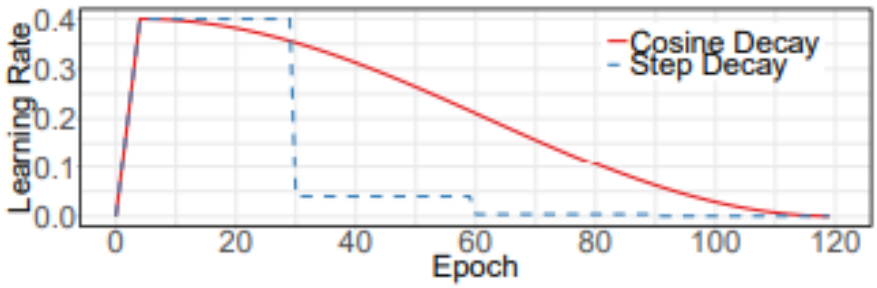


Figure 3.25: Learning rate schedule with warmup and cosine decay. Image taken from (HE *et al.*, 2019)

maximum and minimum learning rate, respectively, and η_t is the learning rate at iteration t.

The complete learning rate schedule then consists of a warm-up stage followed by cosine decay, as exemplified in Figure 3.25. In this case, the variable t and parameter T in equation 3.52 only consider the iterations after the warm-up is done.

As an alternative to the decrease in learning rate, (SMITH *et al.*, 2017) proposed to increase the batch size instead to a similar effect. Based on this work, it should be noted that learning rate should be proportionate to the batch size.

3.6.4 Optimizers

Optimizers refers to the algorithms used to update the parameters of deep neural networks. Most of these algorithms use the backpropagation algorithm to obtain the error gradient of each network parameter and use these values to update the neural network parameters. Two of those algorithms available in the Pytorch framework are the stochastic gradient descent (RUDER, 2016) and the AdamW (LOSHCHILOV, 2017) optimizers.

3.6.4.1 Stochastic Gradient Descent

The gradient descent algorithm updates the parameters based on the gradient computed from the training data. Three different gradient descent algorithms may be defined based on how much data is used in each update step (RUDER, 2016). Batch gradient descent uses the entire training set at each iteration, stochastic gradient descent uses a single sample, and mini-batch gradient descent uses an arbitrary number of samples at each training step. The Pytorch

framework, however, uses the term stochastic gradient descent (SGD) to refer to this algorithm in a more general manner, independently of the amount of data used in each iteration. The basic SGD algorithm is defined by

$$\theta_t = \theta_{t-1} - \eta \nabla_\theta J(\theta_{t-1}, x^{(i:i+n)}, y^{(i:i+n)}), \tag{3.53}$$

where θ_t are the parameters at iteration t, η is the learning rate hyperparameter and $\nabla J(\theta, x^{(i:i+n}, y^{(i:i+n)})$ are the gradients calculated with training samples i to $i+n$ and the parameters from iteration $t-1$.

To accelerate the SGD in relevant directions and dampening oscillations, the SGD algorithm can be improved by a momentum term (QIAN, 1999; RUDER, 2016) and the algorithm is updated to

$$\begin{aligned} v_t &= \gamma v_{t-1} + \eta \nabla_\theta J(\theta) \\ \theta_t &= \theta_{t-1} - v_t, \end{aligned} \tag{3.54}$$

where v_t is a velocity variable, γ is the momentum hyperparameter, usually set to 0.9 or similar values, and the remaining algorithm is equivalent to equation 3.53. The momentum SGD algorithm can be further improved as the Nesterov accelerated gradient (NAG) (SUTSKEVER *et al.*, 2013; RUDER, 2016) algorithm defined as

$$\begin{aligned} v_t &= \gamma v_{t-1} + \eta \nabla_\theta J(\theta - \gamma v_{t-1}) \\ \theta_t &= \theta_{t-1} - v_t, \end{aligned} \tag{3.55}$$

where the gradient is calculated based on the predicted value of the parameter instead of the current value.

3.6.4.2 AdamW

(KINGMA, 2014) proposed Adam, an optimizer based on adaptive estimates of lower-order moments. Like SGD, it starts by calculating the gradients during backpropagation at iteration t as

$$g_t = \nabla_\theta J(\theta_{t-1}). \tag{3.56}$$

The adam optimizer then computes the bias-corrected first moment estimate, m'_t, and the

bias-corrected second raw moment estimate, v'_t,

$$\begin{aligned} m_t &= \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t \\ v_t &= \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2 \\ m'_t &= m_t / (1 - \beta_1^t) \\ v'_t &= v_t / (1 - \beta_2^t), \end{aligned} \tag{3.57}$$

where β_1 and β_2 are momentum parameters, usually set to 0.9 and 0.999, respectively. With the moments estimates, the parameters are then updated by

$$\theta_t = \theta_{t-1} - \alpha \cdot m'_t / (\sqrt{v'_t} + \epsilon), \tag{3.58}$$

where α is the learning rate, usually set to 0.001 and ϵ is a constant included to avoid division by zero, usually set to e^{-8}.

In the Adam optimizer, the weight decay regularization would usually be incorporated in equation 3.56. (LOSHCHILOV, 2017) proposed the Adam with decoupled weight regularization, also known as AdamW, in which the weight decay regularization is instead added at equation 3.58. The modified equation with decoupled weight decay becomes

$$\theta_t = \theta_{t-1} - \eta_t (\alpha \cdot m'_t / (\sqrt{v'_t} + \epsilon) + \lambda \theta_{t-1}), \tag{3.59}$$

where λ is the weight decay hyperparameter and η_t is a scheduled multiplier, which can be obtained through a learning rate scheduling function, as defined in Section 3.6.3.

3.6.5 Loss functions

A loss function is used to compare the predicted output of a model with its desired output, or ground truth, measuring the difference or similarity between the predicted and expected outputs (TERVEN *et al.*, 2024; ZHAO *et al.*, 2015; CIAMPICONI *et al.*, 2023). Minimizing the loss value is the goal of the machine learning training procedure. The loss value is what is back-propagated in the back-propagation algorithm. Defining the best loss function for a given task is essential. For image classification, cross entropy loss (KULLBACK; LEIBLER, 1951) is usually used, while for image restoration the Charbonier loss (CHARBONNIER *et al.*, 1994) may be used.

3.6.5.1 Cross Entropy Loss

The cross entropy loss (MAO *et al.*, 2023), also known as Kullback-Leibler divergence (KULLBACK; LEIBLER, 1951; KIM *et al.*, 2021), is usually the function used for image classification optimization, in particular the softmax cross entropy loss (KIM *et al.*, 2021), which is defined as

$$\mathcal{L}(x,y) = -\frac{1}{N}\sum_{n=0}^{N}\sum_{c=0}^{C} w_c log\frac{e^{x_{n,c}}}{\sum_{i=0}^{C} e^{x_{n,i}}} y_{n,c}, \tag{3.60}$$

where N is the number of samples in the iteration, C is the number of classes, w_c is a weight associated with class c, which may be used to deal with class imbalance in the dataset. $x_{n,c}$ and $y_{n,c}$ are, respectively, the predicted and expected output for sample n and class c.

3.6.5.2 Charbonier Loss

The charbonier loss(CHARBONNIER *et al.*, 1994; ZAMIR *et al.*, 2020; WANG *et al.*, 2022), in the context of image restoration is defined as

$$\mathcal{L}(I,I') = \sqrt{\| I - I' \|^2 + \epsilon^2}, \tag{3.61}$$

where I' is the expected output, I is the restored imaged and ϵ is a constant, usually set to 10^{-3}.

3.6.6 Data Augmentation

Data augmentation is a technique to generate new data with various orientations by changing some characteristics of the training data itself (MAHARANA *et al.*, 2022; SHORTEN; KHOSHGOFTAAR, 2019). Data augmentation generates more data from a limited amount of data and reduces overfitting(MAHARANA *et al.*, 2022). These methods can be applied offline, by saving multiple variations of images in an augmented dataset; or online by only modifying images during training at each iteration as part of the pre-processing step.

Common data augmentation methods for computer vision include removing or masking parts of the image (DEVRIES; TAYLOR, 2017; ZHONG *et al.*, 2020b; SINGH *et al.*, 2018; CHEN *et al.*, 2020), combining different images (INOUE, 2018; ZHANG, 2017; VERMA *et al.*,

2019; YUN *et al.*, 2019; SUMMERS; DINNEEN, 2019; TAKAHASHI *et al.*, 2019), and basic image manipulations(SHORTEN; KHOSHGOFTAAR, 2019), including automatic machine learning methods to search for data augmentation policies(CUBUK *et al.*, 2019; LIM *et al.*, 2019; MÜLLER; HUTTER, 2021).

A standard data augmentation strategy for computer vision is the random crop and random horizontal flip (HE *et al.*, 2016; HUANG *et al.*, 2017). Horizontal flip consist of mirroring an image horizontally, given a probability, usually 50%. Random crop can be done in two ways. For smaller images it's usually done by adding empty pixels to each side of the image and then cropping a random region of the original image size. This random crop method has an effect of shifting the image, training the neural network model for translation invariance. The other method is to crop a random region of the image and then resizing it to a target size. This second method has effects of shifting, resizing and changing image width/height ratio, training the model for invariance in those aspects.

The Autoaugment(CUBUK *et al.*, 2019) and TrivialAugment(MÜLLER; HUTTER, 2021) data augmentation methods apply two image manipulation functions in sequence to each image, and each function has a probability and a magnitude parameter. The probability parameter means that specific image manipulation function will only be applied given that probability. As a result, though two image manipulations are provided, there is a chance that only one of the two methods is applied, or none, or both. For autoaugment, the two methods are applied in a fixed order with a fixed magnitude for each function. For TrivialAugment, the order and magnitude are chosen at random from a uniform distribution for each image sample. These methods are usually used together with the preprocessing functions.

3.7 MODELS

For the experiments, two baseline models were defined, an efficientNet-b0 (TAN; LE, 2019) for tumor classification and an Uformer-T(WANG *et al.*, 2022) for MR image restoration. These two models are described in more detail in this section.

3.7.1 EfficientNet

The baseline model used for image classification in this work is EfficientNet (TAN; LE, 2019), particularly the EfficientNet-b0 variant. The EfficientNet architecture was designed through an algorithm search for neural architectures, obtaining an efficient design with state-of-the-art accuracy with significantly fewer parameters and a reduced computational cost compared to other CNNs. Although some transformer-based models have obtained higher accuracy than CNNs in vision tasks with large datasets, the inductive bias of CNNs makes these models more appropriate for classification tasks with more limited data, such as medical image classification which often struggles with data. The EfficientNet also has publicly available pre-trained weights, which provide a good weight initialization. Given all these consideration, the EfficientNet was the best choice for this work, also proven by model comparisons in the experimental results section.

The basic block of the EfficientNet architecture is the mobile inverted bottleneck residual (SANDLER *et al.*, 2018; HOWARD *et al.*, 2019). This residual function consist of a 1x1-convolutional layer that expands the number of features by a factor e, usually 6, followed by a depthwise convolution and a second 1x1-convolutional layer that reduces the number of feature to the target output width. Each of the three convolutional layers are followed by a batch normalization layer (IOFFE; SZEGEDY, 2015), but only the first two normalization layers are followed by an activation layer. For efficientNet, the activation layer is the SiLU/swish activation layer (RAMACHANDRAN *et al.*, 2017)3.5.1.4. A Squeeze-and-excitation function (HU *et al.*, 2018)3.5.5.1 is added before the third convolutional layer of each residual, with a contracting factor of 16x in the feed forward function.

The dimensions of the baseline EfficientNet-b0 architecture was defined by a neural network architecture search algorithm which aims to optimize model accuracy and number of FLOPS (floating point operations). The resulting model of the search is the EfficientNet-b0 model defined in table 3.1, in which MB refers to the mobile inverted bottleneck block, Conv+BN+swish indicates a sequence of convolutional, batch normalization and swish activation function are used, GAP means global average pooling layer and FC is the fully connected classification layer. Each inverted mobile block has a specified expansion factor, e, and a given kernel size. When the residual function has stride 2 or the output width is different from the input width, i.e. the

Block	expansion	kernel size	output width	stride	repeats
Conv+BN+swish	-	3x3	32	2	1
MB	1	3x3	16	1	1
MB	6	3x3	24	2	1
MB	6	3x3	24	1	1
MB	6	5x5	40	2	1
MB	6	5x5	40	1	1
MB	6	3x3	80	2	1
MB	6	3x3	80	1	2
MB	6	5x5	112	1	3
MB	6	5x5	192	2	1
MB	6	5x5	192	1	3
MB	6	3x3	320	1	1
Conv+BN+swish	-	1x1	1280	1	1
GAP+FC	-	-	#classes	1	1

Table 3.1: Layers of the EfficientNet-b0 architecture.

output tensor has a different shape than the input tensor, the architecture simply removes the skip connection. Table 3.1 shows the sequence of blocks used to make up the model, including the number of times that each specific block is repeated.

To define larger versions of the EfficientNet architecture, (TAN; LE, 2019) proposed a compound scaling method in which the depth of the model increases by 1.2, the width of the model increases by 1.1, and the resolution of the image increases by 1.15. Applying this scale factor to EfficientNet-b0 produces EfficientNet-b1, repeating this generates EfficientNet-b2, and so on. More variations of the EfficientNet architecture have been proposed by other works (XIE *et al.*, 2020; TAN; LE, 2021).

The Pytorch framework provides weights for all EfficientNet models trained on the ImageNet dataset (DENG *et al.*, 2009). EfficientNet models with ImageNet transfer learning have been used to classify brain tumors in other works (ISLAM *et al.*, 2024; BAUCHSPIESS; FARIAS, 2024).

3.7.2 Uformer

The baseline model used for image restoration in this work is the Uformer architecture (WANG *et al.*, 2022), in most cases the Uformer-T variant. The Uformer architecture is shown in Fig. 3.26.

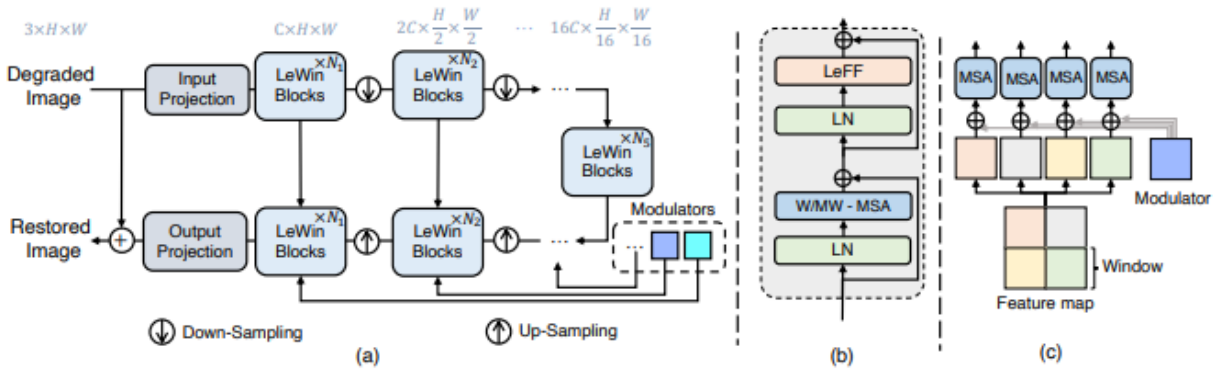


Figure 3.26: Uformer architecture. Image taken from (WANG *et al.*, 2022)

The entire Uformer model may be described as a residual function

$$Uformer(x) = x + U(x), \tag{3.62}$$

where the input of the model, x, is added to its output $U(x)$ to generate the final output, in which $U(x)$ is a U-shaped neural network, which combines the transformer architecture with convolutional layers. In this way, the parameterized part of the model, U(x), is used to estimate the change required by the input, rather than the final output itself. Since the model aims to reconstruct the input while removing image distortions, it can be considered a type of autoencoder.

The first and last layers of the neural network are the input and output projection layers, which are 3x3 convolutional layers that increase the number of features of the input from 3 features (RGB image) to an arbitrary number of features, C, and vice versa for the output projection. For the input projection, the convolutional layer is also followed by a leakyReLU activation function (MAAS *et al.*, 2013).

The U-shaped model has four encoding stages in the contracting path and four decoding stages in the expanding path, with a bottleneck stage in between. At the end of each contracting stage, there is a downsampling layer, which is a 4x4 convolutional layer with stride 2 that doubles the number of features and half the resolution at each spatial dimension. At the start of each expanding stage there is an upsampling layer, which is a 2x2 transposed convolutional layer with step 2 that halves the number of features and doubles the resolution at each spatial dimension. The output of each upsampling layer is concatenated with the input of its corresponding downsampling layer to make the input of each expanding stage. Due to this concatenation of features, the decoder or expanding path has twice the width of the corresponding

encoder path.

The basic block used in each stage of the architecture is the LeWin transformer block defined for this architecture. The LeWin block consists of two residual functions, named window Multi-head Self Attention (WMSA) defined in Section 3.5.5.3 and locally-enhanced feed forward (LeFF), which are defined as

$$\begin{aligned} X'_t &= X_{t-1} + WMSA(LN(X_{t-1})) \\ X_t &= X'_t + LeFF(LN(X'_t)), \end{aligned} \tag{3.63}$$

where LN is the layer normalization function (BA *et al.*, 2016). The WMSA uses 8x8 windows in the window attention function.

Like the common feed-forward layer from the transformer architecture, the LeFF has two linear layers separated by an activation function, the first linear layer increases the number of features by a factor of 4 and the second reduces the number of features to the original amount. The difference of the LeFF is that a depth-wise convolution with kernel 3x3 is added between the two linear layers, along with an additional activation layer. This configuration makes the LeFF modeule very similar to the inverted mobile bottleneck layer used by efficientNet. Since the WMSA uses simple windows without shifting, information between different windows would only flow between different windows after downsampling, but the convolutional kernel of the LeFF module enables information flow between neighboring windows. The activation function used in the LeFF module is the GELU (HENDRYCKS; GIMPEL, 2016).

As an additional modification, the Uformer architecture adds modulators parameters to the WMSA modules of the expansion side of the Uformer. The modulators are learnable tensors with MxMxC shape that are added to each window, where M is the window size and C is the number of feature maps and work as a shared bias for each window.

Since the window size is defined as 8x8, the resolution at the bottleneck has to be an integer multiple of 8x8. The bottleneck layer has a resolution 16x lower than the input image, and as a result the input image needs a resolution of an integer multiple of 128x128 pixels.

Each Uformer variant is defined by the number of LeWin blocks in each stage of the network and by the number of channels, C, of the input projection. The smallest variant in the original paper is the Uformer-T, which has 2 LeWin blocks in each stage (18 blocks in total) and has an initial width, C, of 16 features. For this model, weights pre-trained in the Smartphone Image

Denoising Dataset (SIDD) (ABDELHAMED *et al.*, 2018) were available.

3.8 DATASETS

We provide here an overview of the MRI brain tumor datasets that were used for experimentation. We provide here the details of each dataset as well as a qualitative analysis of the images contained in each dataset and the dataset preparation procedures. Since all datasets are brain tumor datasets, we give distinguishing names to each dataset.

3.8.1 Dataset 1 - Figshare

This dataset, provided by (CHENG *et al.*, 2016), consists of a set of brain T1-weighted CE-MRI images, composed of 3,064 slices of 233 patients, including 708 meningiomas, 1426 gliomas, and 930 pituitary tumors. The images were separated into a train, a validation and a test set (FARIAS *et al.*, 2022). The images belonging to the same patient were grouped and moved as a group to one of the three sets. By moving the images as a group, the model cannot recognize a patient from the training set in the test set, leading to a misleading higher precision (BADŽA; BARJAKTAROVIĆ, 2020). 2096 images were moved to the training set, 515 to the test set, and the remaining 413 to a validation set (FARIAS *et al.*, 2022). This dataset is available for download at <dx.doi.org/10.6084/m9.figshare.1512427>, from which we take the name of the dataset. Fig. 3.27 exemplifies the three classes in the dataset. Slices in this dataset are from all 3 views: sagtital, coronal, and axial. In a qualitative analysis, images from this dataset are of good quality, most images are undegraded, or with low-magnitude artifacts.

3.8.2 Dataset 2 - Siar

This dataset is provided by (SIAR; TESHNEHLAB, 2022; SIAR; TESHNEHLAB, 2019), and contains MRI scans of 136 people without tumors and 138 patients with brain tumors. The version available for download at <https://www.kaggle.com/datasets/masoumehsiar/siardataset> contains 3800 images of Tumors and 3200 images of NoTumor. We applied a train-test split of 80%-20%, resulting in a test set with 760 tumor scans and 640 No Tumor scans. Fig. 3.28

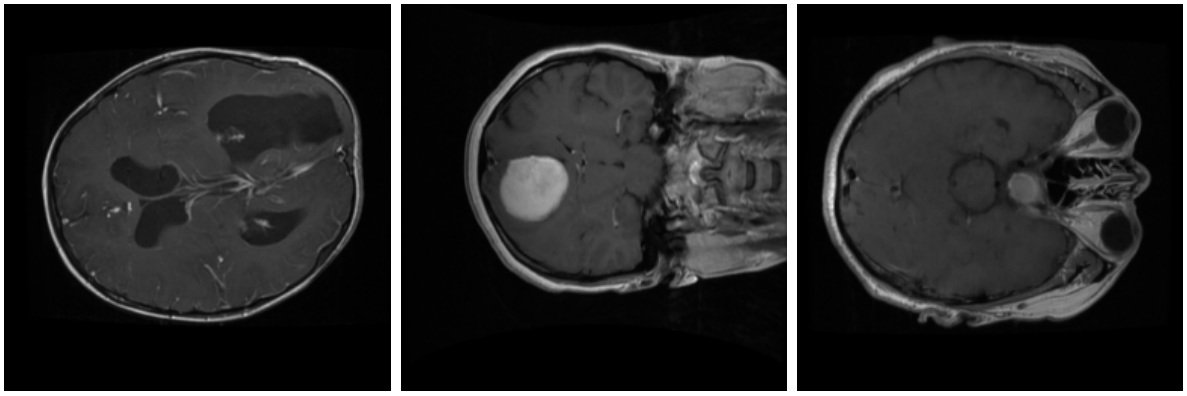


Figure 3.27: Image samples taken from (CHENG *et al.*, 2016). From left to right: glioma, meningioma and pituitary tumors.

exemplifies the two classes in the dataset.

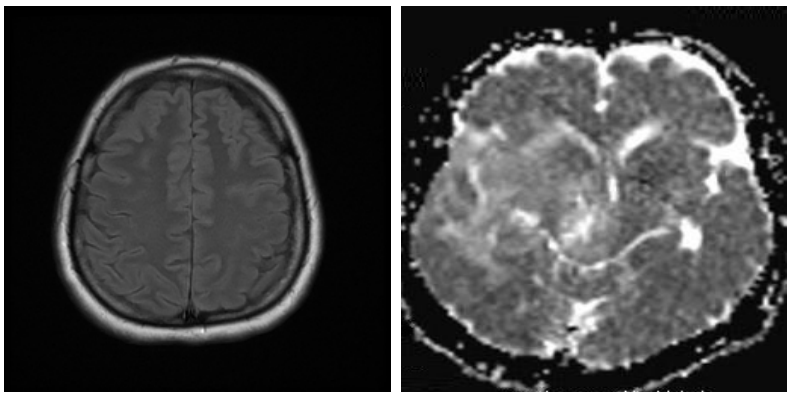


Figure 3.28: Image samples taken from (SIAR; TESHNEHLAB, 2022). From left to right: NoTumor and tumor classes.

A visual inspection of this dataset shows that all 'Normal' scans are from the axial view and mostly undegraded, while the 'Tumor' scans are a mix of axial, sagittal, and coronal views and include several degraded images. We also observe that the dataset characteristics create a bias toward classifying sagittal and coronal view scans as 'Tumor' scans, since these views almost only appear in the 'Tumor' class in this dataset.

3.8.3 Dataset 3 - Br35H 2020

This dataset (HAMADA, 2020) consists of 1500 'Tumor' scans and 1500 'NoTumor' scans and is available at <kaggle.com/datasets/edhamada0/brain-tumor-detection>. We split the dataset into a train and test set, so that the train set has 1200 'Tumor' and 1200 'NoTumor' scans while the test set has 300 'Tumor' and 300 'NoTumor' scans. Fig. 3.29 exemplifies the

two classes in the dataset.

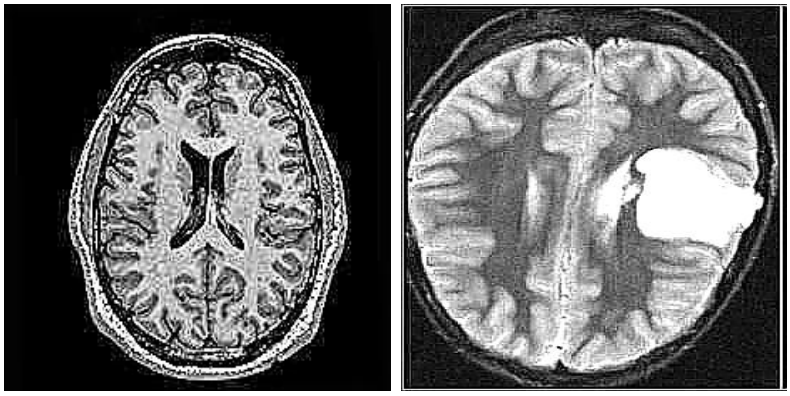


Figure 3.29: Image samples taken from (HAMADA, 2020). From left to right: NoTumor and tumor classes.

Visual inspection of this dataset shows that all images are from the axial plane. In qualitative analysis, most images would be classified as poor quality or degraded images, on both image classes.

3.8.4 Dataset 4 - Kaggle

This dataset provided by (NICKPARVAR, 2021) is a combination of three datasets: the Figshare (CHENG *et al.*, 2016) Br35H 2020 (HAMADA, 2020) and Sartaj (BHUVAJI *et al.*, 2020) datasets. It contains 7023 brain MRI scans are split into 4 categories: glioma, meningioma, pituitary, and NoTumor. Images for the NoTumor class come from both the Sartaj and Br35H datasets, images from the meningioma and pituitary class come from both the Figshare and sartaj datasets, while Glioma images come from the Figshare dataset, due to some observations that glioma images in the sartaj dataset may be wrongly labeled. The dataset provides its own train and test set division. The training set has 13,21 glioma scans, 13,39 meningioma scans, 14,57 pituitary scans, and 15,95 scans without tumor. The test set has 300 glioma scans, 306 meningioma scans, 300 pituitary scans, and 405 scans without tumor. Fig. 3.30 exemplifies the four classes in the dataset.

The images taken from Figshare are mostly undegraded, while the images taken from the other two datasets include several degraded images. As a result, there is a higher proportion of degraded images in the NoTumor class, and most images in this category are from the axial plane. The Glioma class is largely undegraded. The meningioma and pituitary classes show

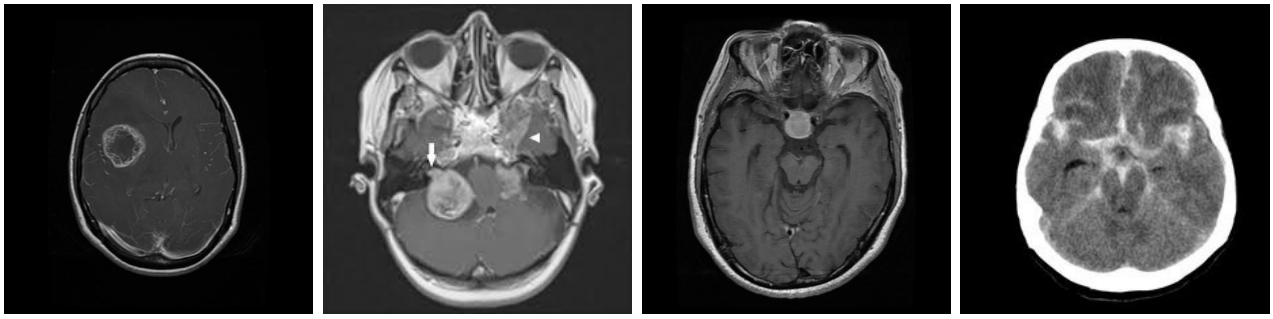


Figure 3.30: Image samples taken from (NICKPARVAR, 2021). From left to right: glioma, meningioma, pituitary tumors and NoTumor classes.

a mixture of degraded and undegraded images. In the meningioma sample in Fig. 3.30 we also see some markings, including an arrow that points to the tumor. From this we point out that, although we treat several types of artifacts in our work, there is still some image degradation that will not be modeled, such as those arrows, but might still have some impact on performance.

In this dataset sets of slices from the same subject are separated, with some slices in the train set and others in the test set. Fig. 3.31 shows three examples of successive slices in which one slice is in the training set and the other in the test set. The similarity of the successive slices can be seen even in the angle of the head in the scan. A consequence of this is that during testing, the type of tumor can be identified based on the subject, rather than the tumor itself, leading to higher test accuracy.

3.8.5 Dataset 5 - Zenodo

The Zenodo brain magnetic resonance image dataset (QADRI *et al.*, 2022) consists of 1,287 MRI scans divided into three categories of brain tumors: adenoma, glioma, and meningioma. There are 414 adenoma scans, 414 meningioma scans, and 459 glioma scans. We generate a random train-test split, with 80% of the images for training and 20% for testing. Fig.3.32 exemplifies the three classes of this dataset.

The images in this dataset are almost exclusively from the axial plane, with few exceptions in the adenoma class. Several images show low-intensity image degradations, such as ghosting, noise, and blurring. In most images, characteristics of T2 weighted MRI can be identified, with brighter gray matter and darker white matter.

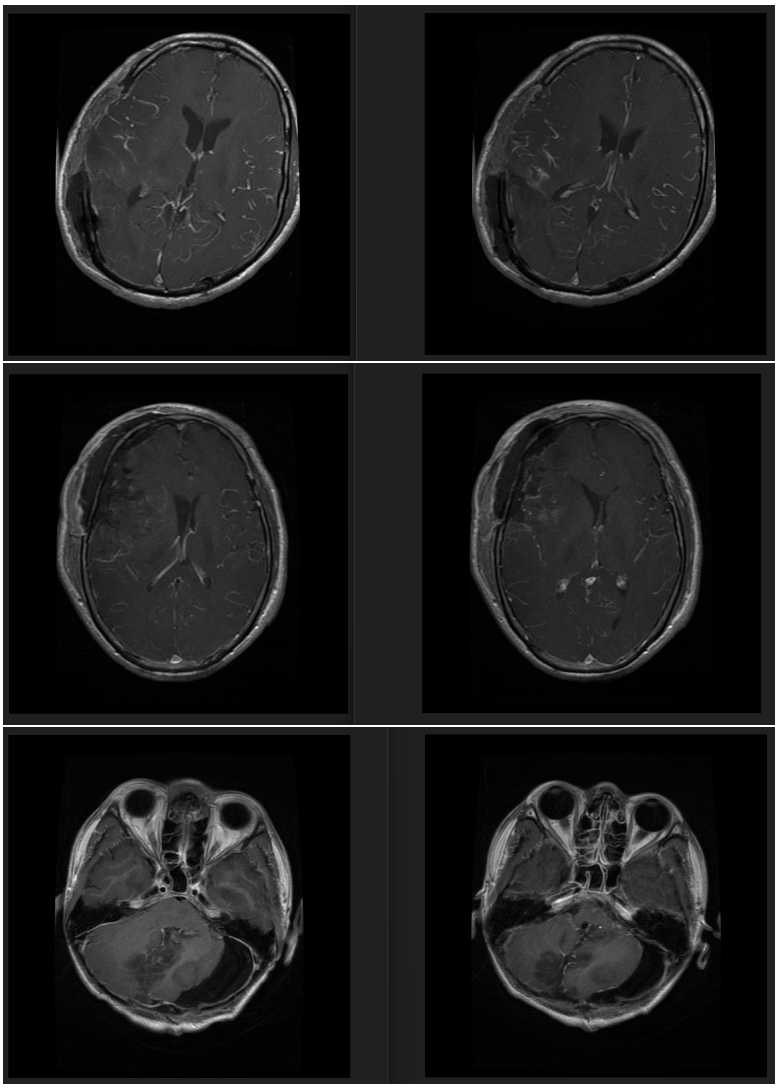


Figure 3.31: Pairs of successive slices in which one image is in the training set and one is in the test set from the Kaggle dataset (NICKPARVAR, 2021). Left: image from the training set. Right: image from the test set.

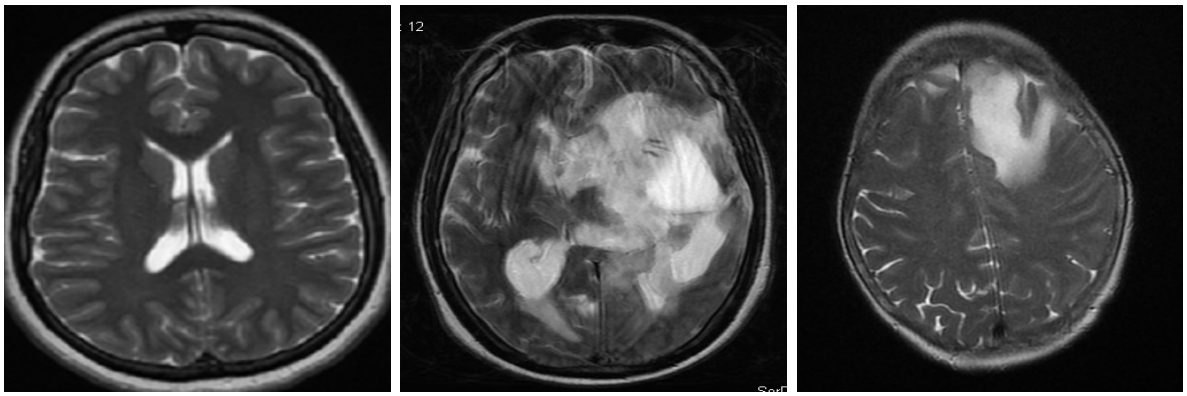


Figure 3.32: Image samples taken from (QADRI *et al.*, 2022). From left to right: adnoma, glioma and meningioma tumors.

CHAPTER 4

ARTIFACT GENERATION AND MODEL TRAINING

Our proposed solution to perform tumor diagnosis from magnetic resonance images consists of two neural network models. The initial model in our framework is an image classification model, which categorizes magnetic resonance images into classifications of non-tumorous or tumorous, further subdividing into specific tumor types based on the parameters of the designated test set. The second model is an image restoration model that ensures the image quality of the MRI to be analyzed by the classification model. Each model is trained independently. However, to train the image restoration model, image pairs are required, consisting of a good quality target image and a degraded input image. To generate these requisite image pairs, high-quality image datasets were employed alongside an artifact generation function to produce their degraded counterparts.

Fig. 4.1 shows the proposed framework, which, as illustrated, corresponds to a modular solution consisting of training dataset, pre-processing, artifact generator, restoration model, and classification model. In our experiments, we evaluated the impact of the dataset, the artifact generator, image restoration, and image classification by testing various training dataset, neural network models, and artifact generation methods. In this chapter, we describe the formulation of the artifact generator function, the methodologies used in training the models, and the procedures used to evaluate the solution.

4.1 ARTIFACT GENERATOR

Our proposed artifact generator function consists of a sequence of modules in which each module adds a specific artifact type to an image. As such, the artifacts are added sequentially to an image. The exact order in which the modules are placed may change the aspect of the final degraded image output. Fig. 4.2 shows a flowchart of the artifact generator, while Algorithm 1

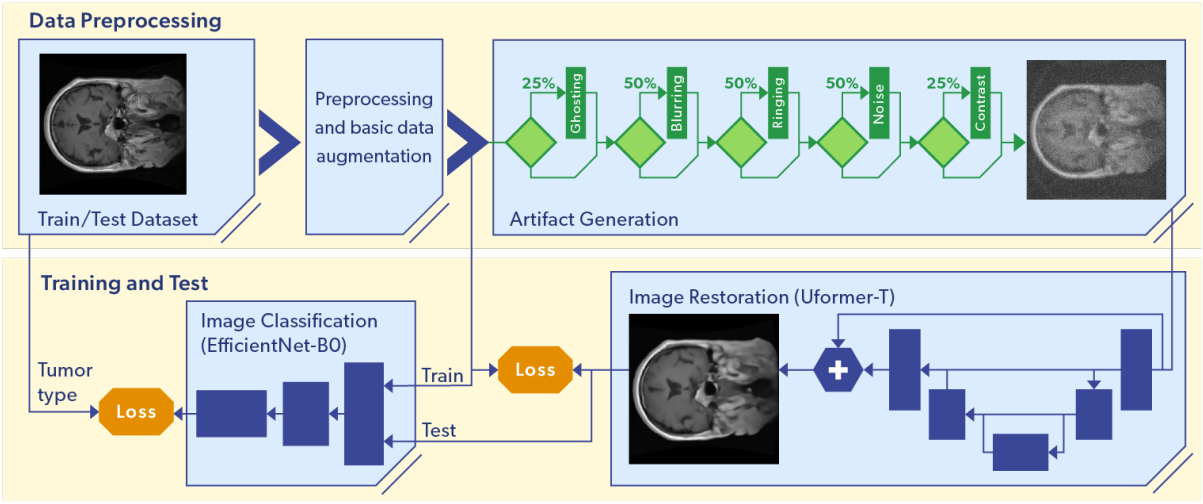


Figure 4.1: Proposed framework split into five modules: Dataset, pre-processing, artifact generator, restoration model and classification model. Modified from (BAUCHSPIESS; FARIAS, 2024).

shows the pseudocode. Taking some inspiration from image augmentation methods (CUBUK *et al.*, 2019; LIM *et al.*, 2019; MÜLLER; HUTTER, 2021), each module requires a specific artifact type, a probability value, and magnitude information, all of which we defined with six control parameters, named artifacts *prob*, μ, σ, max and min. The last four parameters refer to statistics of the pixel intensities.

Algorithm 1 Artifact module

```
procedure AddArtifact(image, prob, artifact, μ, σ, min, max)
    u ← random.uniform(0, 1)
    if u < prob then
        n ← random.normal(mean = μ, std = σ)
        magnitude ← clip(n, min, max)
        image ← apply(image, magnitude, artifactType = artifact)
    end if
    return image
end procedure
```

The artifact generator starts by drawing a random value u from a uniform distribution $\mathcal{U}(0,1)$. If this value is lower than the probability parameter *prob*, the module will add an artifact to the image; otherwise, the process is finished without modifying the image. This first step ensures that the artifact will not be applied to all images and, as a consequence, each artificially degraded image will have an arbitrary number of artifact types. In this way, we can simulate the datasets in which each image has an arbitrary number of artifact types. Naturally, for higher *prob* values, the specified artifact type will be applied more frequently.

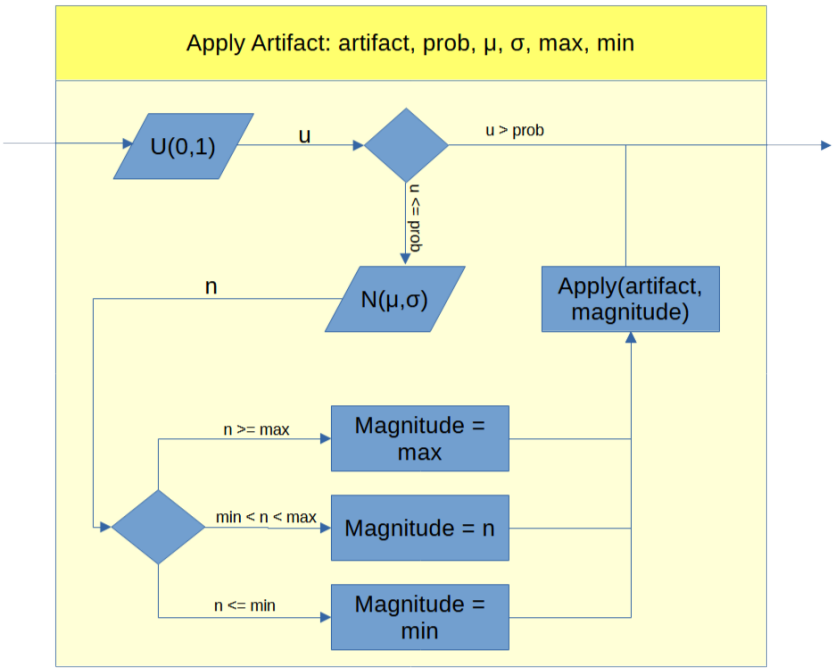


Figure 4.2: Module to degrade an image with a given artifact type. The image is only degraded given a probability *prob* and the magnitude of the degradations is drawn from a normal distribution and limited to the [min,max] range.

If a decision is made to apply the artifact, the next step is to define the magnitude of the image distortion, which is done with the parameters μ, σ, max and min. A random magnitude n is initially selected from a normal distribution $\mathcal{N}(\mu, \sigma)$, where μ represents the average value and σ denotes the standard deviation. The final magnitude value is obtained by limiting the value of n to the range $[\min, \max]$. If $n >$ max, the magnitude value will be set at max, if $n <$ min the magnitude value will be set at min, and in any other case the magnitude value will be set to n. The distorted MRI scans are not all distorted to the same degree, and the magnitude value is used to control this variation. These four control parameters are used to restrict the magnitude value to relevant values.

The final step consists in adding the synthetic artifacts to the image. This is done by applying one of the six image processing operations defined in Section 3.3.The artifact parameter determines the operation to apply to the image, to simulate Rician noise, Gibbs ringing, Gaussian blur, poor contrast, ghosting, or JPEG compression. In Section 3.3 each operation has a parameter indirectly defined by the variable *intensity*, but in the module proposed in this section these parameters are directly defined by the previously defined magnitude value. The "Apply" block in the flowchart is then responsible for applying the specified artifact type with

the specified magnitude to the image. Once the module's procedure is completed, the image, which may have been altered, is forwarded to the subsequent module. This module introduces a distinct type of artifact to the image through a similar process. If no additional modules remain in the sequence, the image produced is the ultimate degraded version.

To define which artifact modules should be included in the artifact generator, as well as the parameters of these modules and the order in which the modules will be placed, we evaluated the degraded images in the Kaggle dataset (NICKPARVAR, 2021). We identified six main types of MRI artifacts: Rician noise, Gibbs ringing, Gaussian blur, poor contrast, ghosting, and JPEG compression artifact. For every identified kind of artifact, we establish a module corresponding to that type and calculate the mean, standard deviation, minimum, and maximum parameters to ensure a realistic distribution of image degradations. To determine the parameters for each artifact, we created artificially degraded images and assessed them against reference degraded images (from the Kaggle dataset) through histogram comparison and image qualitative analysis. In this manner we define the magnitude distribution of each artifact type. After defining the artifact types and magnitudes, we define the order in which the artifacts have to be applied, based on how the artifacts affect each other and logical understanding of the moment the artifact is introduced in images. Next, we describe the artifact generation process for each of the identified artifacts.

4.1.1 Rician Noise

To evaluate the noise present in noisy MRI scans, we compute the histogram of pixels in the background regions of these images, as exemplified in Fig. 4.3. For noise-free images, the background histogram should be composed almost exclusively of '0' values, as exemplified in Fig. 4.4. Different background pixel intensity distributions may be attributed to the presence of additive noise in the image. Fig. 4.5 shows the histogram of the pixel intensities of four noisy images from the Kaggle dataset.

In order to produce noisy images, we attempted to craft images whose histograms closely resemble those of noisy images from the Kaggle dataset, which served as reference noisy images. We noticed that the histograms of these noisy images typically have a mean ranging from 10

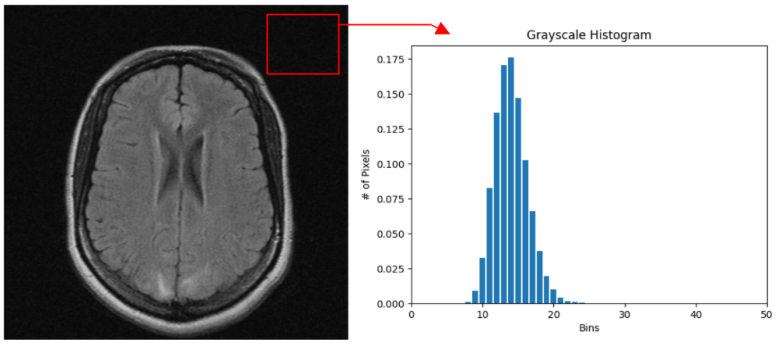


Figure 4.3: Noisy image and histogram of a background region of that image.

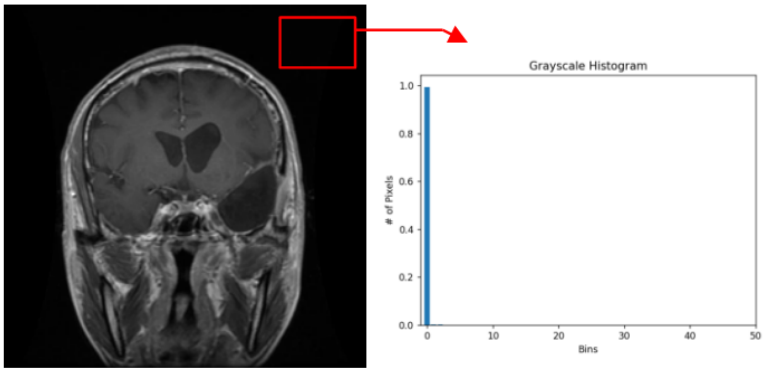


Figure 4.4: Undegraded image and histogram of a background region of that image.

to 20. Furthermore, we observed that the top two histograms in Fig. 4.3 appear not to align with the Rician distribution depicted in Fig. 3.7, in contrast to the other three. We suspect that this discrepancy arises from extra image degradations, such as those introduced by JPEG compression artifacts.

To reproduce the noise found in these MRI scans, we used the approach detailed in Section 3.3.1. The skimage library was employed to simulate Rician noise in images, after which we derived the pixel intensity histogram for the resulting noisy images. We created the histograms depicted in Fig. 4.6 using standard deviation parameters of 0.025, 0.05, 0.075, and 0.1. It is evident that the histogram distribution of these images resembles the Rician distribution illustrated in Fig. 3.7. The standard deviation ranges from 0.05 to 0.1, and the mean falls within the interval [10,20], similar to what is shown in Fig. 4.5.

From these findings, we specify the noise module parameters within our artifact generation function. The normal distribution is characterized by a mean of 0.075 and a standard deviation of 0.02. Consequently, this produces noise with mean pixel values ranging between [10,20] in background areas, consistent with the noisy images found in the Kaggle dataset. The standard

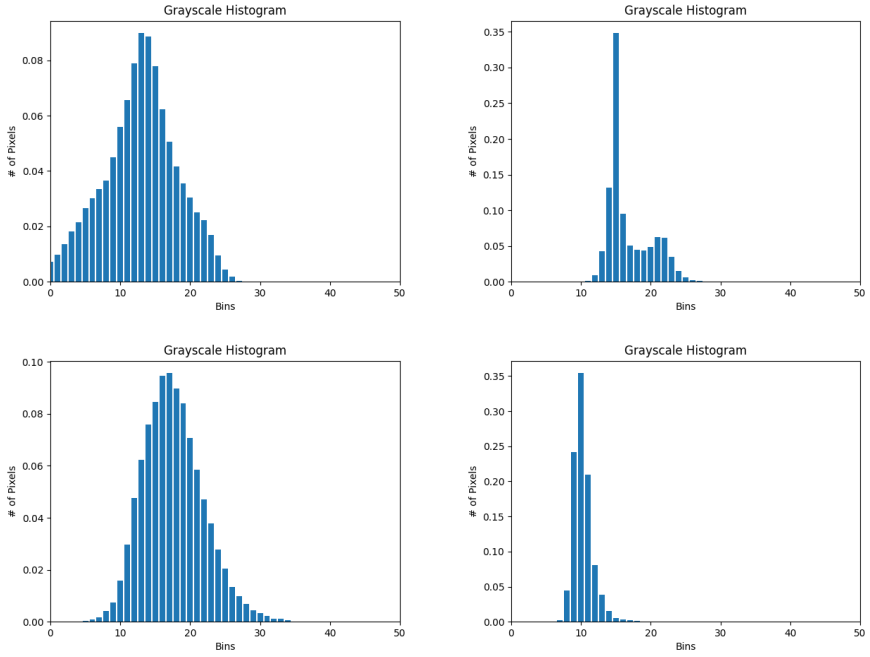


Figure 4.5: Histograms of the background of scans taken from the Kaggle dataset, in which noise was identified.

deviation's limits are set at 0.005 and 0.125, respectively.

4.1.2 Gibbs Ringing

Upon examining the histograms in Fig. 4.6 and 4.5, derived from the Kaggle dataset images, we observed they exhibit a narrower range of pixel values. This variation can likely be ascribed to the influence of low-pass filters, specifically the ideal and Gaussian low-pass filters. In Fig. 4.7, two histograms of Rician noise with a standard deviation of 0.1 are presented. An ideal low-pass filter with a radius of 66 was applied to the histogram on the right. Evidently, the ideal low-pass filter constricted the range of pixel values, making the resulting histogram resemble those in Fig. 4.5 more closely.

As discussed in Section 3.3.2, the application of an ideal low-pass filter can lead to the occurrence of the Gibbs ringing effect. This ringing artifact can also be detected in images from the Kaggle dataset. Figure 4.8 displays two instances in which ripples align parallel to the sharp edges of the cranium. These scans exhibit both Gibbs ringing and Rician noise.

To generate images with both noise and ringing, we add Rician noise and afterwards apply an ideal low-pass filter. Fig. 4.9 shows two images subjected to Rician noise and Gibbs ringing.

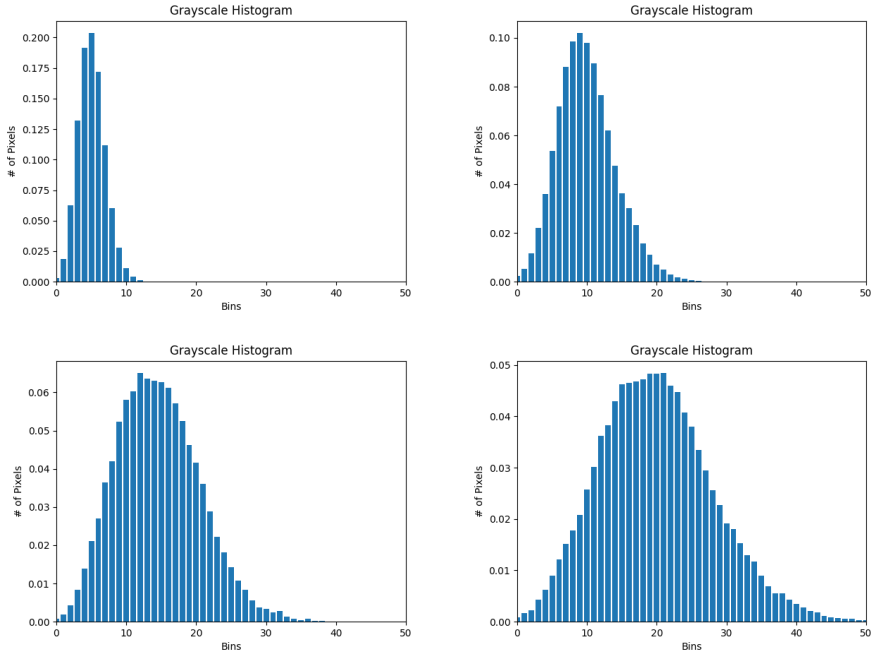


Figure 4.6: Histograms of Rician noise with standard deviation 0.025, 0.05, 0.075 and 0.1.

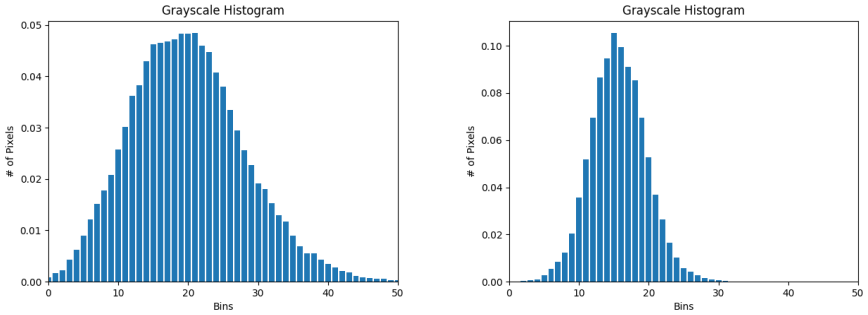


Figure 4.7: Histograms of the background of scan with added noise at standard deviation 0.1 without low pass filtering on the left and with a circular ideal low pass filter with radius 66 on the right.

In the first image, Rician noise with a standard deviation of 0.25 was added, while the second image featured noise with a standard deviation of 1.25. We employed an ideal low-pass filter with a radius of 66 on both these noisy images to produce the final results. It is evident from Fig. 4.8 that the ringing effect appears less pronounced compared to Fig. 3.9, where the ringing is isolated. Furthermore, Fig. 4.9 indicates that combining noise with a low-pass filter can mitigate the ringing effect. Additionally, as previously discussed in Section 3.3.2, blurring can also reduce the Gibbs ringing effect. Moreover, Fig. 4.9 reveals that the ideal low-pass filter gives the noise a visually "thicker" appearance compared to the pure noise shown in Fig. 3.8. Qualitatively, this "thicker" noise resembles the noise found in most images from the Kaggle

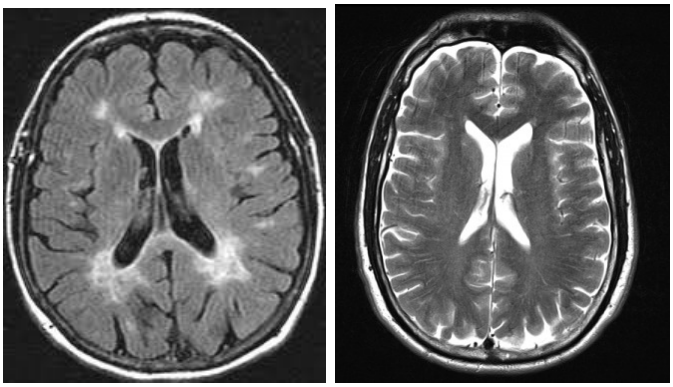


Figure 4.8: MRI scans with ringing artifacts taken from the Kaggle dataset.

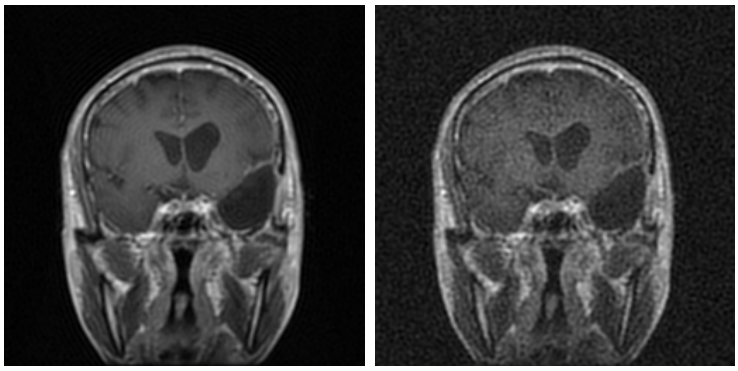


Figure 4.9: MRI images with Rician noise added with standard deviation 0.25 on the left and 1.25 on the right and ideal low pass filter with radius 66 applied to both images.

dataset.

The magnitude of the artificial Gibbs-ringing artifact is determined by the radius of the optimal low-pass filter; in which a smaller radius results in more intense ringing. Based on the histogram shown in Fig. 4.7 and the qualitative evaluations in Figs. 3.9, 4.8, and 4.9, we have chosen a mean radius of 80, with a standard deviation of 10, a minimum of 40, and a maximum of 150. These parameters cause the ideal low-pass filter to produce the ringing effect, but primarily at the lower intensity levels observed in the Kaggle dataset.

4.1.3 Gaussian Blur

In addition to employing the ideal low-pass filter, we applied a Gaussian blur filter to further align the noise histograms with those found in Fig. 4.5. Using the OpenCV Gaussian blur function, the kernel size can be determined by the standard deviation of the Gaussian blur. Fig. 4.10 presents histograms of noise processed with both the ideal low-pass filter and

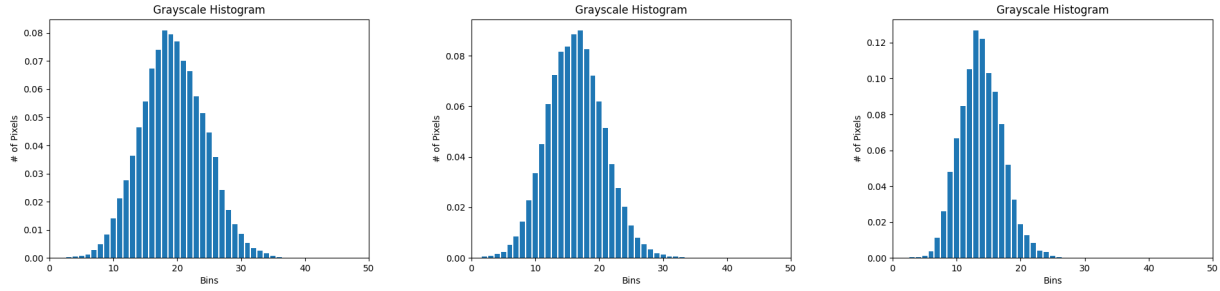


Figure 4.10: Histograms of noise generated at standard deviation 0.1, followed by ideal low pass filter with radius 66 and Gaussian blur with standard deviations of 0.015, 0.568 and 1.122.

Gaussian blur at standard deviations of 0.015, 0.568, and 1.122. As anticipated, enhancing the Gaussian blur filter's intensity narrows the pixel value range, thus rendering this synthetic noise more comparable to that in Fig. 4.5.

In order to further assess the blurring artifact, we conducted a qualitative analysis contrasting blurry images from the Kaggle dataset with those that were artificially blurred. Fig. 4.11 depicts two blurred images from the Kaggle dataset at the top and two artificially blurred images below, which were applied with Gaussian blur at standard deviations of 1.25 and 2.5, respectively. The blurring effect can be identified as a reduction in edge sharpness. From our observations, the blur applied with a standard deviation of 1.25 closely resembles the reference images, while the more pronounced blur appears stronger than any images present in the reference datasets. As a result of these evaluations, for the Gaussian blur module, we established the mean and standard deviation of the blur magnitude to be 1.25 and 0.75, respectively. The range for blur intensity was set from a minimum of 0.015 to a maximum of 2.

4.1.4 JPEG Compression

JPEG compression artifacts were identified by the "blockiness" aspect of some images in the reference dataset. These artifacts are not produced by the image acquisition process, but by the data reduction process obtained with the JPEG compression algorithm. Figure 4.12 displays two images highlighting the artifacts introduced by JPEG compression. For a qualitative analysis, we employed the OpenCV library's compression function to compress an original, undegraded image at quality levels of 55% and 10%. The outcomes are presented in Fig. 4.13. At reduced quality settings, the square artifacts are predominantly represented by a single

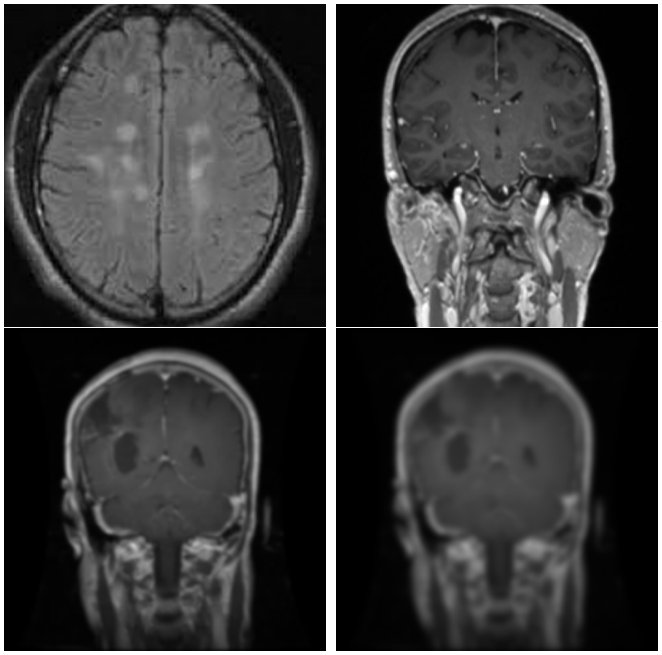


Figure 4.11: Top: real MRI scans with with blurring artifact. Bottom: MRI scans with artificially added blurring artifact, generated with standard deviation values: 1.25 (left) and 2.5 (right).

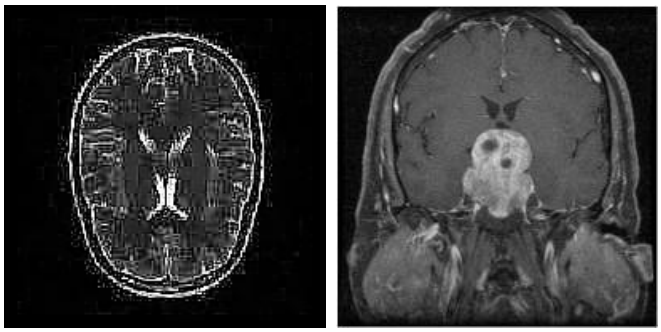


Figure 4.12: MRI with JPEG compression artifact taken from the Kaggle dataset.

value (the DC component), while at enhanced quality, these artifacts incorporate frequency components.

In addition, we assessed how JPEG compression affects noise histograms. We created a Rician noise image with a standard deviation of 0.1, then applied an ideal low-pass filter with a radius of 66, followed by a Gaussian blur filter with a standard deviation of 1.122. We produced three JPEG compressed versions of this image at quality levels of 80%, 50%, and 30%. The histograms for these three images are depicted in Fig. 4.14. These histograms show that, as compression quality decreases, the noise histograms diverge from the Rayleigh distribution, which was also seen in some noise histograms from real cases shown in Fig. 4.5.

Utilizing noise histograms and qualitative assessments of JPEG compression shown in

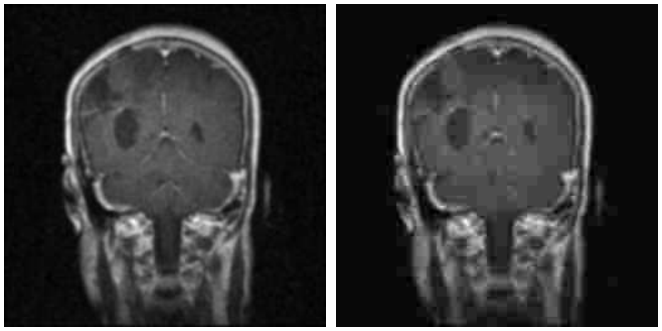


Figure 4.13: MRI scan compressed with JPEG at 55% (left) and 10% (right) quality.

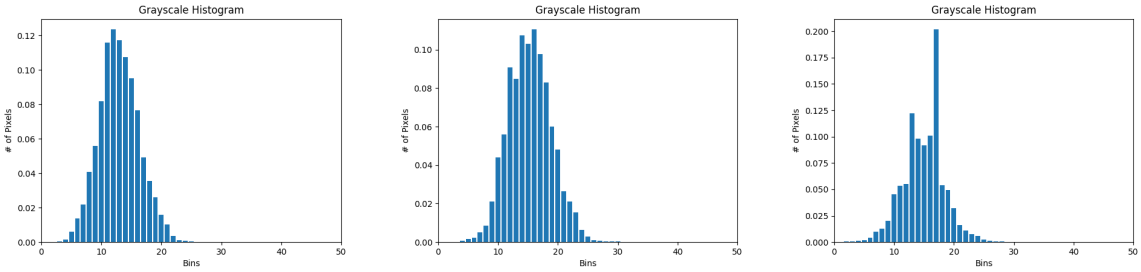


Figure 4.14: Histograms of noise images. For all images, the noise was generated with standard deviation 0.1, followed by ideal low pass filter with radius 66 and Gaussian blur at standard deviation 1.12. Each image was compressed with JPEG at 80%, 50% and 30% quality, respectively.

Figs. 4.12 and 4.13, we establish the magnitude parameters for the JPEG compression segment within the artifact generator. The magnitude in this module specifies the compression quality employed by the OpenCV JPEG compression function, where a lower magnitude results in a more degraded image. We determine the mean magnitude as 70%, with a standard deviation of 40%, and set the parameters to range from 20% to 100%.

4.1.5 Poor Contrast

One form of degradation detected in MR images is low contrast. To assess this artifact, the histograms of the images were obtained from the Kaggle reference data set(NICKPARVAR, 2021). Fig. 4.15 scans and their corresponding histograms are shown, with Fig. 4.15(a) exemplifying a good contrast image and Fig. 4.15(b - c) exemplifying poor contrast images.

In the good quality scan, Fig. 4.15(a), the histograms reveal a dense cluster of pixels at or near the zero value, representing the background, accompanied by a spread of gray-value pixels. In contrast, the histogram from Fig. 4.15(b) has a larger number of pixels concentrated at the

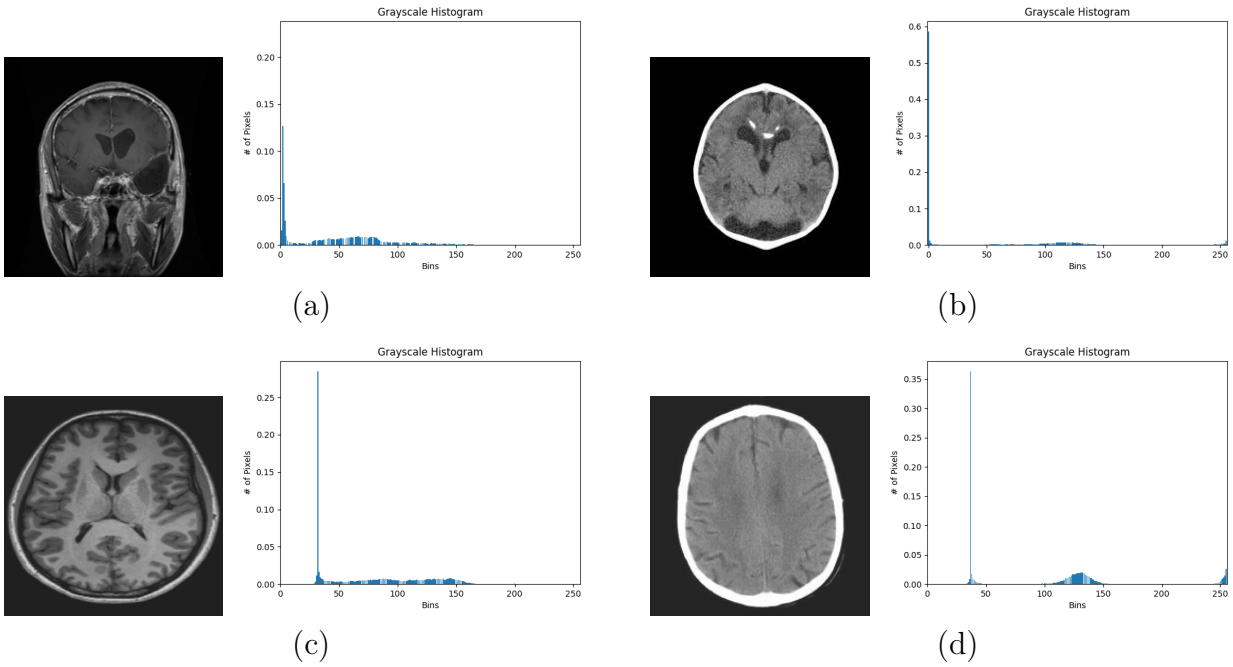


Figure 4.15: Scans and their corresponging histograms. (a) good contrast image. (b-d) poor contrast images.

zero value, as well as a significant presence of pixels at the maximum value, 256, which may be due to saturation following contrast enhancement. The cluster of pixels at the 256 value pertains to the cranium region, where pixel saturation probably resulted in a loss of detail. Another effect observed in this image is the presence of noise in the brain region, or gray areas, and lack of noise in the black and white saturated regions. In Fig. 4.15(c), it is evident that the background has shifted from black to light gray. Consequently, the histogram now shows that the minimum pixel value has increased from 0. This alteration might be linked to a contrast reduction function or could be attributed to a low-contrast artifact.

Fig. 4.15(d) exhibits a mixture of features of high- and low-contrast artifacts: a gray background, a saturated maximum brightness in the cranium region, and noise present in the brain area but absent in the background. Based on these features, it is possible that the third poor-contrast image resulted from a sequence of high-contrast followed by low-contrast artifacts within the same image. Considering the high contrast artifact appears to eliminate background noise, it suggests this might have been an intentional post-processing technique responsible for creating those artifacts.

Based on the insights from Fig. 4.15, we establish the poor contrast function as a series of two artifact modules: one targeting low contrast and the other high contrast. This combination

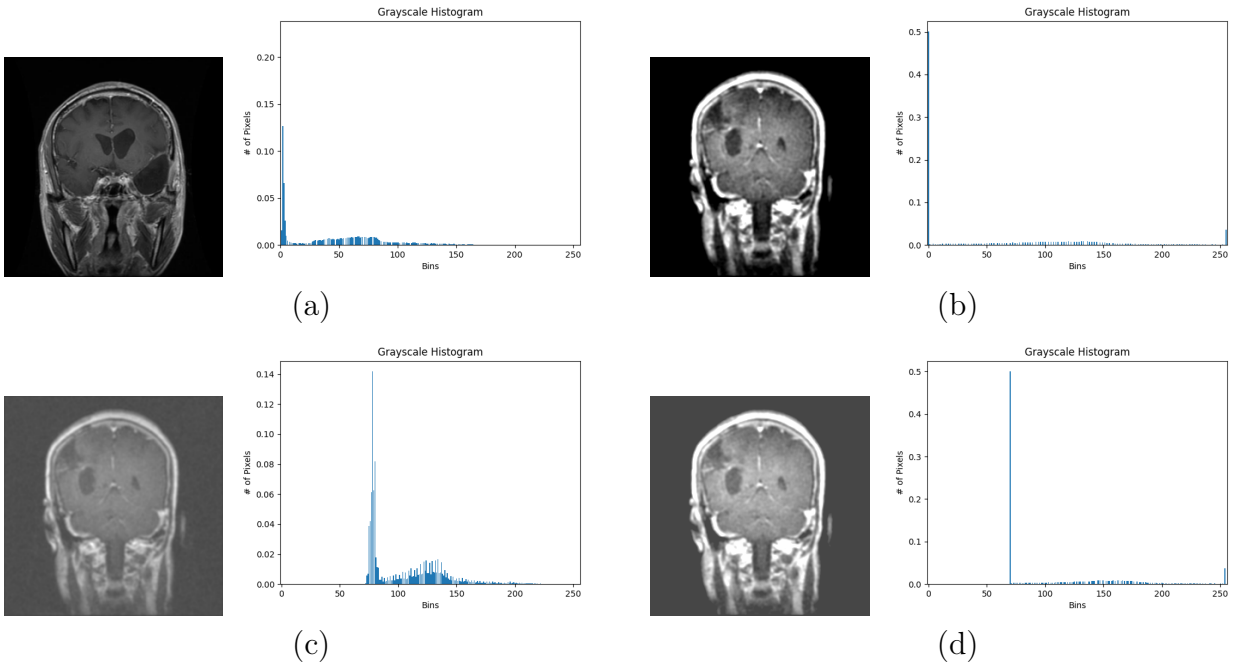


Figure 4.16: Scans with good quality and for possible outputs for the poor contrast module pair.

of artifact modules enables the creation of all four contrast scenarios depicted in Fig. 4.15. Additionally, noise was introduced to the three poor contrast images, and the characteristics of these three types of poor contrast can be observed both in the images and their corresponding histograms.

Taking into account the existence of two distinct contrast artifacts, equation 3.11 was adjusted to accommodate this scenario. Both algorithms implement a linear transformation bounded by the following equation:

$$\text{Image}' = \min\left(\max\left(\alpha \cdot \text{Image} + \beta, 0\right), 255\right), \tag{4.1}$$

where saturation levels of 0 and 255 are applied to prevent values from exceeding the permissible range, particularly for high contrast contexts when the data type permits values outside the conventional range, such as when pixel values are expressed as floating points. Two different sets of equations were formulated to establish the parameters α and β for the contrast function, both of which are influenced by a single parameter m that controls magnitude. For scenarios with high contrast artifacts, the specifications are given by

$$\alpha = 1 + 0.4 \cdot m$$

$$\beta = -30 - 6 \cdot m,$$

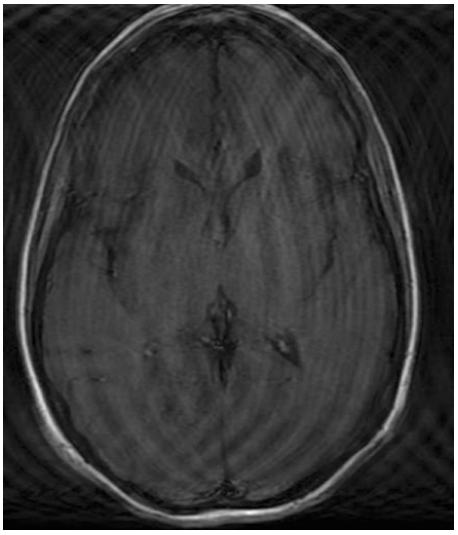


Figure 4.17: MRI with ghosting artifact, taken from the Kaggle dataset.

while for low contrast artifacts, we have

$$\alpha = (11 - m) \cdot 0.09$$
$$\beta = 255 \cdot (1 - \alpha).$$

These equations were empirically adjusted to ensure that the histograms depicted in Fig. 4.16 resemble more closely the ones shown in Fig. 4.15.

In the high contrast module, the magnitude parameter m is derived from a normal distribution characterized by a mean of 3, standard deviation of 2, with a range confined between 0 and 8. Conversely, in the low contrast module, the magnitude parameter m is taken from a normal distribution having a mean of 2, standard deviation of 1, and is similarly limited to values between 0 and 4.

4.1.6 Ghosting

Among the various artifacts present in the Kaggle dataset, ghosting was relatively less frequent. The reference dataset's most prominent example of a ghosting artifact, depicted in Fig.4.17, exhibits a significant occurrence of ghost images.

To generate ghosting, we used a function provided by the TorchIO library (<https://torchio.readthedocs.io/>), which allows us to set the number of ghosts, its intensity, and in which axis the ghosts are spread. We define the magnitude of the ghosting artifact as a combination of

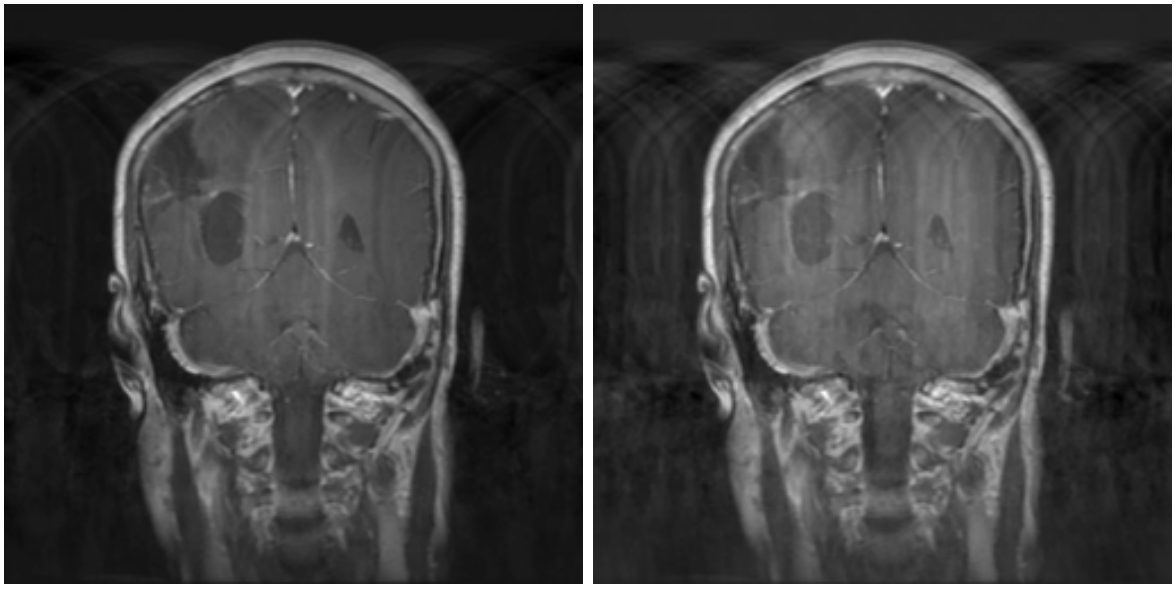


Figure 4.18: MRI with simulated ghosting artifact, with five ghosts on the left and 21 ghosts on the right.

the number of ghosts and the intensity of the ghosts, based on a magnitude parameter m:

$$n\_\text{ghosts} = 1 + 2 \cdot m$$

$$\text{intensity} = 0.3 + 0.16 \cdot m,$$

where m is always rounded down to an integer, since the number of ghosts has to be an integer. Fig. 4.18 shows examples of images with artificially added ghosting artifacts at magnitude, m, set to 2 and 10.

In the ghosting module, the magnitude m is sampled from a normal distribution characterized by a mean of 7 and a standard deviation of 6, constrained to the interval [3, 21], and consistently rounded down to the nearest integer.

4.1.7 Complete Artifact Generator

The complete artifact generation function is composed of a series of seven modules: Rician noise, Gaussian blur, Gibbs ringing, ghosting, high-contrast, low-contrast, and JPEG compression, which are applied sequentially in that order. This specific order was determined by empirical findings from experiments, theoretical insights into artifact generation, and logical assessments of how the degraded images were produced, as discussed in this section. Details of these empirical findings will be presented in the Experimental Section.

Given that the noise is affected by the other artifacts, Rician noise is the first artifact in the sequence. Gaussian blur and Gibbs ringing are generated by multiplicative low pass filter in the frequency domain and are, therefore applied next in the sequence. The ghosting artifact is associated with the conversion from the frequency to the spatial domain and is thus applied after ringing. The contrast artifact are associated with the spatial domain, so we apply the high contrast artifact after ghosting followed by the low contrast artifact as described in Section 4.1.5. Since the JPEG compression artifact is associated with image storage purposes, it is the last artifact added to the images.

The final parameter that must be set for each artifact module is the probability that the module will actually apply its designated artifact on the input image. We have noticed that not every image contains every type of artifact, and the probability parameter controls the frequency that the specified artifact will be applied. With 7 artifact modules in total and each module having the option to apply or not apply an artifact, there are 2^7 potential combinations of artifacts that could be applied to each image, ranging from none to all 7. Furthermore, since the intensity of each artifact is selected randomly, even if the same artifact combination is used more than once, the specific intensities are likely to vary, leading to a wide range of differing images from the same input, thus preventing overfitting.

To determine the probability for each artifact, we considered how often the artifact was observed in the datasets and how much the artifact affected diagnosis accuracy. In order to ensure the model learns to correct a variety of artifacts, we established a minimum probability of 10% while setting a maximum of 50% to prevent excessive degradation and to ensure that no single artifact type dominates the training process. Empirical results that will be shown in the experiment section show that contrast and specially ghosting have lower impact on diagnostic accuracy. Given that the ghosting artifact was also the least frequent in the Kaggle dataset, we allocated it the minimum probability of 10%, whereas the contrast artifact was assigned a probability of 20%. The most common artifact type in the dataset was a "thick"noise, which we have shown to be associated with noise followed by low pass filtering. Given that observation we set the probabilities for Rician noise, Gaussian blur, and Gibbs ringing at 50%. Probability for JPEG compression, being less prevalent, was set at 40%.

Algorithm 2 presents the entire artifact generation function, detailing the stated parameter

values and the sequence of modules. Figure 4.19 displays 16 instances of degraded images produced using this function from a single input image.

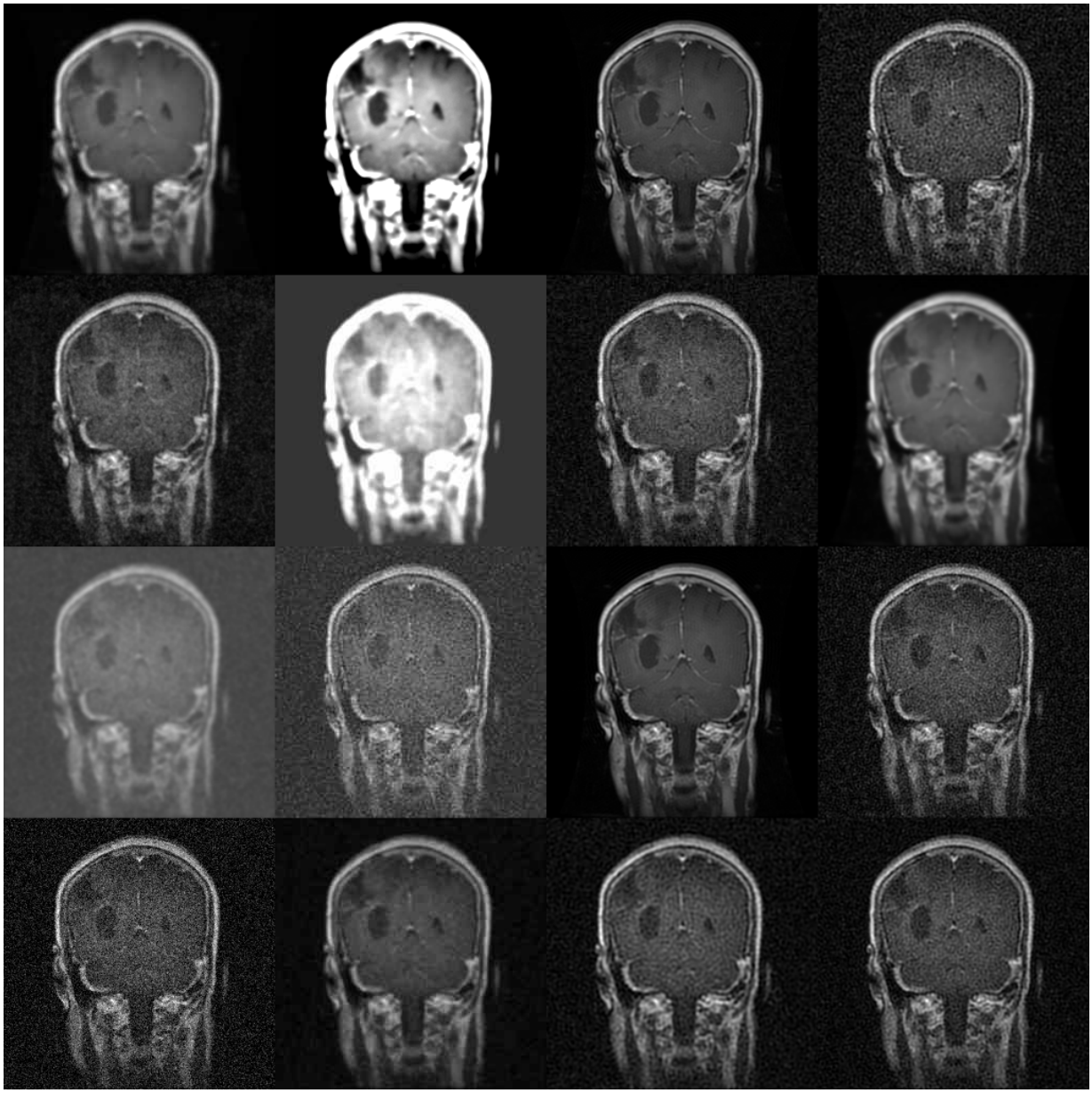


Figure 4.19: Samples of one MRI scan with artificially added artifacts by the artifact generator function

4.2 MODELS TRAINING

Two training procedures were proposed, one for the image restoration models and one for the image classification model.

Algorithm 2 Artifact generation function

```
procedure AddArtifacts(image)
  if random.uniform(0, 1) < 0.5 then
    intensity ← random.normal(mean = 0.0075, std = 0.02)
    intensity ← clip(intensity,min = 0.005, max = 0.125)
    image ← ricianNoise(image, magnitude = intensity)
  end if
  if random.uniform(0, 1) < 0.5 then
    intensity ← random.normal(mean = 1.25, std = 0.75)
    intensity ← clip(intensity,min = 0.015, max = 2)
    image ← gaussianBlur(image, magnitude = intensity)
  end if
  if random.uniform(0, 1) < 0.5 then
    intensity ← random.normal(mean = 80, std = 10)
    intensity ← clip(intensity,min = 40, max = 150)
    image ← gibbsRinging(image, magnitude = intensity)
  end if
  if random.uniform(0, 1) < 0.1 then
    intensity ← random.normal(mean = 7, std = 6)
    intensity ← clip(intensity,min = 3, max = 21)
    image ← ghosting(image, magnitude = intensity)
  end if
  if random.uniform(0, 1) < 0.2 then
    intensity ← random.normal(mean = 3, std = 2)
    intensity ← clip(intensity,min = 0, max = 8)
    image ← highContrast(image, magnitude = intensity)
  end if
  if random.uniform(0, 1) < 0.2 then
    intensity ← random.normal(mean = 2, std = 1)
    intensity ← clip(intensity,min = 0, max = 4)
    image ← lowContrast(image, magnitude = intensity)
  end if
  if random.uniform(0, 1) < 0.4 then
    intensity ← random.normal(mean = 70, std = 40)
    intensity ← clip(intensity,min = 20, max = 100)
    image ← JPEGCompression(image, magnitude = intensity)
  end if
  return image
end procedure
```

4.2.1 Restoration Model Training

We adapted the training strategy outlined in (WANG *et al.*, 2022) to train the image restoration models. The models were optimized using the AdamW optimizer, with momentum parameters set to (0.9, 0.999), and incorporated a weight decay of 0.02 along with gradient scaling. Given memory constraints, we configured the batch size to 4, meaning each training cycle utilized 4 images. In an effort to refine the training, a learning rate warm-up phase was included. This phase involved a linear increase of the learning rate from 0 to 4e-5 over the initial 10 epochs. Following the warm-up, we employed cosine annealing for learning rate decay down to a minimum of 1e-6 across 90 epochs, within a total of 100 training epochs. The models were trained using Charbonier loss, calculated between the restored and the original unmodified images.

In the pre-processing phase, images are resized to 256x256 pixels, which aligns with the Uformer models' requirement for sizes that are multiples of 128x128. Following the standard CIFAR data augmentation strategy (HE *et al.*, 2016), each image is then padded with 16 pixels on all sides, from which a random 256x256 crop is extracted. The image may also undergo a horizontal flip with a 50% chance. The resulting processed image serves as the target output for the restoration model. To create the corresponding degraded image for training input, an artifact generator function is employed online during training, meaning the degraded image is created by modifying the target image without being stored. A unique degraded image is generated per undegraded image at each epoch. Furthermore, due to the random cropping and potential horizontal flipping, target images can vary with each training epoch.

For the Uformer-T model, unless specified differently in the experiments, weights initialized using the pre-trained weights from the SIDD denoising task were employed.

4.2.2 Classification Model Training

For the training of image classification models, we employed the methodology outlined in (HE *et al.*, 2019). The optimization was performed using the SGD optimizer from the PyTorch library, incorporating Nesterov momentum set at 0.9 and a weight decay value of 1e-4. A

learning rate warmup strategy was adopted, beginning with a linear increase from 0 to 0.0125 over the initial 10 epochs, followed by a cosine annealing schedule reducing the learning rate to 0 across the subsequent 90 epochs, amounting to a total of 100 epochs. We utilized a batch size of 16 images, training the models with cross-entropy loss.

The classification models were trained using the same pre-processing techniques as the restoration models, as explained in section 4.2.1. However, unless specified differently in the experiments section, the input images were the original, non-degraded versions. The model weights were initialized using weights pre-trained on the ImageNet dataset, available through the Pytorch framework, with the exception of the classification layer, which was substituted with one tailored to the specific number of target classes.

CHAPTER 5

EXPERIMENTAL TESTS AND RESULTS

This chapter provides an overview of the experimental evaluations carried out to assess the performance of the proposed brain tumor diagnosis framework. This framework is specifically engineered to withstand the typical degradations encountered in MRI scans. More specifically, the experimental tests evaluate the diagnostic accuracy of the proposed approach focusing on the classification accuracy of the tumors. Each experiment examines the impact of incorporating the image restoration step versus omitting it.

5.1 DATASETS FOR TESTING THE PERFORMANCE OF THE FRAMEWORK

Our experimental tests are designed to assess the effectiveness of the proposed degradation-resistant framework for diagnosing brain tumors. These experiments evaluate the diagnostic accuracy of our approach by focusing on the classification accuracy of the tumors. Each experiment examines the impact of incorporating the image restoration step versus omitting it. With this goal, as detailed in the previous chapters, we generated datasets with different combinations of artifacts of different strength. We tested the proposed framework on these artificially generated datasets and on publicly available datasets.

5.1.1 Testing the Robustness to Artificial Artifacts

By intentionally introducing artifacts into the images, we can conduct controlled trials to evaluate how specific conditions, such as a particular artifact or level of degradation, influence the accuracy of the framework. Initially, to test the effectiveness of the framework in managing specific artifacts, we employ the Algorithm 3. This algorithm entails specifying an artifact or a set of artifacts, which are then applied to each image in the test set at an intensity level of 1, as detailed in Section 3.3, and then repeats the test for intensity values of 2 through 10. The

Mean Structural Similarity Index (MSSIM) is calculated between each degraded image and its original counterpart without artifacts, followed by computing the average MSSIM for the test set. The restoration model is applied to generate a recovered version of each degraded image, and the MSSIM between these recovered images and the original pristine images is assessed, and the average restored MSSIM is calculated in the same manner throughout the test set.

The classification model is applied to both degraded and non-degraded images, and the quantity of correctly classified images is recorded separately for both degraded and enhanced images. These figures are used to calculate the accuracy with and without the use of the restoration model. This procedure is repeated for artifact intensities ranging from 1 to 10. The function returns a list comprising MSSIM and accuracy metrics for both degraded and enhanced images, which can be visualized using graphs or averaged to determine individual metrics. This function dynamically generates degraded images during testing and, due to the randomness of the noise function, the results may exhibit slight variations with each run.

Algorithm 3 Testing accuracy and MSSIM for a given artifact type.

```
Test(artifact):
mssimList, mssimRestoredList, accuracyList, accuracyRestoredList
for magnitude = 1:10 do
    MSSIM = 0
    restoredMSSIM = 0
    correct = 0
    correctRestored = 0
    for all testImage, target do
        degradedImage = apply(testImage, artifact, magnitude)
        restoredImage = restoration(degradedImage)
        MSSIM = MSSIM + mssim(testImage, degradedImage)
        restoredMSSIM = MSSIM + mssim(testImage, degradedImage)
        correct = correct + classification(degradedImage, target)
        correctRestored = correctRestored + classification(restoredImage, target)
    end for
    MSSIM = MSSIM/testsetSize
    restoredMSSIM = restoredMSSIM/testsetSize
    accuracy = correct/testsetSize
    accuracyRestored = correctRestored/testsetSize
    mssimList.insert(MSSIM)
    mssimRestoredList.insert(restoredMSSIM)
    accuracyList.insert(accuracy)
    accuracyRestoredList.insert(accuracyRestored)
end for
return  mssimList, mssimRestoredList, accuracyList, accuracyRestoredList
```

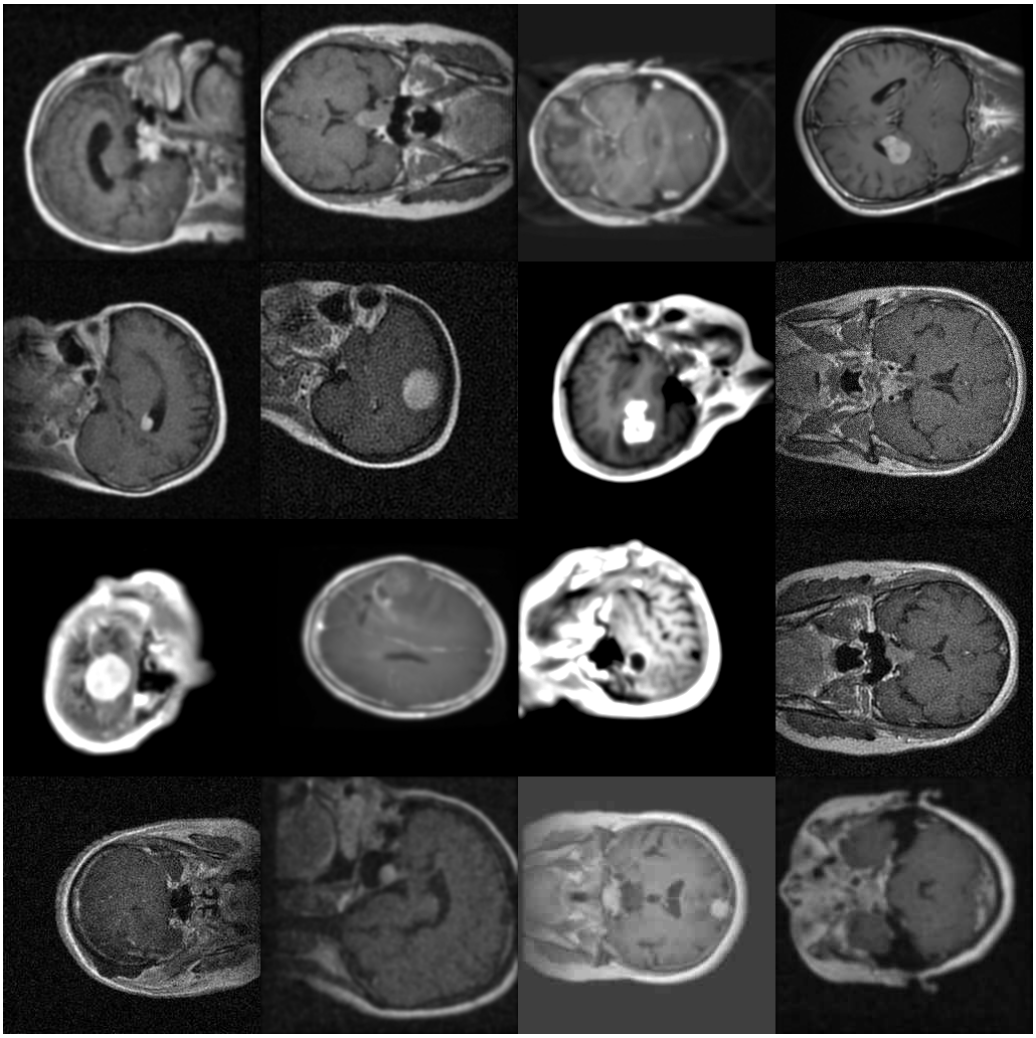


Figure 5.1: Samples of the artificially degraded test set.

Alongside Algorithm 3, we created an artificially degraded test set using the artifact generator function of Algorithm 2 on the test set sourced from the FigShare dataset (CHENG *et al.*, 2016). In addition to employing the image degradation function, we also used the pre-processing procedure described in Section 4.2.1 to produce random alterations of the test images. For each image in the pristine test set, 10 randomly degraded counterparts were generated, paired with the original images. This test set was designed for offline evaluation without introducing artifacts during testing. It comprises 5,650 image pairs, distributed as 890 pairs of meningioma tumors, 29,100 pairs of glioma tumors, and 1,850 pairs of pituitary tumors. Pairing of original and degraded images facilitates measuring the MSSIM metric, and separation by tumor type allows calculating the accuracy metric. This test set was labeled as "Degraded" test set in the experiments. Figure 5.1 presents 16 images sampled from this test set.

5.1.2 Testing the Robustness to Public Datasets

In addition to assessing our framework with artificially introduced artifacts, we also evaluated its performance using the test sets of the datasets described in Section 3.8. Since these datasets lack a reference image, we only measured accuracy, not MSSIM. Various degraded images were detected across many datasets, with no explicit information regarding the nature of these degradations. We assume these were not intentionally introduced, except for the JPEG compression artifacts. We conducted two types of tests on these datasets: in-dataset and cross-dataset. In-dataset testing involved training a classification model with a dataset's training set and evaluating it on its test set. Cross-dataset testing involved training a model on one dataset's training set and testing it across multiple dataset test sets, necessitating adaptations to align their classes. We performed two types of cross-dataset tests, based on the classes involved:

- Tumor type: The classes are meningioma, glioma, and pituitary. The datasets involved are Figshare, "Degraded", Kaggle, and Zenodo. The class "No tumor" was excluded from the Kaggle dataset, and the class "adenoma" was excluded from the Zenodo dataset.

- Tumor/No tumor: The classes consist of "tumor" and "No Tumor". The datasets used include Siar, Br35H, and Kaggle, where the "meningioma", "glioma", and "pituitary" classes from Kaggle were consolidated into a single "tumor" class.

5.2 PERFORMANCE TESTS

The evaluation of the proposed framework's performance involved two key metrics: Mean Structural Similarity Index (MSSIM) for image quality assessment and the Accuracy metric for tumor classification. We examined different parts of the framework on images both with and without MRI artifacts, with each test highlighting the significance of each stage. The experiments are structured as follows:

1. Correlation between MSSIM and Accuracy: Assesses how well the MSSIM metric reflects diagnosis quality based on the nature of image degradation.

2. Classification training: Examines the effect of training classification algorithms both with and without image degradations, as well as the influence of the severity of these image artifacts.

3. Artifact Generator: Assesses the impact of various elements of the artifact generator on the restoration model's performance. The elements examined include the selection of artifacts, strategies for training a model with multiple artifacts, the sequence of artifact application, and the likelihood of each artifact being employed.

4. Restoration Model: Assesses the impact of employing various restoration models.

5. Restoration dataset: Assesses the effect of utilizing diverse datasets for training the restoration model

6. Classification dataset: Evaluates the influence of using different datasets to train the classification model

7. Classification model: Evaluates the impact of using different classification neural network models to classify brain tumor in magnetic resonance images.

5.2.1 Accuracy x MSSIM Correlation

Rodrigues *et al.* emphasizes that the assessment of medical image quality ought to prioritize its diagnostic utility (RODRIGUES *et al.*, 2022). The most reliable metric for diagnostic utility is the precision of tumor classification using our classification models, whereas we assess image quality using the MSSIM metric. In this study, we investigated the relationship between the MSSIM image quality metric and the accuracy of tumor classification models, considering the impact of different types of image artifacts that degrade image quality.

In our experiment, we used the Fighshare dataset (CHENG *et al.*, 2016) along with the Algorithm 3. For image classification, we implemented EfficientNet-b0, and for image restoration, we employed Uformer-T. The testing algorithm produced two groups of 10 data points for Accuracy versus MSSIM: one corresponding to the degraded images and the other to their restored versions. We calculated both the Spearman (SPEARMAN, 1904; ZAR, 2005) and

Table 5.1: Correlation between MSSIM and model accuracy. Classification model trained on clean, non-degraded images. Restoration model trained on images with random artifacts.

Artifact	MSSIM x Accuracy Correlation					
	Degraded		Restored		Both	
	Spearman	Pearson	Spearman	Pearson	Spearman	Pearson
Ghosting	0.972	0.768	0.907	0.942	0.288	0.207
Contrast	0.990	0.663	0.715	0.752	0.704	0.615
Noise	0.836	0.837	0.936	0.958	0.965	0.935
Ringing	0.990	0.963	0.990	0.982	0.974	0.943
Blur	0.997	0.994	1.0	0.980	0.988	0.989
All	0.921	0.982	1.0	0.992	0.985	0.989

Pearson (FREEDMAN *et al.*, 2007) correlation coefficients of these data samples, initially focusing on the degraded ones, then the restored ones, and ultimately considering the complete set of 20 data points, yielding a total of six correlation values. This analysis was repeated for every artifact type listed in Algorithm 3. The experiment included five types of artifacts: ghosting, low contrast, noise, ringing, and blur. Furthermore, we conducted a test that applied the five artifact types to each image. The results of the experiment are detailed in Table 5.1.

The analysis of the data suggests that there is a significant relationship between the MSSIM metric and the precision of CNN-based diagnoses, with most of the entries in Table 5.1 showing correlation values greater than 0.5. However, this relationship varies depending on the type of image degradation, with ghosting and contrast artifacts resulting in a lower correlation than other types. For columns focusing on degraded or restored images, this correlation can be interpreted as one between the degradation intensity and accuracy, which are the variables altered to derive these values. In columns examining both degraded and restored images, the correlation also reflects the impact of image restoration on diagnosis accuracy. Here, ghosting and contrast artifacts exhibit the weakest correlation, with ghosting displaying values below 0.5. Such reduced correlation levels contribute to the low probability of these artifacts in the artifact generator function outlined in Section 4.1.7. We will enter into more detail as to why these two artifacts have less impact on the diagnostic performance in Section 5.2.3.4.

5.2.2 Classification Training

In this experiment, we evaluate the performance of two methods to train classification models to be tested on images with degradations: with and without image degradations. Two

EfficientNet-b0 models were trained, the first on the clean Figshare dataset, without image degradation. For the second model, the same dataset was used; however, the artifact generator function was used to degrade the images online, similar to the procedure to train the image restoration models described in Section 4.2.1. We refer to the first model as the "clean train" and the second model as the "degraded train." With the exception of added image degradations, both models followed the same training procedure described in Section 4.2.2.

In this experiment, we evaluated the accuracy of tumor classification, depending on which artifact affects the image and what the magnitude of that artifact is using Algorithm 3. The quality of the images affected by different artifact types and intensities was measured on the basis of the MSSIM metric, and the accuracy of classification models tested on those images was also tested. Similarly to the previous experiment, the artifacts were ghosting, low contrast, noise, ringing, blur, and all five artifacts. For each artifact, four sets of results are obtained: "clean train" with and without image restoration; and "degraded train" with and without restoration. We refer to "degraded eval" when accuracy is measured on degraded images without restoration and to "restored eval" when the models are tested on restored images. We report the accuracy results as a function of the MSSIM image quality of the degraded test set, so the four sets of results are aligned. The results are shown graphically in Fig. 5.2.

In reviewing all six graphs, it is clear that the model trained on pristine images shows a notable decline in accuracy when tested on degraded images, indicating a direct correlation between image quality and accuracy. Noise was the artifact that most significantly impacted accuracy and image quality, followed by contrast. Ghosting had the least influence on accuracy, maintaining a level above 80%, whereas blurring had a less pronounced effect on image quality.

An intriguing phenomenon is noticed on the graph in the presence of five concurrent artifacts. When the artifact intensity surpasses 7, accuracy continues to decline, while MSSIM exhibits a slight rise. At such high magnitudes of degradation, images tend to focus on their low-frequency components, causing a loss of structural patterns. This results in missing information necessary for accurate image classification, although it reduces structural differences at smaller values.

The six graphs illustrate the beneficial effect of image restoration before tumor classification. The model initially trained on undegraded images exhibits the greatest increase in accuracy when images are restored, as shown in 100% of the artifact conditions. Although the model

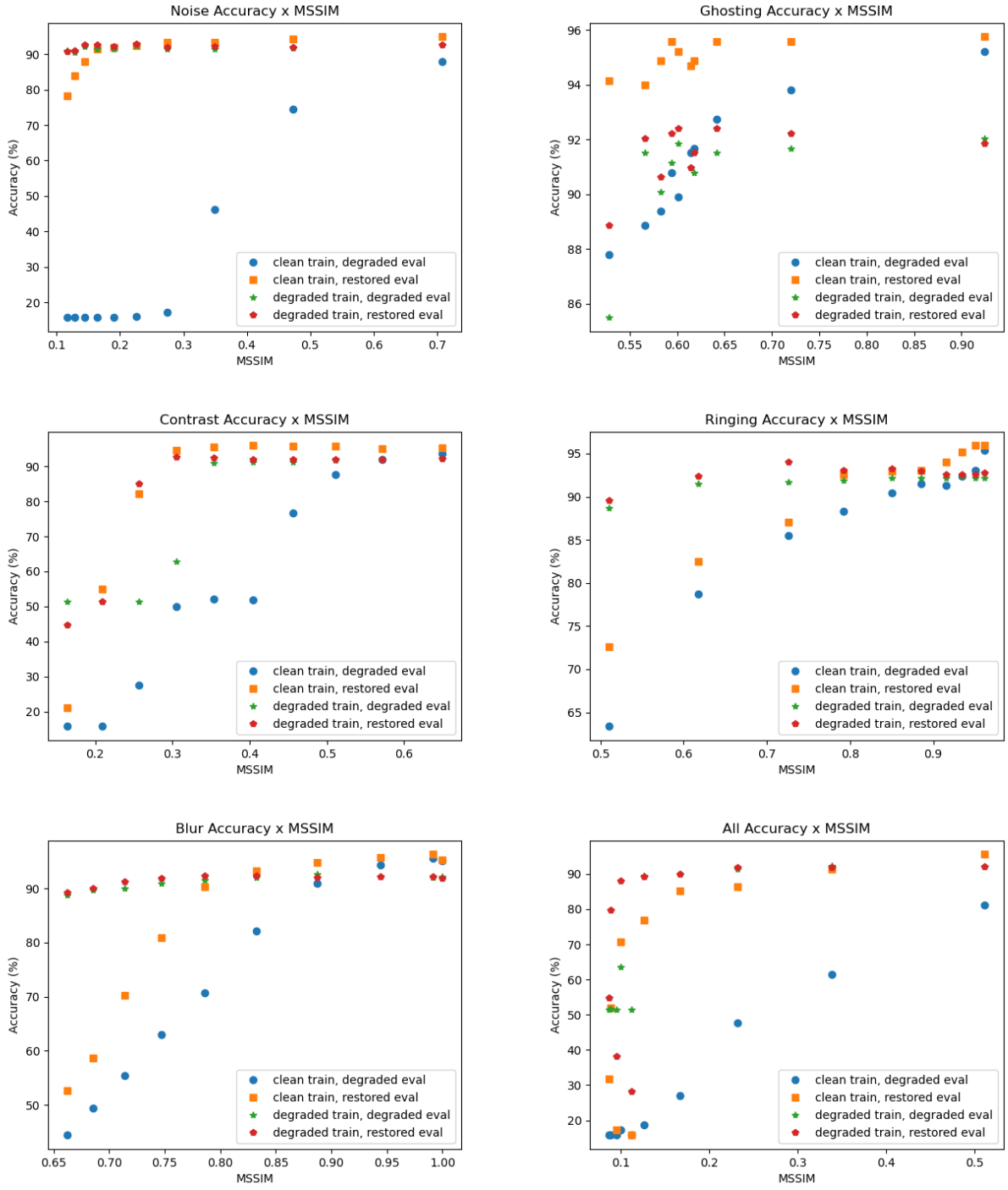


Figure 5.2: Accuracy x degraded image MSSIM, depending on the image degradation type. Two classification models are tested with and without the image restoration, one classification model trained on undegraded images and the other trained on degraded images.

trained in degraded images operates under conditions similar to the test environment, it also gains from restoration, with the "degraded train, restored eval" data points achieving higher accuracy compared to "degraded train, degraded eval" in 90% of the scenarios.

The restored accuracy exhibits two distinct trends based on the type of artifact. For noise,

contrast, and ghosting, the accuracy remains stable during image restoration as quality declines, until a certain point is reached, beyond which the accuracy sharply falls with more severe image degradation. In the case of ringing and blurring, although restoration enhances accuracy, a decline in image quality still results in a proportional decrease in accuracy. This can be attributed to the fact that both artifacts are produced by a low-pass filter, indicating that recovering the high-frequency information is a challenge for the model. Additionally, another factor contributing to this outcome is that Algorithm 2 employs lower intensity ringing and blur for training, which may not be as effective when dealing with stronger artifacts of these types.

In evaluating the performance of the two classification models, it is evident that the model trained with degraded images demonstrates superior robustness to image quality reductions, maintaining higher accuracy when the quality declines, even when both models are assessed on enhanced images. In contrast, the model trained with clean images achieves the highest accuracy across all graphs when the degradation is minimal. This suggests a compromise in training with image artifacts, trading some maximum accuracy for greater resilience against these artifacts. Based on the graphs we infer that the best-performing model varies based on the specific artifact and its intensity affecting the images, though using image restoration techniques is generally advisable.

5.2.3 Artifact Generator

An essential component in training a restoration model is the artifact generation function. Utilizing distinct functions to create the degradations results in varied model training outcomes. This section investigates the effects of altering various facets of the artifact generator function to empirically assess what contributes to an enhanced performance in image restoration models.

Algorithm 3 is once again used to evaluate the models. For each artifact, we report the average accuracy and MSSIM of the 10 intensities. The test artifacts are ghosting, low contrast, noise, ringing, blur, ringing followed by noise, and all five artifacts in the order blur, ringing, ghosting, noise and low contrast. The results are shown in the form of two tables, one for MSSIM and one for accuracy. In addition to restoration results, we also add the degraded

MSSIM and accuracy as reference.

In this section, rather than sampling artifact magnitudes from a normal distribution during training, we randomly select the intensity within the range [1,10], utilizing the algorithm described in Section 3.3 to create the artifacts. This section is divided into subsections, with each one examining a specific aspect of the artifact generator function: which artifacts are incorporated, how to conduct training with multiple artifacts, the sequence of artifact application, and the likelihood of each artifact being applied.

An EfficientNet-b0 model was utilized to train on original images in order to determine the accuracy outcome. For each variant of the artifact generator, three Uformer-T models were trained. The average and standard deviation among these models are presented for both accuracy and MSSIM.

5.2.3.1 Artifact Choice

In this section, we evaluate the impact of choosing which artifacts will be included in the training process. In this experiment, five artifact types were considered: ghosting, low contrast, ringing, blur, and noise. We established five distinct artifact generators, each corresponding to a single artifact type, as well as a sixth generator that randomly selects and applies one of these five artifacts to an image, with each type having an equal chance of selection. In this experiment, only one artifact type may be applied to each training image. Table 5.2 presents the experimental outcomes for MSSIM, while Table 5.3 displays the accuracy results.

From Tables 5.2 and 5.3, it is evident that models specialized in a single artifact exhibit superior performance when assessed solely on that particular artifact compared to models trained for various artifacts. Yet, the multi-artifact model excels when images contain multiple artifact types and ranks second for each individual artifact. This finding suggests that training a unified model for handling multiple artifacts is optimal, especially when the specific artifact affecting an image is undetermined or when multiple artifacts are present simultaneously.

An additional insight from the findings is that models trained to correct a particular artifact also enhance image quality and classification accuracy on images compromised by different artifacts, although blurring is an exception. Blurring serves to eliminate both noise and ringing

Table 5.2: Average MSSIM obtained by each restoration model for each artificial artifact. R+N indicates ringing followed by noise. Mean and standard deviation of 3 runs. Best result of each column is in bold.

Train Artifact	Test Artifact MSSIM							
	None	Ghosting	Contrast	Noise	Ringing	Blur	All	R+N
No Restoration	**1** ±0.0	0.685 ±0.0	0.540 ±0.0	0.350 ±0.008	0.830 ±0.0	0.840 ±0.0	0.296 ±0.003	0.323 ±0.012
Ghosting	0.998 ±0.0	**0.965** ±0.001	0.716 ±0.001	0.386 ±0.004	0.834 ±0.001	0.806 ±0.001	0.304 ±0.008	0.355 ±0.004
Contrast	0.998 ±0.0	0.704 ±0.001	**0.991** ±0.0	0.371 ±0.001	0.834 ±0.0	0.805 ±0.0	0.300 ±0.001	0.341 ±0.002
Noise	0.997 ±0.0	0.684 ±0.0	0.718 ±0.001	**0.889** ±0.0	0.834 ±0.001	0.803 ±0.001	0.467 ±0.001	0.823 ±0.001
Ringing	0.997 ±0.0	0.690 ±0.001	0.683 ±0.006	0.512 ±0.022	**0.895** ±0.0	0.808 ±0.001	0.291 ±0.003	0.502 ±0.025
Blur	0.998 ±0.0	0.685 ±0.001	0.677 ±0.021	0.350 ±0.001	0.801 ±0.002	**0.931** ±0.001	0.245 ±0.002	0.325 ±0.002
Choice	0.992 ±0.001	0.952 ±0.001	0.987 ±0.0	0.881 ±0.001	0.885 ±0.001	0.911 ±0.001	**0.553** ±0.001	**0.825** ±0.001

effects, suggesting that these models somewhat "learn" to blur images for quality enhancement. In contrast, the deblurring model might focus on sharpening images, thereby intensifying other artifacts. With regard to other artifacts, it appears that the models learn not only to eliminate specific issues but also to recognize the appearance of pristine images, thus tending to address additional out-of-distribution image degradations.

In training models for individual artifacts, the noise-trained model emerged as the top performer under the "All" training scenario. This can be attributed to the fact that noise critically affects image quality and classification accuracy both pre- and post-restoration; consequently, eliminating noise from images is of paramount importance. This outcome supports the rationale behind assigning noise the highest probability in Algorithm 2.

5.2.3.2 Artifact Combination

The prior experiment demonstrated that a single model can be effectively trained to manage various image artifact types, achieving the highest median performance across these types compared to models specialized for individual artifact types. In this section, we evaluate the most effective strategies for incorporating various artifact types into the training set. The

Table 5.3: Average accuracy obtained by the EfficientNet model when presented with images degraded with different artifacts and restored by different models. R+N indicates ringing followed by noise. Mean and standard deviation of 3 runs. Best result in each column is in bold.

Train Artifact	Test Artifact Accuracy (%)							
	None	Ghosting	Contrast	Noise	Ringing	Blur	All	R+N
No Restoration	95.9 ±0.0	93.6 ±0.0	89.2 ±0.0	49.8 ±0.3	86.0 ±0.0	85.2 ±0.0	42.9 ±0.1	54.5 ±0.9
Ghosting	**96.0** ±0.1	**95.4** ±0.1	86.1 ±0.7	44.3 ±0.3	86.3 ±0.1	85.4 ±0.1	44.4 ±0.4	50.2 ±0.5
Contrast	**96.0** ±0.1	94.3 ±0.0	**95.6** ±0.1	49.4 ±0.2	86.4 ±0.0	85.6 ±0.0	45.7 ±0.1	56.6 ±0.2
Noise	95.7 ±0.2	94.3 ±0.1	91.6 ±0.1	**88.3** ±0.1	86.3 ±0.0	85.4 ±0.1	69.0 ±0.5	**83.3** ±0.1
Ringing	95.9 ±0.0	94.3 ±0.1	91.2 ±0.3	50.0 ±1.2	**88.5** ±0.1	85.6 ±0.1	44.6 ±0.1	53.2 ±1.9
Blur	**96.0** ±0.1	94.3 ±0.1	91.1 ±0.1	46.8 ±0.7	83.4 ±0.2	**90.8** ±0.1	41.6 ±0.2	55.3 ±0.2
Choice	95.6 ±0.1	95.1 ±0.1	95.4 ±0.1	86.6 ±0.5	86.9 ±0.1	89.7 ±0.1	**72.9** ±0.1	**83.3** ±0.1

methods evaluated are the following.

- Choice: From a selection of potential artifact types, an artifact is randomly selected for each image and applied to it.

- Parallel: Five instances of the image are created in parallel, each subjected to one of five artifact types: ghosting, blurring, ringing noise, and low contrast. Finally, these five altered images are averaged to produce a single image that integrates all five artifacts.

- Sequential: The five artifacts are applied to the image in sequential order. In this particular case, the order is blurring, ringing, ghosting, noise, and contrast. Each artifact has 50% probability of being applied. Similar to algorithm 2.

Tables 5.4 and 5.5 show the MSSIM and accuracy, respectively, for this experiment.

The parallel method showed considerably worse performance than the other two methods and, for this reason, was only trained once, instead of 3 times, like the other two. By averaging five images, in which only one of the five has a specific artifact, the impact of each artifact is reduced in the images. It has five artifacts affecting the image, but each artifact is at 1/5

Table 5.4: Average MSSIM obtained by each restoration model for each artificial artifact. R+N indicates ringing followed by noise. Mean and standard deviation of 3 runs. Best result of each column is in bold.

Method	Test Artifact MSSIM							
	None	Ghosting	Contrast	Noise	Ringing	Blur	All	R+N
No Restoration	**1** ±0.0	0.685 ±0.0	0.540 ±0.0	0.350 ±0.008	0.830 ±0.0	0.840 ±0.0	0.296 ±0.003	0.323 ±0.012
Choice	0.992 ±0.001	**0.952** ±0.001	**0.987** ±0.0	**0.881** ±0.001	**0.885** ±0.001	**0.911** ±0.001	0.553 ±0.001	0.825 ±0.001
Parallel	0.894	0.827	0.788	0.785	0.782	0.762	0.473	0.720
Sequential	0.996 ±0.0	0.945 ±0.001	0.978 ±0.001	0.874 ±0.0	0.881 ±0.0	0.868 ±0.001	**0.788** ±0.0	**0.834** ±0.001

Table 5.5: Average accuracy obtained by the EfficientNet model when presented with images degraded with different artifacts and restored by different models. R+N indicates ringing followed by noise. Mean and standard deviation of 3 runs.

Method	Test Artifact Accuracy (%)							
	None	Ghosting	Contrast	Noise	Ringing	Blur	All	R+N
No Restoration	**95.9** ±0.0	93.6 ±0.0	89.2 ±0.0	49.8 ±0.3	86.0 ±0.0	85.2 ±0.0	42.9 ±0.1	54.5 ±0.9
Choice	95.6 ±0.1	**95.1** ±0.1	**95.4** ±0.1	**86.6** ±0.5	**86.9** ±0.1	**89.7** ±0.1	72.9 ±0.1	83.3 ±0.1
Parallel	95.7	93.3	91.3	70.4	85.8	86.3	54.7	71.5
Sequential	95.3 ±0.0	93.9 ±0.1	94.1 ±0.1	85.1 ±0.1	**86.9** ±0.1	87.0 ±0.1	**78.1** ±0.2	**83.4** ±0.1

of image intensity; consequently, the model has less capability at restoring images when the artifact impacts an image at full intensity.

The "choice" model has the best MSSIM and accuracy values for tests with single artifact types, as they are more similar to the training conditions in which these restoration models were trained. However, only a second artifact that affects the image is enough for the "choice" model to have lower accuracy than the "sequential" model, as shown by the result of the ringing + noise (R + N). The difference becomes more significant when all 5 artifacts are applied.

The results of this experiment are similar to those in Section 5.2.3.1. Models perform best when tested under the same conditions in which they were trained. In contrast, for a given test condition, the best model would be the one trained in that same condition. Additionally, increasing the number of conditions for which a model is trained leads to a lower restoration quality in each of these conditions, as the model becomes less specialized in specific degradation conditions but generalizes better for more degradation conditions. The "choice" model is trained

for 5 degradation conditions (each individual artifact), while the "sequential" model is trained for 32 degradation conditions (2^5 combinations of 5 artifacts in the specified order), so the "choice" model has the best performance on those 5 conditions, while the "sequential" model is better in the other 27 conditions.

Defining the best training generator is dependent on the expected degradations for testing and practical purposes. If only noise was expected, the "noise" model of Table 5.3 would be the best model. If more than one type could affect an image, but only one artifact type affects an image, the "choice" model would be the best. However, if an image can be affected by more than one artifact type, the "sequential" model would be the best. Evaluations of the Kaggle dataset in Section 4.1 have shown that the most common degradation was noise followed by ringing and/or blur, often also followed by poor contrast artifacts. For our testing conditions, in which we want to cover several artifact combinations, the best model is the sequential model.

5.2.3.3 Artifact Order

In the previous section, the "sequential" model is trained applying artifacts in a specific order. In this experiment, we want to evaluate the effect of changing this order. Each artifact has a 50% of being applied to each image and are applied considering a specific order. The evaluated training artifact orders are:

- BRGNC: Artifacts are applied in the order blurring, ringing, ghosting, noise, and low contrast.
- NCRGB: Artifacts are applied in the order noise, low contrast, ringing, ghosting, and blurring.
- GBRNC: Artifacts are applied in the order ghosting, blurring, ringing, noise, and low contrast.
- Random: for each image, the order of the artifacts is chosen at random.

Tables 5.6 and 5.7 show the MSSIM and accuracy result, respectivelly, for this experiment. Tables 5.6 and 5.7 show that the ability to remove ghosting is largely unaffected by the position

Table 5.6: Average MSSIM obtained by each restoration model for each artificial artifact. R+N indicates ringing followed by noise. Mean and standard deviation of 3 runs. Best result of each column is in bold.

Order	Test Artifact MSSIM							
	None	Ghosting	Contrast	Noise	Ringing	Blur	All	R+N
No Rest.	**1** ±0.0	0.685 ±0.0	0.540 ±0.0	0.350 ±0.008	0.830 ±0.0	0.840 ±0.0	0.296 ±0.003	0.323 ±0.012
BRGNC	0.996 ±0.0	**0.945** ±0.001	**0.978** ±0.001	0.874 ±0.0	0.881 ±0.0	0.868 ±0.001	**0.788** ±0.0	**0.834** ±0.001
NCRGB	0.995 ±0.0	0.944 ±0.0	0.842 ±0.001	0.872 ±0.0	**0.884** ±0.0	**0.909** ±0.0	0.649 ±0.001	0.818 ±0.0
GBRNC	0.995 ±0.0	**0.945** ±0.0	**0.978** ±0.0	**0.875** ±0.001	0.876 ±0.001	0.862 ±0.001	0.779 ±0.001	0.832 ±0.001
Random	0.995 ±0.0	0.940 ±0.0	0.960 ±0.001	0.871 ±0.001	0.883 ±0.0	0.890 ±0.001	0.772 ±0.001	0.831 ±0.001

Table 5.7: Average accuracy obtained by the EfficientNet model when presented with images degraded with different artifacts and restored by different models. R+N indicates ringing followed by noise. Mean and standard deviation of 3 runs.

Order	Test Artifact Accuracy (%)							
	None	Ghosting	Contrast	Noise	Ringing	Blur	All	R+N
No Restoration	**95.9** ±0.0	93.6 ±0.0	89.2 ±0.0	49.8 ±0.3	86.0 ±0.0	85.2 ±0.0	42.9 ±0.1	54.5 ±0.9
BRGNC	95.3 ±0.0	93.9 ±0.1	94.1 ±0.1	85.1 ±0.1	86.9 ±0.1	87.0 ±0.1	78.1 ±0.2	83.4 ±0.1
NCRGB	95.0 ±0.1	**94.1** ±0.1	83.0 ±0.5	83.7 ±0.1	**87.3** ±0.1	**89.3** ±0.1	65.3 ±0.4	80.2 ±0.4
GBRNC	95.5 ±0.2	**94.1** ±0.1	**94.2** ±0.1	**85.7** ±0.2	86.6 ±0.1	87.0 ±0.1	**78.3** ±0.3	**84.1** ±0.4
Random	95.0 ±0.1	93.9 ±0.1	93.0 ±0.1	83.7 ±0.2	86.6 ±0.1	88.8 ±0.2	76.0 ±0.3	81.5 ±0.2

at which this artifact is introduced, as the three tested positions produced roughly similar results.

In contrast, contrast restoration appears to be sensitive to order; if it is not the last artifact applied, the quality and accuracy of the restored image suffer notably. The best results for ringing and blurring restoration are achieved when these artifacts follow the noise application. Among the models tested, the GBRNC demonstrated superior performance, particularly in terms of classification accuracy. It should be mentioned that this model was trained using an order distinct from the "All" testing order, but achieved the highest accuracy for that testing scenario.

Table 5.8: Average MSSIM obtained by each restoration model for each artificial artifact. R+N indicates ringing followed by noise. Mean and standard deviation of 3 runs. Best result of each column is in bold.

GC prob.	Test Artifact MSSIM							
	None	Ghosting	Contrast	Noise	Ringing	Blur	All	R+N
No Restoration	**1** ±0.0	0.685 ±0.0	0.540 ±0.0	0.350 ±0.008	0.830 ±0.0	0.840 ±0.0	0.296 ±0.003	0.323 ±0.012
50%	0.995 ±0.0	**0.945** ±0.0	**0.978** ±0.0	0.875 ±0.001	0.876 ±0.001	0.862 ±0.001	**0.779** ±0.001	0.832 ±0.001
25%	0.996 ±0.0	0.938 ±0.001	0.977 ±0.001	**0.877** ±0.001	**0.878** ±0.0	**0.869** ±0.001	0.770 ±0.002	**0.833** ±0.001

The "Random" model is designed to accommodate any sequence of artifact applications, leading to lower performance in particular situations but offering better adaptability to numerous artifact conditions applied in various sequences. Nonetheless, the assessments in Section 4.1 indicate an optimal sequence exists for artifact application on images. Hence, within this context, we regard the GBRNC model as the preferred choice.

5.2.3.4 Artifact Probability

In earlier experiments, sequential models had an equal probability of 50% for the application of each artifact. This experiment investigates how altering the likelihood of certain artifacts being applied affects the results. In Table 5.1, ghosting and contrast exhibited the weakest correlations between MSSIM and accuracy. Furthermore, in preceding experiments, ghosting and contrast emerged as the testing artifacts with the highest accuracy and MSSIM following image restoration. Considering these insights, we suggest that ghosting and contrast need not be incorporated as frequently as other artifact types during training. For this experiment, we developed two restoration models: one where all artifacts have a 50% likelihood of being incorporated, and another where ghosting and contrast have only a 25% chance, while the other artifact types retain a 50% chance. Tables 5.8 and 5.9 present the MSSIM and accuracy results for this experiment, correspondingly.

Tables 5.8 and 5.9 indicate that reducing the likelihood of contrast and ghosting artifacts indeed compromises the model's performance against these specific types. However, there is a notable increase in accuracy in all testing scenarios that involve the noise artifact. Hence, re-

Table 5.9: Average accuracy obtained by the EfficientNet model when presented with images degraded with different artifacts and restored by different models. R+N indicates ringing followed by noise. Mean and standard deviation of 3 runs. Best result in each column is in bold.

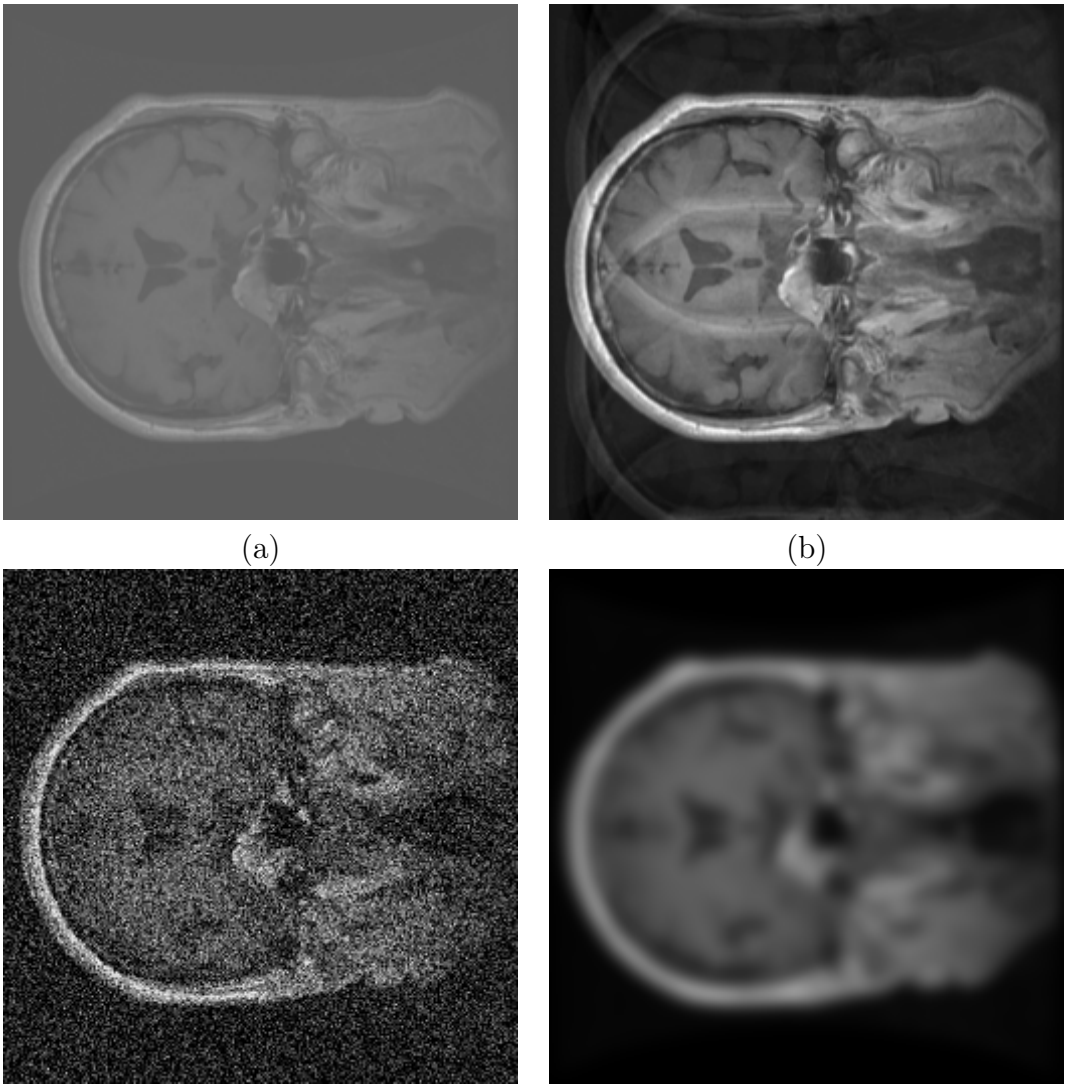
GC prob.	Test Artifact Accuracy (%)							
	None	Ghosting	Contrast	Noise	Ringing	Blur	All	R+N
No Restoration	**95.9** ±0.0	93.6 ±0.0	89.2 ±0.0	49.8 ±0.3	86.0 ±0.0	85.2 ±0.0	42.9 ±0.1	54.5 ±0.9
50%	95.5 ±0.2	**94.1** ±0.1	**94.2** ±0.1	85.7 ±0.2	86.6 ±0.1	**87.0** ±0.1	78.3 ±0.3	84.1 ±0.4
25%	95.6 ±0.1	93.9 ±0.1	94.1 ±0.1	**86.5** ±0.1	**86.7** ±0.1	86.8 ±0.1	**79.6** ±0.1	**85.0** ±0.1

ducing the occurrence of contrast and ghosting artifacts presents an advantageous compromise. The results of this experiment partly justify the probability settings in Algorithm 2.

We qualitatively evaluate the reason why contrast and ghosting have a lower correlation with diagnostic accuracy in Fig. 5.3. We can see that the contrast, ghosting, and noise artifacts have global effects on images, even affecting the background. As a consequence, images with these three artifacts have the lowest image quality measured by MSSIM. However, in Figs. 5.3 (a) and (b) we observe that the ghosting and contrast artifacts still preserve details in the images and consequently do not have as much impact in diagnostic accuracy, since details used to classify the images are still preserved. For subsequent experiments, this artifact generator, as described in the Algorithm 2, will be employed.

5.2.4 Restoration model

Among the components of the framework described in Fig. 4.1 is the restoration model. In earlier tests, the Uformer-T model served as the restoration model. This experiment examines the effectiveness of different restoration models. The Figshare dataset was used to train the models, with the artifact generator 2. To test the models, we first evaluate the MSSIM quality metric in the artificially degraded test set. In addition, we evaluated how much the restoration improved diagnostic accuracy. For this, we trained an EfficientNet-b0 on the Figshare dataset for tumor classes “meningioma”, “glioma”, and “pituitary.” This classification model is then tested on the “Degraded” test set, on the Figshare test set, on the Kaggle test set, and on the Zenodo test set. All test sets were adapted to only included the three defined tumor



(a) (b)

(c) (d)

Figure 5.3: Comparison of samples with different artifacts types. (a) Sample with contrast artifact. (b) Sample with ghosting. (c) Sample with noise. (d) Sample with blurring.

classes. Since both classification and restoration models were trained on the FigShare, but testing includes 4 test sets, this experiment also evaluates the generalization capability from one dataset to another.

The restoration models evaluated were an Autoencoder(EL-SHAFAI *et al.*, 2022b), a residual CNN(ZHONG *et al.*, 2020a), a residual UNet (MUCKLEY *et al.*, 2021) and the Uformer-T(WANG *et al.*, 2022). All models were trained from scratch. Additionally, we also trained an Uformer-T model using the pre-trained weights on the SIDD dataset as weight initialization. The results are shown in Table 5.10.

For models trained from scratch, the Residual UNet model demonstrated superior perfor-

Table 5.10: Comparison of different restoration models, measured on MSSIM on the degraded undegraded pair and accuracy of an EfficientNet-b0 model on 4 test sets. Results are mean and standard deviation of 3 runs. Best result for each column is in bold and second best is underlined.

Restoration Model	Params	MSSIM	Figshare	Degraded	Kaggle	Zenodo
No Restoration	**0**	0.596	95.38	78.39	<u>85.21</u>	42.73
Autencoder (EL-SHAFAI *et al.*, 2022b)	<u>0.20M</u>	0.692 ±0.001	91.06 ±0.22	83.04 ±0.18	62.69 ±0.54	31.65 ±0.51
Residual CNN (ZHONG *et al.*, 2020a)	0.56M	0.697 ±0.009	<u>95.97</u> ±0.14	84.85 ±0.58	79.40 ±1.86	38.60 ±1.97
Uformer-T (WANG *et al.*, 2022)	5.23M	<u>0.844</u> ±0.001	**96.14** ±0.07	<u>91.05</u> ±0.16	83.11 ±0.16	42.92 ±0.90
Residual UNet (MUCKLEY *et al.*, 2021)	7.76M	0.613 ±0.001	95.34 ±0.03	79.83 ±0.09	85.17 ±0.14	<u>43.11</u> ±0.14
pre-trained Uformer-T (WANG *et al.*, 2022)	5.23M	**0.889** ±0.004	95.36 ±0.04	**92.78** ±0.12	**87.71** ±0.14	**45.74** ±0.23

mance on the Kaggle and Zenodo datasets. In contrast, the Uformer-T model excelled with the Figshare and "Degraded" datasets and achieved the highest MSSIM metric, whereas the Residual UNet showed the lowest MSSIM value. Although the Residual UNet has the most parameters, making it the largest model, the Uformer is slower due to the computational intensity of its attention layers, despite having fewer parameters compared to simpler convolutional blocks.

An interesting result is in the Figshare test. This dataset is formed by undegraded images, yet the Residual CNN and Uformer-T models showed improvements in accuracy in that data set. Potentially, these models made the images more similar to the Figshare training set. The Residual UNet and pre-trained Uformer-T obtained accuracies on the Figshare similar to not using image restoration, which might be considered positive, as the restoration models probably learned to not change much of the test images, which are mostly undegraded. However, when comparing the "Residual UNet" line with the "No Restoration"' we observe however that the model barely changed the values in every column. So, contrary to the pre-trained Uformer-T, the residual CNN model did not learn to not restore good quality images; instead it learned to not restore or barely restore every image.

For the Kaggle column we observe that the pre-trained Uformer-T model was the only model that improved classification accuracy. Every other model led to an accuracy decrease and the Residual UNet only had the second-best result because it affected the images the least.

It is noticeable that the results of the Zenodo show considerably lower classification accuracy than the other columns, which we will discuss later in Section 5.2.6. Taking into account the fact that the Residual UNet had a low impact on the images, we can consider the Uformer-T the best-evaluated model trained from scratch. The Uformer model uses a combination of attention layers and convolutional layers, obtaining the advantages of both methods, to which we attribute the better performance of the model.

The results also show that pretraining the model with natural images for denoising led to a better model, as the pretrained Uformer-T was the best overall model. We also qualitatively evaluate the impact of using pretrained weights for model initialization. Fig. 5.4 shows an image from the Zenodo dataset that has noise, which appears to be affected by some form of low-pass filter. Three restoration versions are presented: using a randomly initialized model without training, using a model with transfer learning weight initialization but without task-training, and finally a fully trained model. The only model that effectively removed the noise from the images was the fully trained model. Although the pretrained model was trained to remove noise from natural images, it does not appear to transfer as well for the thicker noise that may affect MRI images. However, the model with randomly initialized weights appears to create even more artifacts in the image. The pretraining is thus shown to stabilize the start of the training, but the model still has to learn to remove the particular noises and other artifacts that affect MRI images.

5.2.5 Restoration Training Dataset

In this experiment, we evaluated the impact of the image restoration dataset. The test is similar to the previous experiment, MSSIM is measured in the "Degraded" test set and accuracy is measured in the Figshare, "Degraded", Kaggle and Zenodo test sets, using an EfficientNet-b0 model trained in the Figshare training set for classes "Meningioma", "Glioma" and "Pituitary".

The training sets for the restoration model were the Figshare dataset and the Siar dataset, but for the Siar dataset we only use images from the "No Tumor" class, as they have the best image quality. These two training sets were selected because they were the two with the best image quality, which provides the undegraded target image for model training. In this

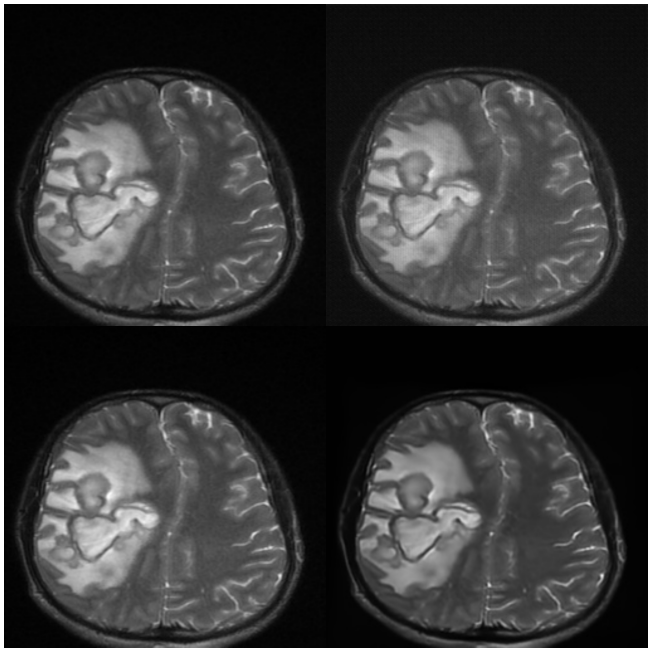


Figure 5.4: Qualitative influence of the pre-training. Top left: image from the Zenodo dataset(QADRI *et al.*, 2022). Top right: image restored by model without training. Bottom left: image restored by model with SIDD(ABDELHAMED *et al.*, 2018) pre-training only. Bottom right: image restored by fully trained model.

Table 5.11: Restoration performance depending on training dataset and number of training epochs. Uformer-T restoration model used.

Training set	Restoration train epochs	MSSIM	Test set accuracy (%)			
			Figshare	Degraded	Kaggle	Zenodo
No Restoration	-	0.596	95.38	78.39	85.21	42.73
Figshare (CHENG *et al.*, 2016)	100 epochs	0.895	95.42	92.94	87.86	45.48
	500 epochs	0.908	95.47	93.63	88.30	45.93
Siar (SIAR; TESHNEHLAB, 2022)	100 epochs	0.860	95.13	92.28	86.31	47.19

experiment, we also evaluated increasing the number of training epochs from 100 to 500 for a model trained on the Figshare training set. The results are shown in Table 5.11.

The models trained on the Figshare training set had the best results for MSSIM and accuracy on the Figshare, "Degraded" and Kaggle test sets. Note that all of these test sets were produced at least in part using images from the Figshare dataset. However, we note that no image from the Figshare and "Degraded" test set is part of the Figshare training set, and not even different images from the same patients are split between the training and test set from the Figshare dataset. However, those test sets should have images with conditions similar to those in the training set.

The model trained on the Siar training set had the best results when tested on the Zenodo

dataset, even better than the model trained for a larger number of epochs. In this case, both the Zenodo test set and the Siar training have images almost exclusively from the axial view, while the other training set and test sets have a mix of all three views. Given that observation, we can argue that a restoration model can specialize in restoring a single type of MRI view and perform better for that specific view. The two Uformer-T models trained in the FigShare dataset have shown the best overall results in Table 5.11 and are probably better suited to generalizing to other datasets without specified MRI views. These two models will be used in the following experiments.

5.2.6 Classification Training Dataset

Similarly to the previous experiment, in this experiment we evaluate different training sets, except this time it is for training the classification model. The Uformer-T model is used for restoration in all experiments. We divided this experiment into three parts. In the first part, the classification models are trained in training sets with its specific classes and then are tested in the test set from the same dataset. For the other two experiments, the datasets are adapted to have specific classes, and models are trained on specific adapted training sets and tested on all test sets that have those classes. For the second part of this experiment, the classes are "Meningioma", "Glioma" and "Pituitary". For the third part of the experiment, the classes are "Tumor" and "NoTumor".

5.2.6.1 Corresponding Test Set

In this experiment, models were trained and tested in the datasets Figshare, "Degraded", Zenodo, Kaggle, Siar, and Br25H 2020. The models were trained in each training set and tested in the corresponding test set of each dataset. For the "Degraded" test set, models were trained in the Figshare training set with artificially added image degradations. The classification model was EfficientNet-b0 and the restoration model was Uformer-T. Table 5.12 shows the result of this experiment.

We observe that, with the exception of the Kaggle and Siar datasets, models see an improvement in accuracy when image restoration is applied before image classification. Considering

Table 5.12: Classification accuracy when the model is trained and tested on the train and test set of the same dataset.

Model	Restoration	Dataset accuracy (%)					
		Figshare	Zenodo	Degraded	Kaggle	Siar	Br35H
EfficientNet-b0 + Uformer-T	No Restoration	95.38	94.26	93.03	99.78	99.71	99.17
	Restored	95.42	94.86	93.58	99.68	98.93	99.33

that the "Degraded", Zenodo, and Br35H datasets include degraded images, this result reinforces the observation made in Section 5.2.2 that even if a model was trained on degraded images it can still be improved by image restoration. In Section 5.2.6.3 we will discuss the causes of the drop in accuracy for the Kaggle and Siar datasets after image restoration.

Table 5.12 shows that the accuracy results above 90% were obtained for all datasets. Of particular note are the results obtained for the Zenodo dataset, which shows considerable improvement over the results from the previous section. In previous sections, the model used to classify images from all datasets was trained on Figshare, which shows that the data from the Figshare dataset do not generalize well to images from the Zenodo dataset. However, the data from the Zenodo dataset seem to be a good representation to achieve high accuracy in tumor classification. In the following section, we will deepen the analysis of the generalization capabilities of the training sets.

When comparing the accuracy obtained for different datasets, we first observe that for the Figshare, Zenodo, and "Degraded" datasets, each with 3 possible classes, the accuracy is proportional to how much the images are degraded. This can indicate both that image distortions may make tumor type recognition more difficult and that "noise learning" might have occurred during training, leading to overfitting. For the Siar and Br35H datasets the higher accuracy is expected, given that a binary classification between the presence or not of tumor should be easier than discerning different types of tumor. The high accuracy in the Kaggle dataset is another case. Repetition of the subject in the training and test set of the Kaggle dataset is the first aspect that leads to misleadingly higher accuracy. The other reason is intense noise learning that even correlates expected outputs in the test set, as we will discuss in the next experiments.

Table 5.13: Classification accuracy for meningioma, glioma and pituitary tumor types datasets, depending on training dataset and whether or not restoration was used for testing.

Training set	Restoration	Test set accuracy (%)			
		Figshare	Degraded	Kaggle	Zenodo
Figshare	No Restoration	95.38	78.39	85.21	42.73
	Restored	95.42	92.94	87.86	45.48
degraded Figshare	No Restoration	94.02	93.03	76.93	32.53
	Restored	94.00	93.58	77.04	33.10
Kaggle	No Restoration	91.41	40.48	99.78	45.36
	Restored	90.19	55.86	99.68	45.25

5.2.6.2 Meningioma, Glioma and Pituitrary Tumor Classes

In this experiment, the MRI classes were "Meningioma", "Glioma" and "Pituitary". The training sets were the Figshare, the Figshare with artificially degraded images, and the Kaggle training set without the "NoTumor" class. The test sets were Figshare, "Degraded", Kaggle test set without the "NoTumor" class and Zenodo test set without the "Adenoma" class. Table 5.13 shows the result of this experiment.

Table 5.13 shows that the best accuracy is obtained when a classification model is trained in the corresponding training set for a given test set. The results still show that the models are capable of generalizing to other test sets with equivalent classes, with the notable exception of the Zenodo dataset. The Kaggle dataset training also did not generalize well to the "Degraded" dataset. We also note that the Kaggle dataset training had the worse reaction in image restoration, losing accuracy in all test sets except the "Degraded" test set. As a counterpart, the model trained on the Figshare training set was improved by image restoration in all test sets, while it also seems to be the model that best generalizes to the other test sets, especially after image restoration. To evaluate why each classification model had such results and why image restoration had the observed effect, we obtained the confusion matrix for each test, with and without image restoration. Fig. 5.5 shows the confusion matrices of the model trained on the Figshare training set and tested on the "Degraded" and Zenodo datasets. Fig. 5.6 shows the confusion matrices of the model trained on the Kaggle training set and tested on the "Degraded" and Zenodo test sets.

Based on the confusion matrices, we can say that for the model trained on the Figshare train set the error is more evenly distributed among the target classes. For the Kaggle-trained model,

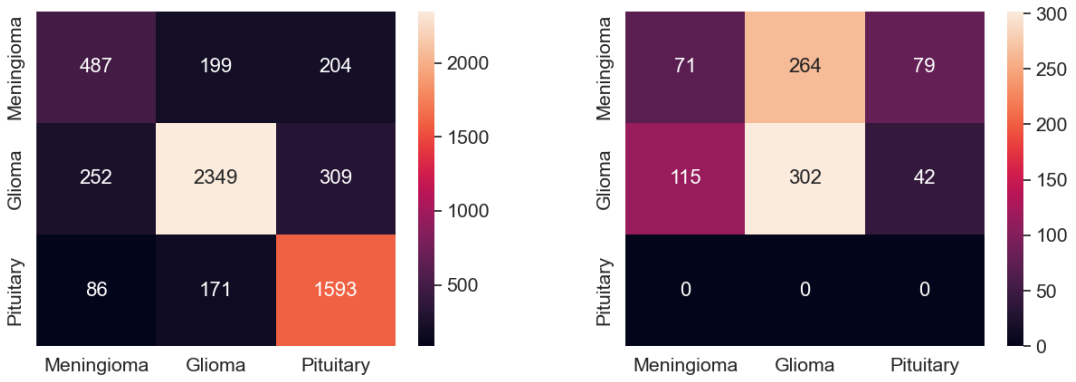


Figure 5.5: Confusion matrix when training the model on the Figshare train set and tested on the "Degraded" and Zenodo test sets. Lines are true labels and columns are predicted labels.

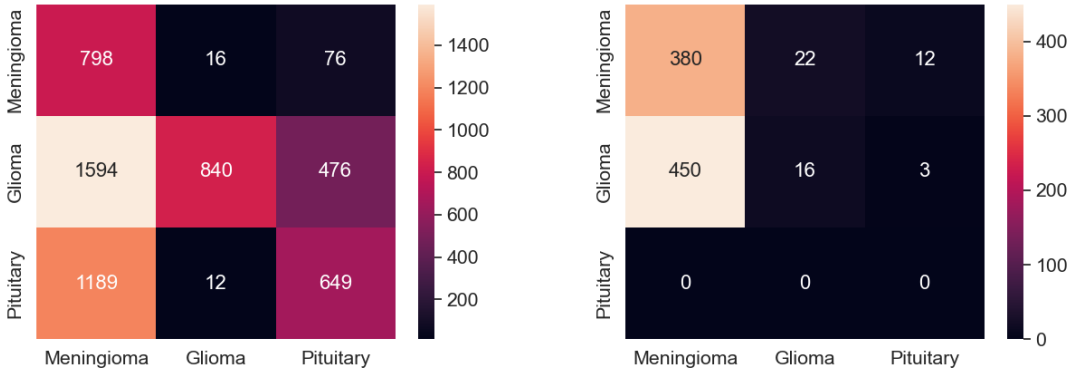


Figure 5.6: Confusion matrix when training the model on the Kaggle dataset and tested on the "Degraded" and Zenodo test sets. Lines are true labels and columns are predicted labels.

there is a noticeable lower tendency to classify images as glioma tumors and a higher tendency to classify images as meningioma. In the Kaggle dataset most glioma images come from the Figshare dataset, being good quality images, while the meningioma and pituitary classes are a mix of undegraded and degraded images. Considering the fact that the "Degraded" and Zenodo test sets include large amounts of poor-quality images, we may infer that the model trained in the Kaggle dataset learned to associate good-quality images with the glioma class and degraded images with the pituitary class and even more to the meningioma class.

Section 5.2.2 showed that training a classification model on degraded images can make the model more robust against these degradations. But that is only true if the degradations are evenly distributed between the classes and if the training degradations match the testing degradations. The Kaggle training set has more images than the Figshare, consequently being trained for more iterations, and consists of a mix of good quality and poor quality, yet it had

considerably lower accuracy in the Figshare and "Degraded" test sets than the models trained on the Figshare training set. This poorer performance of the Kaggle model is a consequence of the Kaggle training set having degraded images unevenly distributed among the target classes, leading the model to associate image degradations to specific classes, resulting in errors when that association is not the case.

When comparing Figshare training with and without degradations, the model trained with degradations and tested with restored images is slightly better on "Degraded", but performs considerably worse on the Figshare, Kaggle and Zenodo datasets. As observed in Section 5.2.2, training with image degradation leads to a lower maximum accuracy on lower intensity image degradations, and these three test sets have several undegraded images and images with lower intensity degradations, leading to poorer performance.

Although the model trained on the Figshare dataset has less of a degradation bias toward any particular class, it is noticeable that the trained models have particularly low accuracy on the Zenodo dataset. This can be attributed to the fact that the Figshare dataset and to a large extent the Kaggle dataset are made of T1-CE-weighted MR images, while the Zenodo dataset contains mostly T2-weighted images. In Fig. 5.7 we compare images samples from the Kaggle dataset on top and the Zenodo dataset on bottom, of the glioma class on the left and the meningioma class on the right.

In Fig. 5.7 we notice that in the examples from the Zenodo dataset the external regions of the brain are much brighter than the rest of the scan, including the cranium region, matching the description of T2-weighted magnetic resonance imaging. In the images from the Kaggle dataset the external regions are darker than the internal white matter, matching the T1-weighted MRI description. In the meningioma sample from the Zenodo dataset, the tumor makes the outer region of the brain darker, though still slightly lighter than the white matter region. In the Kaggle meningioma sample, the reverse is true, the tumor is lighter than the brain regions, particularly the external regions. For the glioma tumor, the pattern is inverted; in the Zenodo image the tumor is lighter than the brain tissue, while on the Kaggle image the tumor is darker than the brain tissue. This pattern is not true for all images in both datasets, however it is representative of a large portion of the images, and is one explanation why models trained only on a dataset of T1-MRI images aren't as effective on datasets with T2-MRI images.

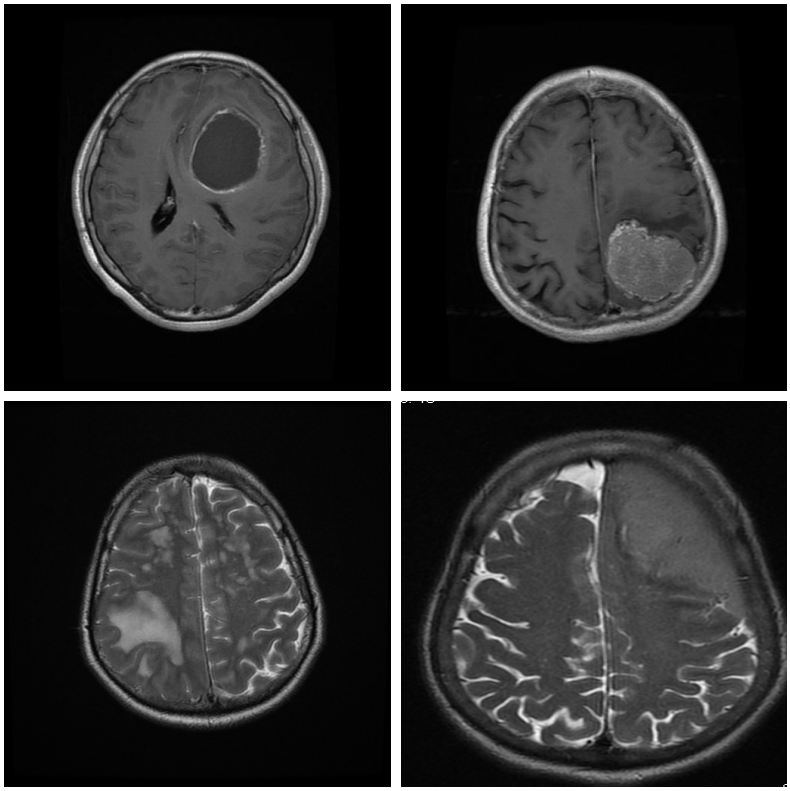


Figure 5.7: Glioma and meningioma examples from the Kaggle and Zenodo datasets. Top: Kaggle dataset. Bottom: Zenodo dataset. Left: glioma tumor. Right: meningioma tumor.

The results of this experiment have shown that training a model on a dataset with image degradations can lead a model to learn the "wrong" features. The very high accuracy of models trained and tested on the Kaggle dataset is, at least in part, a result of the models learning a correlation between image quality and specific classes, making those models unreliable for real-world applications. Training a classification model on undegraded images and using image restoration to handle degraded images is a more robust and reliable approach. The results also show that there is poor generalization between T1-weighted images and T2-weighted images, and brain tumor classification models should only be used to classify images of the same modalities for which they were trained.

5.2.6.3 Tumor x No Tumor

Similar to the previous experiment, we defined the classes "Tumor" and "NoTumor", and trained models to identify these two classes. The "NoTumor" class means that there are no tumors on the MRI scan, or that the brain is healthy. "Tumor" indicates that there is a tumor in the image, without distinction of the type of tumor. Three datasets were used for

Table 5.14: Classification accuracy for "Tumor" and "NoTumor" classes depending on training set and whether restoration was applied. Models were tested in three different test sets with equivalent classes.

Training set	Restoration	Test set accuracy (%)		
		Siar	Kaggle	Br35H
Siar	No Restoration	99.71	60.34	59.67
	Restored	98.93	71.31	77.00
Kaggle	No Restoration	49.36	100	74.67
	Restored	49.93	99.77	73.83
Br35H	No Restoration	75.64	92.91	99.17
	Restored	77.14	92.91	99.33

this experiment: Siar, Kaggle, and Br35H. The Siar and Br35H are already split in those two classes, but the Kaggle dataset had to be adapted. The Kaggle dataset includes the "NoTumor" class, but has 3 different types of tumor classes, and we merged all three classes into a single "Tumor" class, without distinction of the type of tumor. Table 5.14 shows the result of this experiment.

As can be seen in Table 5.14, the models perform better when tested on the corresponding test set as the training set. The Siar and Kaggle models observe an accuracy reduction on their corresponding test sets when images are restored, and the worse restored performances are for the Siar and Kaggle datasets when switching the test sets of those two datasets. The model trained on the Br35H is the more robust and generalizes better to other datasets, having the highest average, median, and minimum accuracy on the three test sets. In the majority of the cases and on average, image restoration improves the models accuracy. To evaluate why each training set leads to such results, we obtained the 9 confusion matrices for the tests without image restoration. Fig. 5.8 shows the 9 confusion matrices obtained.

In Fig. 5.8, the confusion matrices in the principal diagonal of the image are models trained and tested on the same dataset (trained on the training set and tested on the test set); in those three confusion matrices results show that most images were correctly classified. For all other confusion matrices, the number of errors is greater, which shows a limited generalization capability of these datasets.

From the confusion matrices we can see that the model trained on the Siar train set when tested on the other datasets tends to classify images as Tumor images, with a large number of both "Tumor" and "NoTumor" being predicted as "Tumor". When tested in the Kaggle test

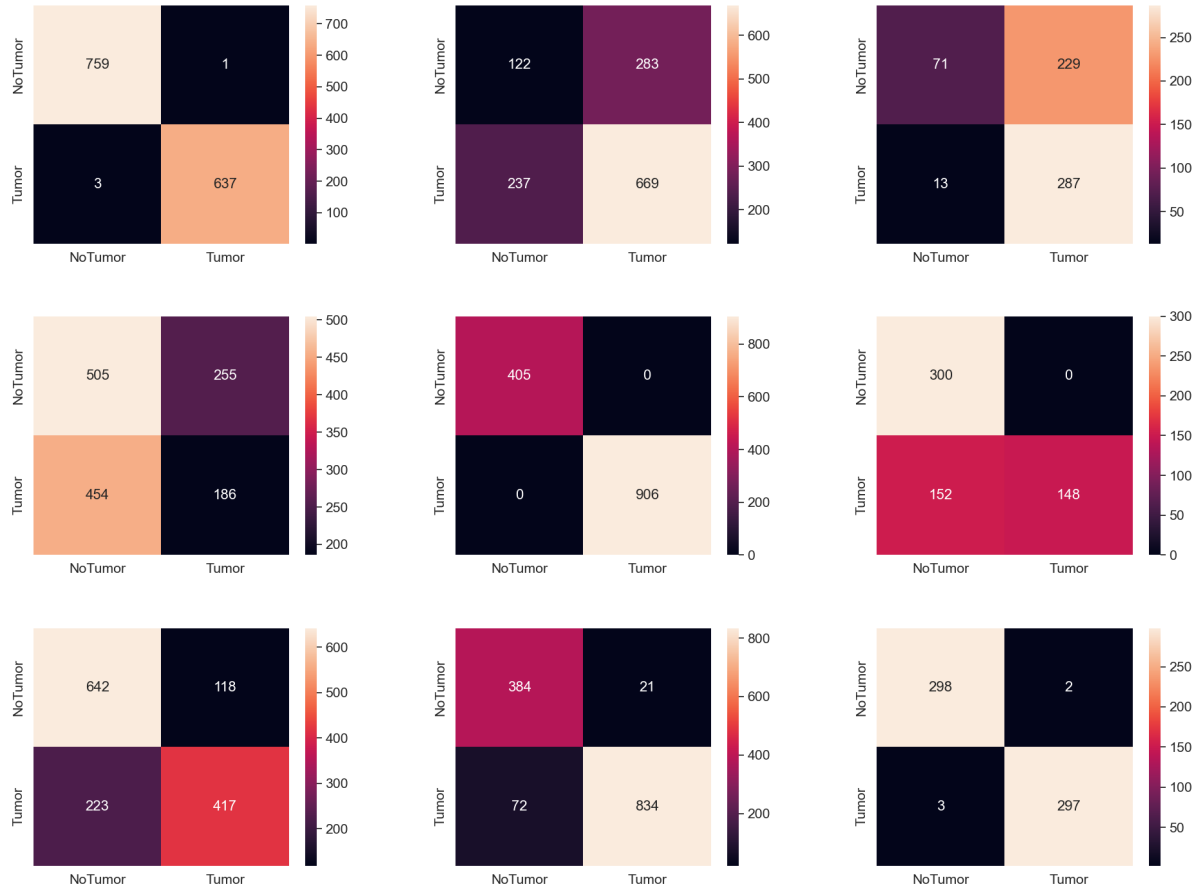


Figure 5.8: Confusion matrices for table 5.14. Rows are the true labels and columns are the predicted labels. From left to right, the columns represent the tests sets Siar, Kaggle, and Br35H. From top to bottom, the rows represent the training sets Siar, Kaggle, and Br35H.

set, the Siar model also has a large number of "Tumor" images classified as "NoTumor", even greater than "NoTumor" are classified as "NoTumor". For the model trained in the Kaggle train set, the exact opposite problem is observed: on different test sets, the model tends to classify images as being in the "NoTumor" class, but on the Siar test set it also makes the mistake of predicting the "NoTumor" classes as being in the "Tumor" class. As for the model trained on the Br35H train set, while there is an increase in errors, there is no significant tendency to one specific class and the images are still mostly predicted in the correct class in all cases, directly related to the better performance of this model in Table 5.14.

To identify the cause of this observed effect, we investigated the image quality distribution of the images in those three datasets. In the Siar dataset images of the "NoTumor" class are of mostly good quality, while the images of "Tumor" are mostly degraded. In the Kaggle dataset the reverse is true, "NoTumor" images are mostly degraded and most good quality images are

in the "Tumor" class. Finally, the Br35H dataset has mostly degraded images on both classes. Based on this information and the confusion matrices obtained, we argue that the models learned to correlate image quality with specific classes. The Kaggle model tends to classify poor quality images as "NoTumor" images, and the Siar model tends to classify poor quality images as "Tumor". The Br35H model has image degradation evenly distributed into the two classes and does not make such correlation, consequently being the most reliable model.

In Table 5.13 the best model was the one trained only on undegraded images, while in Table 5.14 the best model was the one trained on degraded images, since there were no fully undegraded datasets in this case. The results show that to train a reliable tumor classification model, the image degradations in the dataset, if any, should be evenly distributed in the dataset classes. Based on the results, we recognized some aspects that influence the training in 2D MRI classification, in addition to tumor class. These aspects are the type of magnetic resonance scan (weighted T1, T2-weighted, etc.), the slice plane (axial, sagittal, and coronal) and the types of image degradation.

5.2.7 Classification Model

The experiment in Section 5.2.4 evaluated the influence of using different restoration models in the proposed framework. In this section, we evaluate the influence of changing the classification model. All models were trained in the Figshare training set with the classes "Glioma", "Meningioma" and "Pituitary" and the accuracy of these classes was tested in the Figshare, "Degraded", Kaggle and Zenodo datasets, adapted to only include the three aforementioned classes.

The evaluated model architectures were ShuffleNetv2(MA *et al.*, 2018), ViT (DOSOVITSKIY *et al.*, 2020), RegNet (RADOSAVOVIC *et al.*, 2020), DenseNet (HUANG *et al.*, 2017), ResNet(HE *et al.*, 2016), ConvNeXt (LIU *et al.*, 2022) and EfficientNet (TAN; LE, 2019). For the ResNet, ConvNeXt, and EfficientNet architectures, more than one model size was tested. For the ViT model, the resolution was changed from 256x256 pixels to 224x224 pixels, since that is the resolution expected by that model, while other models allow for different resolutions. All models were initialized with the pre-trained weights made available by the Pytorch

Table 5.15: Classification accuracy of different models trained on dataset 1 and tested on various test sets. Uformer-T model used for image restoration. Results are mean and standard deviation of 3 runs. Best result in each column is highlighted in bold.

Model	Restoration	Test set accuracy (%)			
		Figshare	Degraded	Kaggle	Zenodo
ShuffleNetv2-x1.5 (MA *et al.*, 2018)	No Restoration	92.87 ± 0.50	59.47 ± 0.45	65.19 ± 1.87	40.78 ± 2.85
	Restored	92.94 ± 0.56	89.54 ± 0.97	63.87 ± 1.88	43.26 ± 2.01
ViT-b-32 (DOSOVITSKIY *et al.*, 2020)	No Restoration	89.14 ± 0.26	67.90 ± 1.43	76.11 ± 1.63	41.69 ± 0.57
	Restored	89.26 ± 0.24	88.57 ± 0.18	72.74 ± 0.50	42.39 ± 1.14
RegNet-Y-3.2GF (RADOSAVOVIC *et al.*, 2020)	No Restoration	92.81 ± 1.11	66.16 ± 5.54	77.89 ± 3.77	25.79 ± 5.51
	Restored	92.60 ± 1.18	84.36 ± 5.14	77.37 ± 3.75	29.72 ± 2.44
DenseNet-201 (HUANG *et al.*, 2017)	No Restoration	93.03 ± 0.45	72.30 ± 3.79	84.62 ± 2.08	41.20 ± 2.65
	Restored	93.04 ± 0.50	91.13 ± 0.92	84.58 ± 1.49	44.79 ± 1.59
ResNet-18 (HE *et al.*, 2016)	No Restoration	91.69 ± 0.80	72.62 ± 2.20	79.17 ± 2.01	37.92 ± 5.39
	Restored	91.57 ± 0.78	90.28 ± 0.68	77.08 ± 3.82	40.85 ± 5.11
ResNet-50 (HE *et al.*, 2016)	No Restoration	93.41 ± 0.13	67.94 ± 1.76	79.73 ±2.46	38.79 ± 6.50
	Restored	93.35 ± 0.21	90.15 ± 1.59	80.13 ± 2.60	39.52 ± 7.02
ResNet-101 (HE *et al.*, 2016)	No Restoration	92.67 ± 1.36	71.80 ± 5.66	82.71 ± 2.24	42.69 ± 1.40
	Restored	92.65 ± 1.36	90.75 ± 0.61	83.33 ± 2.07	43.98 ± 1.31
ConvNeXt-T (LIU *et al.*, 2022)	No Restoration	94.42 ± 1.44	80.04 ± 4.43	78.85 ± 11.58	39.71 ± 3.43
	Restored	94.45 ± 1.48	92.55 ± 0.91	78.11 ± 12.70	43.87 ± 2.92
ConvNeXt-S (LIU *et al.*, 2022)	No Restoration	94.66 ± 0.61	82.26 ± 1.17	84.21 ± 1.80	37.38 ± 6.43
	Restored	**94.71 ± 0.68**	**92.77 ± 0.60**	84.40 ± 1.69	40.74 ± 5.10
EfficientNet-b0 (TAN; LE, 2019)	No Restoration	92.52 ± 0.24	61.83 ± 0.99	86.98 ± 2.98	42.46 ± 3.76
	Restored	92.30 ± 0.09	87.52 ± 0.02	88.23 ± 2.52	43.80 ± 1.74
EfficientNet-b2 (TAN; LE, 2019)	No Restoration	93.79 ± 0.39	69.49 ± 0.38	88.78 ± 1.83	45.17 ± 2.04
	Restored	93.54 ± 0.43	87.81± 1.16	**90.36 ± 0.99**	**46.24 ± 1.49**
EfficientNet-b4 (TAN; LE, 2019)	No Restoration	92.82 ± 0.31	67.42 ± 3.29	85.43 ± 2.01	44.90 ± 1.28
	Restored	92.43 ± 0.37	83.25 ± 0.88	86.72 ± 1.65	45.17 ± 1.27

framework. The results are shown in Table 5.15.

The results in Table 5.15 show that there is not a single best model for all cases. Instead, the best model for the Figshare and "Degraded" datasets was the ConvNeXt-S model and the best model for the Kaggle and Zenodo datasets was the EffiientNet-b2 model. However, in all cases, the best result was obtained when image restoration was used. We may infer that the ConvNeXt architecture is better at learning the target dataset, while the EfficientNet architecture generalizes better.

Comparing different models of the same architecture, we observe that the best model isn't simply the largest model. The EfficientNet-b2 model performed better than the larger EfficientNet-b4 model on all test sets, although the EfficientNet-B4 model still performed better than the smallest EfficientNet-b0 model. Also, the smaller ConvNeXt-T was better than the larger ConvNeXt-S on the Zenodo dataset and the intermediate ResNet-50 was the best ResNet model on the Figshare dataset.

To evaluate the impact of image restoration, we first observed that for the "Degraded" and

Zenodo datasets, all models saw accuracy improvements when images were restored. For the Figshare test set, only 5 of the 12 models saw improvements in image restoration. Since the Figshare dataset is compromised of undegraded images, an improvement is expected in about 50% of the cases. We also observe that the largest average accuracy change when adding image restoration to the Figshare dataset is only 0.39%, so this dataset is not significantly affected by image restoration, as desired.

The Kaggle dataset, with the exception of "NoTumor" class, also has mostly undegraded images and saw an increase in accuracy by restoration for only 6 out of the 12 models. However, more important is to observe which models were improved by image restoration. The six models that had a decrease in accuracy by image restoration were the five models with the lowest accuracy and the DenseNet-201 models. However, the DenseNet model only saw a 0.04% decrease in average accuracy, much lower than its accuracy standard deviation. The models that observed an improvement in accuracy by image restoration were the 7 most accurate models, except DenseNet, and the model that had the highest nominal improvement from image restoration was also the most accurate model, EfficientNet-b2, which had an increase in accuracy of 1.58% by image restoration. The results indicate that more accurate models become more sensitive to image quality, therefore ensuring good image quality leads to more accurate results.

We note here that all models were trained for only 100 epochs on a dataset of little more than 2000 training images, while the methods reported in the original papers include much longer training. This indicates that there is potential for improvements in the training of those classification models, though current results focus in evaluate the performance of the solution with image restoration.

5.3 QUALITATIVE ANALYSIS

To evaluate the performance of restoration qualitatively, we generated degraded images by applying the artifact generator function to images from the Figshare dataset, which we then restore using the trained Uformer-T model. We analyze the image degradation and perform a qualitative comparison with the restored images.

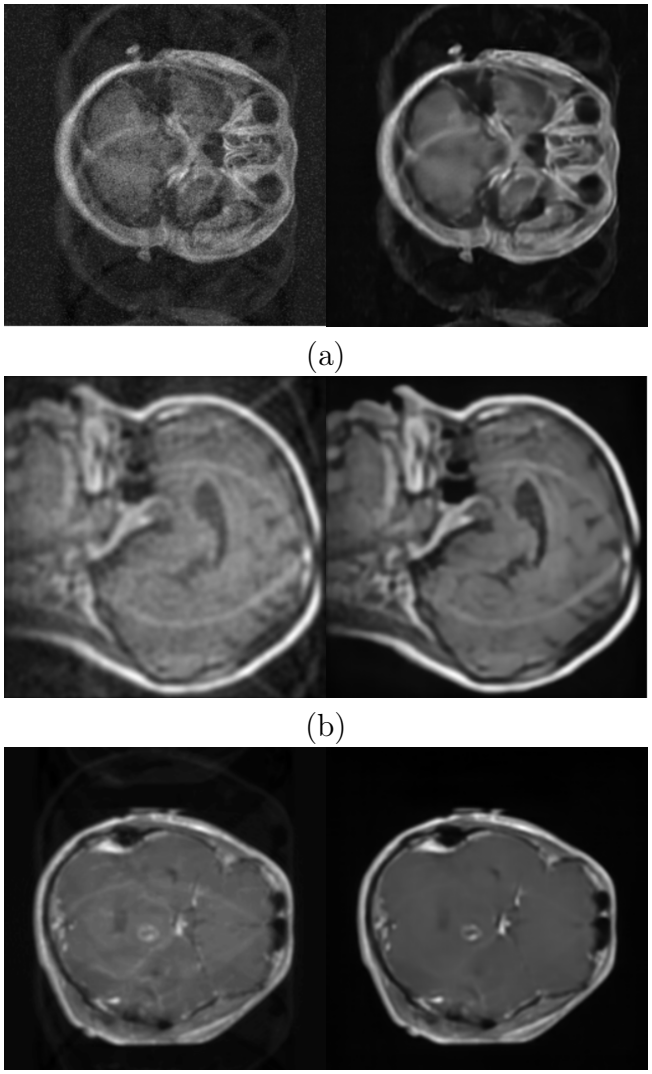
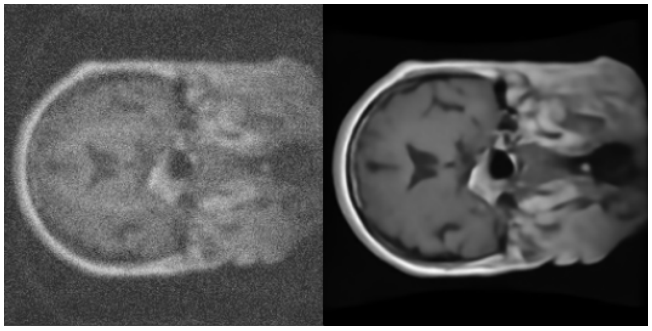
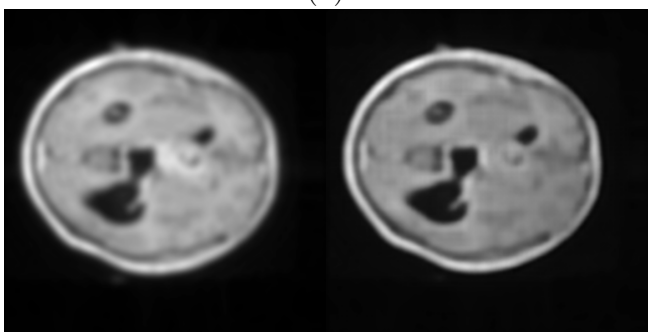


Figure 5.9: Examples of image restoration. Left: Artificially degraded image. Right: corresponding restored image.

In addition, we selected some images from the datasets that have shown some form of image degradation or lower image quality. These images have artifacts that were not generated by us, with unknown degradation values unseen during the training of the restoration model. The restoration results on these images show the generalizability of the restoration model. Similarly to what is done with artificially degraded images, we perform a qualitative comparison of the degraded and restored images. We split these results based on the dataset of origin of the images, so that we can evaluate the particularities of each dataset.



(a)



(b)

Figure 5.10: Examples of image restoration. Left: Artificially degraded image. Right: corresponding restored image.

5.3.1 Artificially Degraded Images

Using the artifact generator function and the restoration model, we generated pairs of degraded and restored images. Figs. 5.10 and 5.9 show some examples generated for qualitative analysis. In all pairs from Fig. 5.9 some degree of ghosting can be observed. In Fig.5.9(a), even after restoration, the ghosting artifact still remains, though at lower intensity and with parts entirely removed from the background. In Fig. 5.9(b), the ghosting artifact was completely removed from the background region, but the ghosting lines that go over the brain region still remain at lower intensity. In Fig. 5.9(c), the ghosting artifact is more perceptible in the brain region, similar to the middle restored image, but after restoration extra attention is required to perceive the remaining traces.

In all five restored images, some degree of image sharpening is perceived, which is best perceived in the brighter and darker regions, particularly in the darker region separating the brain from the cranium, but also in the dark tumor regions of Fig. 5.10(b), which shows the deblurring property of the model.

Noise can be seen in Figs. 5.10 (a) and (b) and in Figs. 5.9 (a). In the corresponding

restored images, the previously noisy images become smoother, without losing image sharpness, as previously mentioned. Additionally, this noise removal and image sharpening makes the structural details of the brain and cranium much clearer to the human eye and may also be associated with the increase in accuracy for the classification model seen in all experiments.

In Fig. 5.9(b) some ripple effect can be seen next to the cranium on the right and upper part of the image. This effect may be associated with the Gibbs ringing artifact. On the corresponding restored image this effect seems to have been smoothed over and is harder to perceive.

In Figs. 5.9(a) and 5.10(a), the low contrast may be identified mainly by the lighter background. In the corresponding restored images the background is completely black, and the cranium images may be perceived as brighter. The combination of contrast adjustment, sharpening, and noise removal leads to a better contrast between regions, so that segments are easier to identify.

5.3.2 Kaggle dataset

In Fig. 5.11 we show three pairs of input and restored images using images from the Kaggle dataset (NICKPARVAR, 2021) as input. All three images show some degree of poor contrast and brightness. For Figs. 5.11 (a) and (b), the restoration darkens the background, but also results in lower brightness of the signal itself. In Fig. 5.11 (c), the restoration leads to an increase in signal intensity, which makes the structural details of the image clearer.

In particular for Fig. 5.11 (a), noise removal is noticeable, with regions of the brain becoming smoother and some spots removed in the darker regions of the image, such as the gap separating the brain from the cranium in the upper part of the image. The restored image shows more distinguishable regions, which are highlighted in the tumor in the upper right part of the brain. The edges between the tumor and the brain are more clear, but there are also more distinguishable regions within the tumor, characterized by different signal intensities.

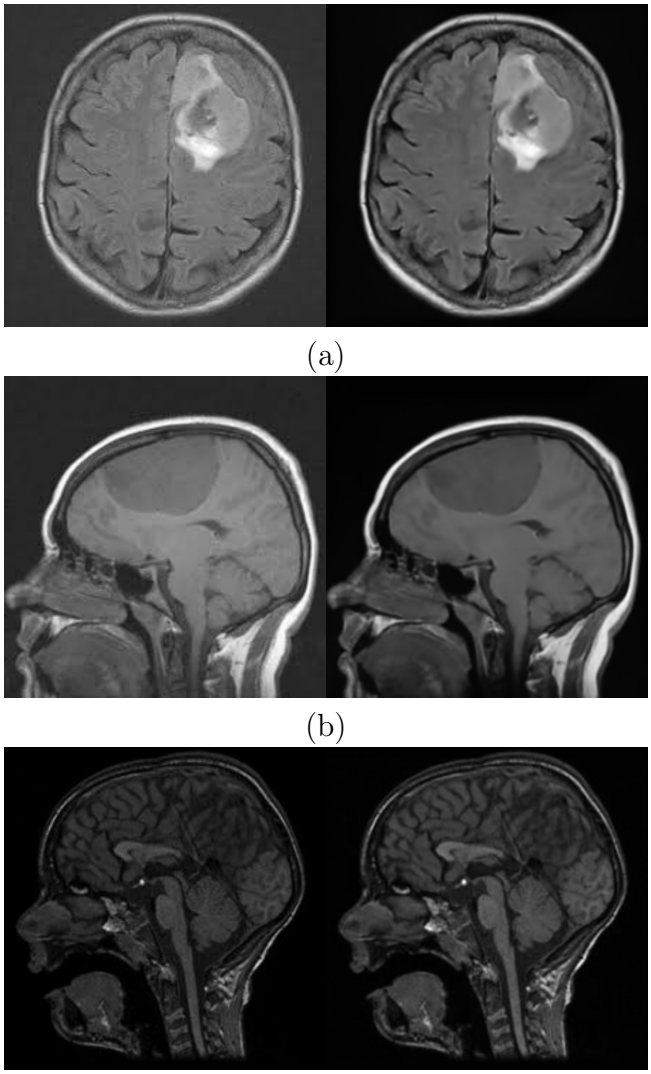
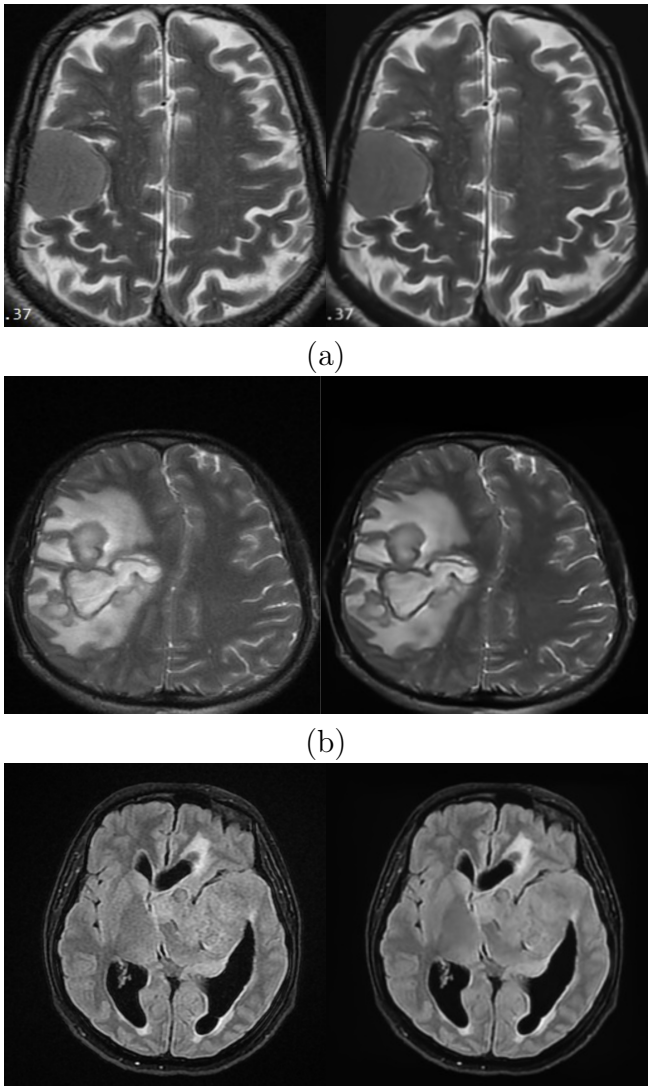


Figure 5.11: Examples of image restoration for images from the Kaggle dataset (NICKPARVAR, 2021). Left: Original image. Right: corresponding restored image.

5.3.3 Zenodo dataset

In Fig. 5.12 we show three pairs of input and restored images using images from the Zenodo dataset (QADRI *et al.*, 2022) as input. The brighter gray matter (external region of the brain) of the T2-weighted images can be clearly identified in the top image pair. Since this characteristic is quite different from what is seen in T1-weighted images, this high contrast between gray matter and white matter can lead to the lower accuracy of the classification models trained on the Figshare and Kaggle datasets when tested on the Zenodo dataset.

All three images show noise, which is more noticeable in the white matter (darker internal region of the brain) and tumor regions. In the restored images those regions show smoother,



(a)

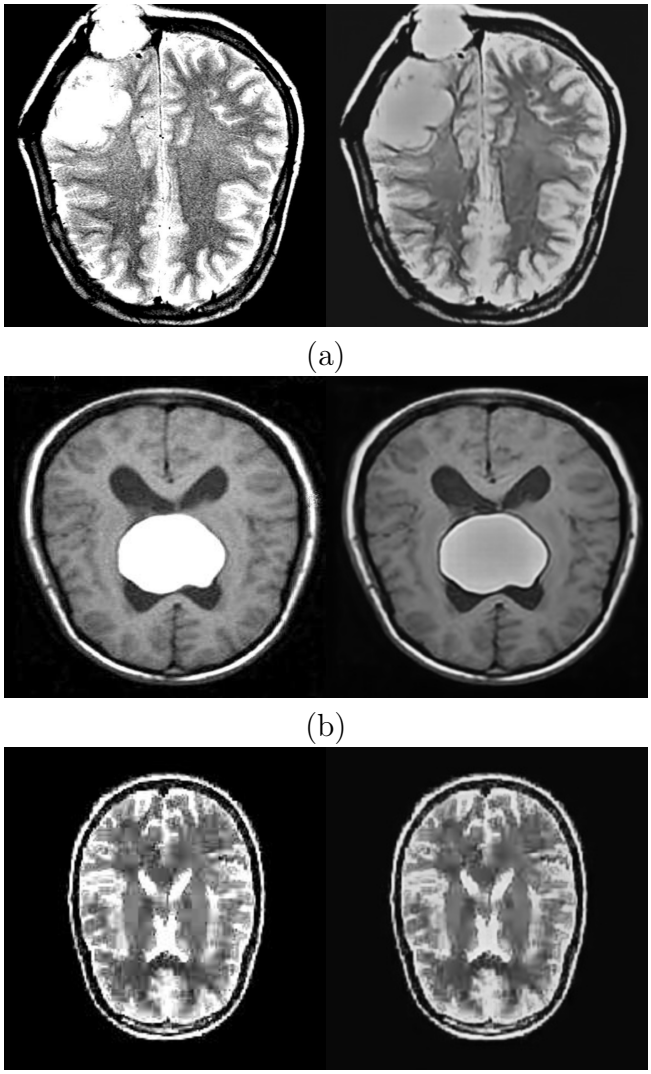
(b)

(c)

Figure 5.12: Examples of image restoration for images from the Zenodo dataset (QADRI *et al.*, 2022). Left: Original image. Right: corresponding restored image.

more constant, intensity values. Despite the smoother, less noisy regions, the restored images have sharper edges and more clearly defined regions, particularly for the middle image pair. A noticeable effect is the thicker black line in the tumor of the middle image pair.

In Fig. 5.12(a), Gibbs ringing is perceived close to the central column of the image. That is, close to the division between the left and right hemispheres of the brain. In this division, there is a dark gap with bright gray matter on each side, creating the sharp edge that generates the Gibbs ringing artifact. The smoothing effect that reduced the noise in the restored image also affected the ringing artifact. In the original image, several ripples can be identified to each side, while on the restored image only one oscillation is very clear.



(a)

(b)

(c)

Figure 5.13: Examples of image restoration for images from the Br35H 2020 dataset (HAMADA, 2020). Left: Original image. Right: corresponding restored image.

5.3.4 Br35H 2020 dataset

In Fig. 5.13 we show three pairs of input and restored images using images from the Br35H 2020 dataset (HAMADA, 2020) as input. Similarly to the examples of the other dataset, in Fig. 5.13 (a) and (b) noise is shown to have been smoothed out. In both these image pairs, the tumor shows with saturated intensity, and in the first image pair the tumor brightness intensity matches that of the also saturated gray matter, which makes it more difficult to identify the tumor at the top left portion of the head, except for structural differences, such as the more rounded aspect of the tumor. All three restored images show lower brightness, showing average brightness more similar to what is seen in images from the Figshare dataset,

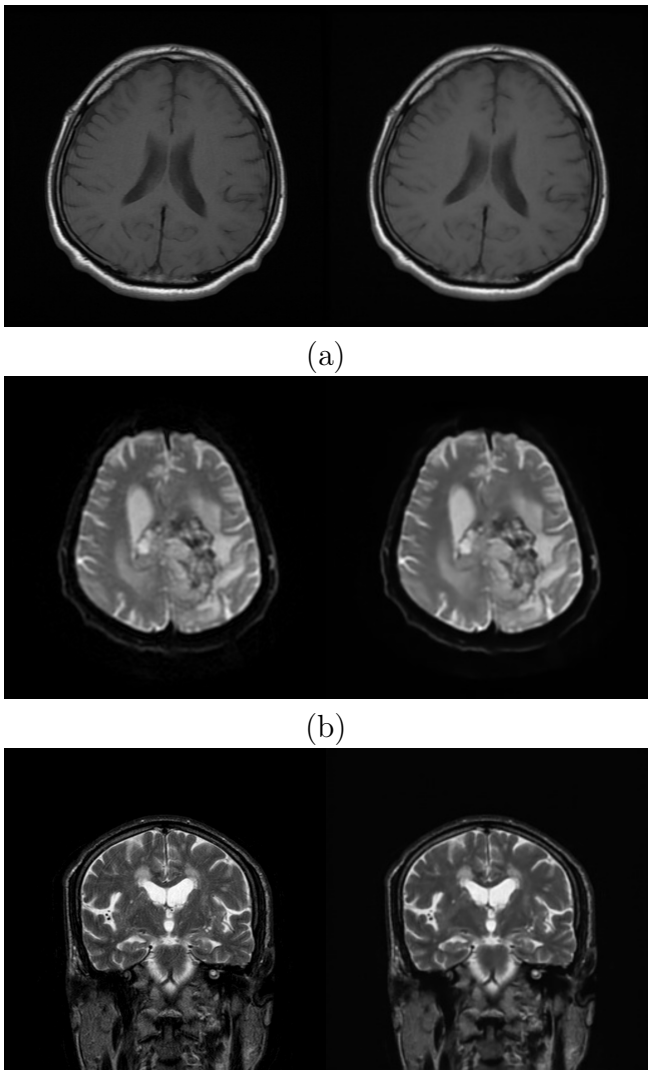
which was used to train the restoration model, which could considered a negative aspect for brain tumor recognition. In Fig. 5.13(b) an interesting effect can be observed; there is a thicker dark edge between the tumor and the brain, which highlights the tumor.

Fig. 5.13(c) is a healthy brain image, but it is also the image with the lowest quality of the three. In that image strong JPEG compression artifacts can be seen in the form of square regions with sharp edges (although on the PDF these might be less clear). In the corresponding restored image, while it is still a very poor quality image, the square regions of the JPEG compression are less distinguishable and appear to have been smoothed out. An interesting aspect of the model is highlighted with this and previous results, while the model smooths out noise, ringing, and JPEG compression artifacts it still keeps and even highlights the edges of tumor regions.

5.3.5 Siar dataset

In Fig. 5.14 we show three pairs of input and restored images using images from the Siar dataset (SIAR; TESHNEHLAB, 2022) as input. The images show similar results to those from the other datasets. In Fig. 5.14(a), Gibbs ringing artifact is seen on both sides of the brain, with fewer ripples being noticeable on the restored image. Figs. 5.14(b) and (c) both show noise removal results, although Fig. 5.14(c) appears to have lost some high-frequency details which are hard to distinguish from noise. In the brain of the bottom degraded image, some lines may be perceived and can be associated with ghosting and/or Gibbs ringing and have mostly been removed in the restored image.

Overall, results for all datasets show that the model trained on T1-CE-weighted brain MRI images with artificially added image artifacts generalizes well to unseen data with image artifacts. The restoration result also works well for T2-weighted images, although this change in MRI modality negatively affects image classification if the classification model was not trained for that particular MRI modality, as seen in the quantitative results. Although the model cannot fully recover the quality of excessively degraded images, such as the examples in Fig. 5.13, it can still soften the artifacts of the image with considerably more intense degradations than the ones used for training.



(a)

(b)

(c)

Figure 5.14: Examples of image restoration for images from the Siar dataset (SIAR; TESH-NEHLAB, 2022). Left: Original image. Right: corresponding restored image.

CHAPTER 6

CONCLUSIONS

We proposed a two-stage solution to aid in the diagnosis MRI scans with potential image degradations, including Rician noise, Gaussian blur, Gibbs ringing, poor contrast, Nyquist ghosting and JPEG compression. The solution uses a Uformer-T model to first restore images, ensuring a minimal image quality for the second stage, in which an EfficientNet model predicts the diagnosis for the restored image. We performed extensive testing with artificially degraded images and with 5 different datasets containing real cases of image degradations. The results show that the proposed approach improved the accuracy of diagnosis prediction, improved the robustness to image degradations, and made computer-aided diagnosis more reliable. Of particular note is the effectiveness of the approach for images with an arbitrary combination of artifact types.

We evaluated real cases of degraded images, highlighting that images are often affected by multiple types of artifacts. For those cases, we observed that different types of image distortion affect each other, particularly noise artifact. Isolated noise in MRI follows the Rician distribution, but when the image is also affected by blurring of the Gibbs artifact, the noise appears thicker and the histogram of the noise has a smaller range of values. When images are affected by contrast artifacts, the noise is constrained to regions with intermediate pixel intensities, such as the brain region, and does not appear on the darker background or lighter regions of the cranium. When images are also affected by the JPEG compression artifact, the noise may diverge from the Rician distribution, and this difference should be considered in the denoising process. To make computer-aided diagnosis reliable for real-world applications, the methods have to account not only for all different artifact types but also the combination of those artifacts, which our solution does. We found in our dataset evaluations that the most common type of artifact in the datasets was “thick” noise that can be simulated with Rician noise plus some form of low-pass filtering.

We have shown empirically that to make the restoration model more effective, the training images have to have image degradations similar to the testing conditions, as expected; but this also depends on the type of artifact. Ghosting and contrast artifacts have a lower correlation between image quality and computer-aided diagnosis accuracy, and do not have to be included too much during training in a multiple-artifact scenario, even if they are frequent during testing.

As an alternative to image restoration, training classification models on degraded images, artificial or not, can also improve model robustness to image degradations. Both methods can also be combined, as even models trained on degraded images see accuracy improvements when the images are restored before classification. Training on degraded images does, however, lower the maximum accuracy of the model as well as its generalization capabilities. The image restoration approach not only improves accuracy on the target dataset, but also significantly improves generalization for other datasets.

We compared different models for image restoration, comparing the Uformer model with CNN-based models, which showed that the Uformer model obtains better image quality and improvements in computer-aided diagnostic accuracy. This shows that models based on the combination of transformer and convolutional layers have great potential in medical image restoration. We also compared several image classification models, which corroborated that the EfficientNet architecture is the most indicated for medical image classification, but the ConvNext architecture also showed competitive results.

We observed that the image restoration method works well on MRI of both T1 and T2 modalities, as well as all three planes, axial, coronal, and sagittal, even when some of those particular cases are not included during training, though with some loss of effectiveness. This, however, is not the case for the classification model, which requires training on each particular variation on brain MRI to be effective, specially between the T1 and T2 MRI types there is a lack of generalization capability.

In all our experiments, we show that during testing, ensuring better image quality leads to more accurate computer-aided diagnosis. However, during the training of automatic image diagnosis methods, image quality presents the greatest risks. When training image classification models on degraded images, the model might learn to correlate image quality with particular diagnostic classes, leading the models to have very high accuracy on test sets with similar condi-

tions, giving a false sense of security for those models. When these models are tested on unseen data with different quality conditions, those models fail, with significant drops in accuracy, making those models unreliable for real-world applications. While datasets with image degradations are good for testing the robustness of the methods, computer-aided diagnostic models should ideally be trained on good quality images to be more reliable, with image restoration as an auxiliary tool to keep robustness on poor quality images.

The method was tested and compared with other models for image restoration and image classification. The results validate the choice of using Uformer-T models for image restoration. When evaluating different image classification models, we observed that the more accurate the model is, the more relevant it is to include an image restoration step, as these more accurate models tend to be more sensitive to small details to classify more difficult cases and image restoration reduces these small variations.

All experiments were performed using image classification models. Future research efforts may involve implementing this approach for tasks such as tumor detection and segmentation. In these scenarios, it is feasible to utilize the existing image restoration model by substituting the image classification module. The method could also be applied to other types of brain pathologies, such as recognizing Alzheimer's disease, using the same restoration model and changing only the classification model. Similarly, the approach could be used for MRI of other body parts besides the brain to recognize other types of pathologies. Another case in which the method could be useful is in compressed sensing. One major problem in MRI is the long acquisition time; compressed sensing enables faster image acquisition, but has risks in image quality reduction. In the last case, the artifact generator function would be replaced by a function to simulate the MRI with fewer frequency components.

REFERENCES

ABDELHAMED, A.; LIN, S.; BROWN, M. S. A high-quality denoising dataset for smartphone cameras. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2018. p. 1692–1700. Cited 4 times in pages ix, 26, 60, and 105.

ANDRIUSHCHENKO, M.; D'ANGELO, F.; VARRE, A.; FLAMMARION, N. Why do we need weight decay in modern deep learning? *arXiv preprint arXiv:2310.04415*, 2023. Cited in page 49.

BA, J. L.; KIROS, J. R.; HINTON, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. Cited 3 times in pages 35, 42, and 59.

BADŽA, M. M.; BARJAKTAROVIĆ, M. Č. Classification of brain tumors from mri images using a convolutional neural network. *Applied Sciences*, MDPI, v. 10, n. 6, p. 1999, 2020. Cited 4 times in pages 1, 9, 10, and 60.

BAHDANAU, D.; CHO, K.; BENGIO, Y. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. Cited in page 39.

BASU, S.; FLETCHER, T.; WHITAKER, R. Rician noise removal in diffusion tensor mri. In: SPRINGER. *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2006: 9th International Conference, Copenhagen, Denmark, October 1-6, 2006. Proceedings, Part I 9*. [S.l.], 2006. p. 117–125. Cited in page 17.

BAUCHSPIESS, R.; FARIAS, M. C. A degradation-robust deep learning framework for mri brain tumor diagnosis. In: IEEE. *2024 IEEE International Symposium on Biomedical Imaging (ISBI)*. [S.l.], 2024. p. 1–4. Cited 6 times in pages v, vii, 2, 4, 57, and 66.

BECKMANN, P. Rayleigh distribution and its generalizations. *Radio Sci. J. Res. NBS/USNC-URSI*, v. 68, n. 9, p. 927–932, 1964. Cited in page 17.

BHUVAJI, S.; KADAM, A.; BHUMKAR, P.; DEDGE, S.; KANCHAN, S. *Brain Tumor Classification (MRI)*. Kaggle, 2020. Disponível em: <https://www.kaggle.com/dsv/1183165>. Cited in page 62.

BONDY, M. L.; SCHEURER, M. E.; MALMER, B.; BARNHOLTZ-SLOAN, J. S.; DAVIS, F. G.; IL'YASOVA, D.; KRUCHKO, C.; MCCARTHY, B. J.; RAJARAMAN, P.; SCHWARTZBAUM, J. A. *et al.* Brain tumor epidemiology: consensus from the brain tumor epidemiology consortium. *Cancer*, Wiley Online Library, v. 113, n. S7, p. 1953–1968, 2008. Cited in page 11.

BRENNER, D. J.; HALL, E. J. Computed tomography—an increasing source of radiation exposure. *New England journal of medicine*, Mass Medical Soc, v. 357, n. 22, p. 2277–2284, 2007. Cited in page 1.

BRIDLE, J. S. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In: SPRINGER. *Neurocomputing: Algorithms, architectures and applications*. [S.l.], 1990. p. 227–236. Cited in page 31.

BUI, F.; BOTT, K.; MINTCHEV, M. A quantitative study of the pixel-shifting, blurring and nonlinear distortions in mri images caused by the presence of metal implants. *Journal of Medical Engineering & Technology*, Taylor & Francis, v. 24, n. 1, p. 20–27, 2000. Cited in page 21.

CÁRDENAS-BLANCO, A.; TEJOS, C.; IRARRAZAVAL, P.; CAMERON, I. Noise in magnitude magnetic resonance images. *Concepts in Magnetic Resonance Part A: An Educational Journal*, Wiley Online Library, v. 32, n. 6, p. 409–416, 2008. Cited 3 times in pages 14, 16, and 17.

CHANDARANA, H.; WANG, H.; TIJSSEN, R.; DAS, I. J. Emerging role of mri in radiation therapy. *Journal of Magnetic Resonance Imaging*, Wiley Online Library, v. 48, n. 6, p. 1468–1478, 2018. Cited in page 1.

CHARBONNIER, P.; BLANC-FERAUD, L.; AUBERT, G.; BARLAUD, M. Two deterministic half-quadratic regularization algorithms for computed imaging. In: IEEE. *Proceedings of 1st international conference on image processing*. [S.l.], 1994. v. 2, p. 168–172. Cited 2 times in pages 53 and 54.

CHATZELLIS, E.; ALEXANDRAKI, K. I.; ANDROULAKIS, I. I.; KALTSAS, G. Aggressive pituitary tumors. *Neuroendocrinology*, S. Karger AG Basel, Switzerland, v. 101, n. 2, p. 87–104, 2015. Cited in page 11.

CHEN, P.; LIU, S.; ZHAO, H.; JIA, J. Gridmask data augmentation. *arXiv preprint arXiv:2001.04086*, 2020. Cited in page 54.

CHENG, J.; YANG, W.; HUANG, M.; HUANG, W.; JIANG, J.; ZHOU, Y.; YANG, R.; ZHAO, J.; FENG, Y.; FENG, Q. *et al.* Retrieval of brain tumors by adaptive spatial pooling and fisher vector representation. *PloS one*, Public Library of Science San Francisco, CA USA, v. 11, n. 6, 2016. Cited 14 times in pages v, vii, 9, 11, 12, 13, 14, 15, 60, 61, 62, 87, 89, and 105.

CHU, X.; TIAN, Z.; WANG, Y.; ZHANG, B.; REN, H.; WEI, X.; XIA, H.; SHEN, C. Twins: Revisiting the design of spatial attention in vision transformers. *Advances in Neural Information Processing Systems*, v. 34, p. 9355–9366, 2021. Cited in page 43.

CIAMPICONI, L.; ELWOOD, A.; LEONARDI, M.; MOHAMED, A.; ROZZA, A. *A survey and taxonomy of loss functions in machine learning*. 2023. Disponível em: <https://arxiv.org/abs/2301.05579>. Cited in page 53.

COUPÉ, P.; MANJÓN, J. V.; GEDAMU, E.; ARNOLD, D.; ROBLES, M.; COLLINS, D. L. Robust rician noise estimation for mr images. *Medical image analysis*, Elsevier, v. 14, n. 4, p. 483–493, 2010. Cited in page 17.

CUBUK, E. D.; ZOPH, B.; MANE, D.; VASUDEVAN, V.; LE, Q. V. Autoaugment: Learning augmentation strategies from data. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. [S.l.: s.n.], 2019. p. 113–123. Cited 2 times in pages 55 and 66.

DAI, C.; KANG, J.; LIU, X.; YAO, Y.; WANG, H.; WANG, R. How to classify and define pituitary tumors: Recent advances and current controversies. *Frontiers in Endocrinology*, v. 12, 2021. ISSN 1664-2392. Disponível em: <https://www.frontiersin.org/journals/endocrinology/articles/10.3389/fendo.2021.604644>. Cited in page 12.

DAI, Z.; LIU, H.; LE, Q. V.; TAN, M. Coatnet: Marrying convolution and attention for all data sizes. *Advances in Neural Information Processing Systems*, v. 34, p. 3965–3977, 2021. Cited in page 44.

DAIMARY, D.; BORA, M.; AMITAB, k.; KANDAR, D. Brain tumor segmentation from mri images using hybrid convolutional neural networks. *Procedia Computer Science*, v. 167, p. 2419–2428, 01 2020. Cited 2 times in pages 14 and 15.

DEANGELIS, L. M. Brain tumors. *New England journal of medicine*, Mass Medical Soc, v. 344, n. 2, p. 114–123, 2001. Cited 3 times in pages 1, 11, and 12.

DENG, J.; DONG, W.; SOCHER, R.; LI, L.-J.; LI, K.; FEI-FEI, L. Imagenet: A large-scale hierarchical image database. In: IEEE. *2009 IEEE conference on computer vision and pattern recognition*. [S.l.], 2009. p. 248–255. Cited 2 times in pages 34 and 57.

DEVRIES, T.; TAYLOR, G. W. Improved regularization of convolutional neural networks with cutout. *CoRR*, abs/1708.04552, 2017. Disponível em: <http://arxiv.org/abs/1708.04552>. Cited in page 54.

DODGE, S.; KARAM, L. Understanding how image quality affects deep neural networks. In: IEEE. *2016 eighth international conference on quality of multimedia experience (QoMEX)*. [S.l.], 2016. p. 1–6. Cited in page 2.

DOSOVITSKIY, A.; BEYER, L.; KOLESNIKOV, A.; WEISSENBORN, D.; ZHAI, X.; UNTERTHINER, T.; DEHGHANI, M.; MINDERER, M.; HEIGOLD, G.; GELLY, S. *et al.* An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. Cited 6 times in pages vi, 9, 41, 42, 114, and 115.

EL-SHAFAI, W.; EL-NABI, S. A.; EL-RABAIE, E.-S. M.; ALI, A. M.; SOLIMAN, N. F.; ALGARNI, A. D.; EL-SAMIE, A.; FATHI, E. Efficient deep-learning-based autoencoder denoising approach for medical image diagnosis. *Computers, Materials & Continua*, v. 70, n. 3, 2022. Cited 3 times in pages 2, 6, and 8.

EL-SHAFAI, W.; EL-NABI, S. A.; EL-RABAIE, E.-S. M.; ALI, A. M.; SOLIMAN, N. F.; ALGARNI, A. D.; EL-SAMIE, A.; FATHI, E. Efficient deep-learning-based autoencoder denoising approach for medical image diagnosis. *Computers, Materials & Continua*, v. 70, n. 3, 2022. Cited 2 times in pages 102 and 103.

FARIAS, M.; OLIVEIRA, P. de C.; LOPES, G. dos S.; MIOSSO, C.; LIMA, J. The influence of magnetic resonance imaging artifacts on cnn-based brain cancer detection algorithms. *Computational Mathematics and Modeling*, Springer, v. 33, n. 2, p. 211–229, 2022. Cited 3 times in pages 2, 16, and 60.

FREEDMAN, D.; PISANI, R.; PURVES, R. Statistics (international student edition). *Pisani, R. Purves, 4th edn. WW Norton & Company, New York*, 2007. Cited in page 90.

FUKUSHIMA, K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, Springer, v. 36, n. 4, p. 193–202, 1980. Cited in page 30.

GALLAGHER, T. A.; NEMETH, A. J.; HACEIN-BEY, L. An introduction to the fourier transform: relationship to mri. *American journal of roentgenology*, Am Roentgen Ray Soc, v. 190, n. 5, p. 1396–1405, 2008. Cited 2 times in pages 14 and 18.

GHIASI, G.; LIN, T.-Y.; LE, Q. V. Dropblock: A regularization method for convolutional networks. *Advances in neural information processing systems*, v. 31, 2018. Cited in page 50.

GLOROT, X.; BENGIO, Y. Understanding the difficulty of training deep feedforward neural networks. In: JMLR WORKSHOP AND CONFERENCE PROCEEDINGS. *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. [S.l.], 2010. p. 249–256. Cited in page 48.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>. Cited in page 49.

GRAHAM, B.; EL-NOUBY, A.; TOUVRON, H.; STOCK, P.; JOULIN, A.; JÉGOU, H.; DOUZE, M. Levit: a vision transformer in convnet's clothing for faster inference. In: *Proceedings of the IEEE/CVF international conference on computer vision*. [S.l.: s.n.], 2021. p. 12259–12269. Cited 3 times in pages 42, 43, and 44.

GUDBJARTSSON, H.; PATZ, S. The rician distribution of noisy mri data. *Magnetic resonance in medicine*, Wiley Online Library, v. 34, n. 6, p. 910–914, 1995. Cited 4 times in pages v, 16, 17, and 18.

HAMADA, A. *brain tumor detection 2020. Kaggle*. 2020. Disponível em: <kaggle.com/datasets/ahmedhamada0/brain-tumor-detection>. Cited 5 times in pages vii, x, 61, 62, and 122.

HANSON, S.; PRATT, L. Comparing biases for minimal network construction with back-propagation. *Advances in neural information processing systems*, v. 1, 1988. Cited in page 49.

HE, K.; ZHANG, X.; REN, S.; SUN, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: *Proceedings of the IEEE international conference on computer vision*. [S.l.: s.n.], 2015. p. 1026–1034. Cited 3 times in pages vi, 32, and 48.

HE, K.; ZHANG, X.; REN, S.; SUN, J. Deep residual learning for image recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2016. p. 770–778. Cited 6 times in pages 36, 50, 55, 83, 114, and 115.

HE, T.; ZHANG, Z.; ZHANG, H.; ZHANG, Z.; XIE, J.; LI, M. Bag of tricks for image classification with convolutional neural networks. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. [S.l.: s.n.], 2019. p. 558–567. Cited 4 times in pages vi, 50, 51, and 83.

HENDRYCKS, D.; GIMPEL, K. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. Cited 3 times in pages vi, 33, and 59.

HEO, B.; YUN, S.; HAN, D.; CHUN, S.; CHOE, J.; OH, S. J. Rethinking spatial dimensions of vision transformers. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. [S.l.: s.n.], 2021. p. 11936–11945. Cited in page 42.

HOCHREITER, S. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, v. 6, p. 107–116, 04 1998. Cited 2 times in pages 30 and 48.

HOWARD, A.; SANDLER, M.; CHU, G.; CHEN, L.-C.; CHEN, B.; TAN, M.; WANG, W.; ZHU, Y.; PANG, R.; VASUDEVAN, V. *et al.* Searching for mobilenetv3. In: *Proceedings of the IEEE/CVF international conference on computer vision*. [S.l.: s.n.], 2019. p. 1314–1324. Cited in page 56.

HU, J.; SHEN, L.; SUN, G. Squeeze-and-excitation networks. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2018. p. 7132–7141. Cited 4 times in pages vi, 37, 38, and 56.

HUANG, G.; LIU, Z.; MAATEN, L. V. D.; WEINBERGER, K. Q. Densely connected convolutional networks. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2017. p. 4700–4708. Cited 4 times in pages 36, 55, 114, and 115.

HUANG, Z.; BEN, Y.; LUO, G.; CHENG, P.; YU, G.; FU, B. Shuffle transformer: Rethinking spatial shuffle for vision transformer. *arXiv preprint arXiv:2106.03650*, 2021. Cited 2 times in pages vi and 43.

HYUN, C. M.; KIM, H. P.; LEE, S. M.; LEE, S.; SEO, J. K. Deep learning for undersampled mri reconstruction. *Physics in Medicine & Biology*, IOP Publishing, v. 63, n. 13, p. 135007, 2018. Cited in page 1.

ILIC, I.; ILIC, M. International patterns and trends in the brain cancer incidence and mortality: An observational study based on the global burden of disease. *Heliyon*, Elsevier, v. 9, n. 7, 2023. Cited in page 1.

INOUE, H. Data augmentation by pairing samples for images classification. *arXiv preprint arXiv:1801.02929*, 2018. Cited 2 times in pages 54 and 55.

IOFFE, S.; SZEGEDY, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: PMLR. *International conference on machine learning*. [S.l.], 2015. p. 448–456. Cited 2 times in pages 34 and 56.

ISLAM, M. M.; TALUKDER, M. A.; UDDIN, M. A.; AKHTER, A.; KHALID, M. Brainnet: precision brain tumor classification with optimized efficientnet architecture. *International Journal of Intelligent Systems*, Wiley Online Library, v. 2024, n. 1, p. 3583612, 2024. Cited 5 times in pages 1, 9, 10, 29, and 57.

JABBAR, H.; KHAN, R. Z. Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study). *Computer Science, Communication and Instrumentation Devices*, Res. Publ Singapore, v. 70, n. 10.3850, p. 978–981, 2015. Cited 2 times in pages 2 and 47.

JARRETT, K.; KAVUKCUOGLU, K.; RANZATO, M.; LECUN, Y. What is the best multi-stage architecture for object recognition? In: IEEE. *2009 IEEE 12th international conference on computer vision*. [S.l.], 2009. p. 2146–2153. Cited in page 32.

JHA, D.; SMEDSRUD, P. H.; RIEGLER, M. A.; JOHANSEN, D.; LANGE, T. D.; HALVORSEN, P.; JOHANSEN, H. D. Resunet++: An advanced architecture for medical image segmentation. In: IEEE. *2019 IEEE international symposium on multimedia (ISM)*. [S.l.], 2019. p. 225–2255. Cited in page 45.

JHAMB, T. K.; REJATHALAL, V.; GOVINDAN, V. A review on image reconstruction through mri k-space data. *International journal of image, graphics and signal processing*, Modern Education and Computer Science Press, v. 7, n. 7, p. 42, 2015. Cited 2 times in pages 1 and 14.

JUREK, J.; MATERKA, A.; LUDWISIAK, K.; MAJOS, A.; GORCZEWSKI, K.; CEPUCH, K.; ZAWADZKA, A. Supervised denoising of diffusion-weighted magnetic resonance images using a convolutional neural network and transfer learning. *Biocybernetics and Biomedical Engineering*, Elsevier, v. 43, n. 1, p. 206–232, 2023. Cited 3 times in pages 2, 7, and 8.

KATTI, G.; ARA, S. A.; SHIREEN, A. Magnetic resonance imaging (mri)–a review. *International journal of dental clinics*, Celesta Software Private Limited, v. 3, n. 1, p. 65–70, 2011. Cited 2 times in pages 1 and 13.

KHALIGHI, S.; REDDY, K.; MIDYA, A.; PANDAV, K. B.; MADABHUSHI, A.; ABEDALTHAGAFI, M. Artificial intelligence in neuro-oncology: advances and challenges in brain tumor diagnosis, prognosis, and precision treatment. *NPJ Precision Oncology*, Nature Publishing Group UK London, v. 8, n. 1, p. 80, 2024. Cited 2 times in pages 1 and 2.

KIM, Y.; LEE, Y.; JEON, M. Imbalanced image classification with complement cross entropy. *Pattern Recognition Letters*, Elsevier, v. 151, p. 33–40, 2021. Cited in page 54.

KINGMA, D. P. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. Cited in page 52.

KOLESNIKOV, A.; BEYER, L.; ZHAI, X.; PUIGCERVER, J.; YUNG, J.; GELLY, S.; HOULSBY, N. Big transfer (bit): General visual representation learning. In: SPRINGER. *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*. [S.l.], 2020. p. 491–507. Cited 2 times in pages 48 and 49.

KOROLEV, S.; SAFIULLIN, A.; BELYAEV, M.; DODONOVA, Y. Residual and plain convolutional neural networks for 3d brain mri classification. In: IEEE. *14th international symposium on biomedical imaging (ISBI 2017)*. [S.l.], 2017. p. 835–838. Cited 2 times in pages 9 and 10.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, v. 25, 2012. Cited 2 times in pages 32 and 36.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, AcM New York, NY, USA, v. 60, n. 6, p. 84–90, 2017. Cited in page 32.

KULLBACK, S.; LEIBLER, R. A. On information and sufficiency. *The annals of mathematical statistics*, JSTOR, v. 22, n. 1, p. 79–86, 1951. Cited 2 times in pages 53 and 54.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *nature*, Nature Publishing Group UK London, v. 521, n. 7553, p. 436–444, 2015. Cited in page 31.

LEE, D.; JIN, K. H.; KIM, E. Y.; PARK, S.-H.; YE, J. C. Acceleration of mr parameter mapping using annihilating filter-based low rank hankel matrix (aloha). *Magnetic resonance in medicine*, Wiley Online Library, v. 76, n. 6, p. 1848–1864, 2016. Cited in page 7.

LEE, J.; JIN, K. H.; YE, J. C. Reference-free single-pass epi n yquist ghost correction using annihilating filter-based low rank h ankel matrix (aloha). *Magnetic resonance in medicine*, Wiley Online Library, v. 76, n. 6, p. 1775–1789, 2016. Cited 3 times in pages 2, 7, and 8.

LI, P.; PEI, Y.; LI, J. A comprehensive survey on design and application of autoencoder in deep learning. *Applied Soft Computing*, Elsevier, v. 138, p. 110176, 2023. Cited in page 44.

LI, Y.; GUO, F.; TAN, R. T.; BROWN, M. S. A contrast enhancement framework with jpeg artifacts suppression. In: FLEET, D.; PAJDLA, T.; SCHIELE, B.; TUYTELAARS, T. (Ed.). *Computer Vision – ECCV 2014*. Cham: Springer International Publishing, 2014. p. 174–188. ISBN 978-3-319-10605-2. Cited in page 21.

LIM, S.; KIM, I.; KIM, T.; KIM, C.; KIM, S. Fast autoaugment. *Advances in neural information processing systems*, v. 32, 2019. Cited 2 times in pages 55 and 66.

LIM, Y.; BLIESENER, Y.; NARAYANAN, S.; NAYAK, K. S. Deblurring for spiral real-time mri using convolutional neural networks. *Magnetic resonance in medicine*, Wiley Online Library, v. 84, n. 6, p. 3438–3452, 2020. Cited 3 times in pages 2, 7, and 8.

LIU, X.; CHEN, S.; CUI, D.; HUI, E. S.; CHAN, Q.; CHEN, N.-K.; CHANG, H.-C. A robust self-referenced 2d nyquist ghost correction for different mri-biomarker measurements based on multi-band interleaved epi. *Frontiers in Physics*, Frontiers, v. 10, p. 1298, 2023. Cited in page 23.

LIU, Z.; LIN, Y.; CAO, Y.; HU, H.; WEI, Y.; ZHANG, Z.; LIN, S.; GUO, B. Swin transformer: Hierarchical vision transformer using shifted windows. In: *Proceedings of the IEEE/CVF international conference on computer vision*. [S.l.: s.n.], 2021. p. 10012–10022. Cited 3 times in pages vi, 43, and 44.

LIU, Z.; MAO, H.; WU, C.; FEICHTENHOFER, C.; DARRELL, T.; XIE, S. A convnet for the 2020s. *CoRR*, abs/2201.03545, 2022. Disponível em: <https://arxiv.org/abs/2201.03545>. Cited 2 times in pages 114 and 115.

LOSHCHILOV, I. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. Cited 2 times in pages 51 and 53.

LOSHCHILOV, I.; HUTTER, F. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. Cited in page 50.

MA, N.; ZHANG, X.; ZHENG, H.-T.; SUN, J. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In: *Proceedings of the European conference on computer vision (ECCV)*. [S.l.: s.n.], 2018. p. 116–131. Cited 2 times in pages 114 and 115.

MAAS, A. L.; HANNUN, A. Y.; NG, A. Y. *et al.* Rectifier nonlinearities improve neural network acoustic models. In: ATLANTA, GEORGIA, USA. *Proc. icml*. [S.l.], 2013. v. 30, n. 1, p. 3. Cited 2 times in pages 32 and 58.

MAHARANA, K.; MONDAL, S.; NEMADE, B. A review: Data pre-processing and data augmentation techniques. *Global Transitions Proceedings*, v. 3, n. 1, p. 91–99, 2022. ISSN 2666-285X. International Conference on Intelligent Engineering Approach(ICIEA-2022). Disponível em: <https://www.sciencedirect.com/science/article/pii/S2666285X22000565>. Cited in page 54.

MAHDAVI, M.; MOGHADDAM, S. S.; ABBASI-KANGEVARI, M.; MOHAMMADI, E.; SHOBEIRI, P.; SHARIFI, G.; JAFARI, A.; REZAEI, N.; EBRAHIMI, N.; REZAEI, N. *et al.* National and subnational burden of brain and central nervous system cancers in iran, 1990–2019: Results from the global burden of disease study 2019. *Cancer Medicine*, Wiley Online Library, v. 12, n. 7, p. 8614–8628, 2023. Cited in page 1.

MAHMOOD, A.; KHAN, S. A.; HUSSAIN, S.; ALMAGHAYREH, E. M. An adaptive image contrast enhancement technique for low-contrast images. *IEEE Access*, v. 7, p. 161584–161593, 2019. Cited in page 21.

MAO, A.; MOHRI, M.; ZHONG, Y. *Cross-Entropy Loss Functions: Theoretical Analysis and Applications*. 2023. Disponível em: <https://arxiv.org/abs/2304.07288>. Cited in page 54.

MELMED, S. Pituitary tumors. *Endocrinology and metabolism clinics of North America*, NIH Public Access, v. 44, n. 1, p. 1, 2015. Cited in page 12.

MEZRICH, R. A perspective on k-space. *Radiology*, v. 195, n. 2, p. 297–315, 1995. Cited in page 14.

MORATAL, D.; VALLÉS-LUCH, A.; MARTÍ-BONMATÍ, L.; BRUMMER, M. E. k-space tutorial: an mri educational tool for a better understanding of k-space. *Biomedical imaging and intervention journal*, Department of Biomedical Imaging, University of Malaya, v. 4, n. 1, 2008. Cited in page 14.

MUCKLEY, M. J.; ADES-ARON, B.; PAPAIOANNOU, A.; LEMBERSKIY, G.; SOLOMON, E.; LUI, Y. W.; SODICKSON, D. K.; FIEREMANS, E.; NOVIKOV, D. S.; KNOLL, F. Training a neural network for gibbs and noise removal in diffusion mri. *Magnetic resonance in medicine*, Wiley Online Library, v. 85, n. 1, p. 413–428, 2021. Cited 7 times in pages 2, 7, 8, 45, 46, 102, and 103.

MÜLLER, S. G.; HUTTER, F. Trivialaugment: Tuning-free yet state-of-the-art data augmentation. In: *Proceedings of the IEEE/CVF international conference on computer vision*. [S.l.: s.n.], 2021. p. 774–782. Cited 2 times in pages 55 and 66.

MZOUGHI, H.; NJEH, I.; SLIMA, M. B.; HAMIDA, A. B.; MHIRI, C.; MAHFOUDH, K. B. Denoising and contrast-enhancement approach of magnetic resonance imaging glioblastoma brain tumors. *Journal of Medical Imaging*, Society of Photo-Optical Instrumentation Engineers, v. 6, n. 4, p. 044002–044002, 2019. Cited 3 times in pages 2, 6, and 8.

MZOUGHI, H.; NJEH, I.; WALI, A.; SLIMA, M. B.; BENHAMIDA, A.; MHIRI, C.; MAHFOUDHE, K. B. Deep multi-scale 3d convolutional neural network (cnn) for mri gliomas brain tumor classification. *Journal of Digital Imaging*, Springer, v. 33, p. 903–915, 2020. Cited in page 6.

NAIR, V.; HINTON, G. E. Rectified linear units improve restricted boltzmann machines. In: *Proceedings of the 27th international conference on machine learning (ICML-10)*. [S.l.: s.n.], 2010. p. 807–814. Cited in page 32.

NICKPARVAR, M. *Brain Tumor MRI Dataset*. Kaggle, 2021. Disponível em: <https://www.kaggle.com/dsv/2645886>. Cited 10 times in pages vii, x, 2, 62, 63, 64, 68, 75, 119, and 120.

OKSUZ, I. Brain mri artefact detection and correction using convolutional neural networks. *Computer Methods and Programs in Biomedicine*, Elsevier, v. 199, 2021. Cited in page 2.

OSZUST, M. Full-reference image quality assessment with linear combination of genetically selected quality measures. *PloS one*, Public Library of Science San Francisco, CA USA, v. 11, n. 6, p. e0158333, 2016. Cited in page 26.

PADMANABAN, S.; THIRUVENKADAM, K.; T., P.; THIRUMALAISELVI, M.; SIVASAKTHIVEL, R. A role of medical imaging techniques in human brain tumor treatment. v. 8, p. 565–568, 01 2020. Cited 2 times in pages 14 and 15.

PASZKE, A.; GROSS, S.; CHINTALA, S.; CHANAN, G.; YANG, E.; DEVITO, Z.; LIN, Z.; DESMAISON, A.; ANTIGA, L.; LERER, A. Automatic differentiation in pytorch. In: *NIPS-W*. [S.l.: s.n.], 2017. Cited in page 50.

POTHUGANTI, S. Review on over-fitting and under-fitting problems in machine learning and solutions. *Int. J. Adv. Res. Electr. Electron. Instrum. Eng*, v. 7, p. 3692–3695, 2018. Cited 2 times in pages 2 and 47.

QADRI, S.; NAWAZ, A.; SAHER, N.; REHMAN, M. ul; RAZZAQ, A.; AHMAD, N.; HUSSAIN, A.; ZAREEN, S. S.; QADRI, S. F. *Brain Tumor MR Image Data Set For Machine Vision Approach for Brain Tumor Classification using Multi Features Dataset*. Zenodo, 2022. Disponível em: <https://doi.org/10.5281/zenodo.7047164>. Cited 10 times in pages v, vii, ix, x, 15, 63, 64, 105, 120, and 121.

QIAN, N. On the momentum term in gradient descent learning algorithms. *Neural networks*, Elsevier, v. 12, n. 1, p. 145–151, 1999. Cited in page 52.

RADOSAVOVIC, I.; KOSARAJU, R. P.; GIRSHICK, R. B.; HE, K.; DOLLÁR, P. Designing network design spaces. *CoRR*, abs/2003.13678, 2020. Disponível em: <https://arxiv.org/abs/2003.13678>. Cited 2 times in pages 114 and 115.

RAMACHANDRAN, P.; PARMAR, N.; VASWANI, A.; BELLO, I.; LEVSKAYA, A.; SHLENS, J. Stand-alone self-attention in vision models. *Advances in neural information processing systems*, v. 32, 2019. Cited in page 44.

RAMACHANDRAN, P.; ZOPH, B.; LE, Q. V. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017. Cited 2 times in pages 33 and 56.

RAYLEIGH, J. W. S. B. *The theory of sound*. [S.l.]: Macmillan, 1896. v. 2. Cited in page 17.

REBUFFI, S.-A.; GOWAL, S.; CALIAN, D. A.; STIMBERG, F.; WILES, O.; MANN, T. A. Data augmentation can improve robustness. *Advances in Neural Information Processing Systems*, v. 34, p. 29935–29948, 2021. Cited in page 2.

REHMAN, A.; NAZ, S.; RAZZAK, M. I.; AKRAM, F.; IMRAN, M. A deep learning-based framework for automatic brain tumors classification using transfer learning. *Circuits, Systems, and Signal Processing*, Springer, v. 39, n. 2, p. 757–775, 2020. Cited 3 times in pages 1, 9, and 10.

RICE, S. O. Mathematical analysis of random noise. *The Bell System Technical Journal*, Nokia Bell Labs, v. 23, n. 3, p. 282–332, 1944. Cited in page 16.

RODRIGUES, R.; LÉVÊQUE, L.; GUTIÉRREZ, J.; JEBBARI, H.; OUTTAS, M.; ZHANG, L.; CHETOUANI, A.; AL-JUBOORI, S.; MARTINI, M.; PINHEIRO, A. M. Objective quality assessment of medical images and videos: Review and challenges. *arXiv preprint arXiv:2212.07396*, 2022. Cited 4 times in pages 2, 26, 29, and 89.

RONNEBERGER, O.; FISCHER, P.; BROX, T. U-net: Convolutional networks for biomedical image segmentation. In: SPRINGER. *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. [S.l.], 2015. p. 234–241. Cited 3 times in pages vi, 45, and 46.

ROSENBLATT, F. The perceptron - a probabilistic model for information-storage and organization in the brain. *PSYCHOLOGICAL REVIEW*, v. 65, n. 6, p. 386–408, 1958. ISSN 0033-295X. Cited 2 times in pages 29 and 31.

RUDER, S. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016. Cited 2 times in pages 51 and 52.

RUMELHART, D. E.; HINTON, G. E.; MCCLELLAND, J. L. *et al.* A general framework for parallel distributed processing. *Parallel distributed processing: Explorations in the microstructure of cognition*, Cambridge, MA: MIT Press, v. 1, n. 45-76, p. 26, 1986. Cited in page 30.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. *nature*, Nature Publishing Group UK London, v. 323, n. 6088, p. 533–536, 1986. Cited 2 times in pages 30 and 46.

SANDLER, M.; HOWARD, A.; ZHU, M.; ZHMOGINOV, A.; CHEN, L.-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2018. p. 4510–4520. Cited in page 56.

SCHWARTZBAUM, J. A.; FISHER, J. L.; ALDAPE, K. D.; WRENSCH, M. Epidemiology and molecular pathology of glioma. *Nature clinical practice Neurology*, Nature Publishing Group UK London, v. 2, n. 9, p. 494–503, 2006. Cited in page 11.

SHORTEN, C.; KHOSHGOFTAAR, T. M. A survey on image data augmentation for deep learning. *Journal of big data*, Springer, v. 6, n. 1, p. 1–48, 2019. Cited 2 times in pages 54 and 55.

SIAR, M.; TESHNEHLAB, M. Brain tumor detection using deep neural network and machine learning algorithm. In: IEEE. *2019 9th international conference on computer and knowledge engineering (ICCKE)*. [S.l.], 2019. p. 363–368. Cited in page 60.

SIAR, M.; TESHNEHLAB, M. A combination of feature extraction methods and deep learning for brain tumour classification. *IET Image Processing*, Wiley Online Library, v. 16, n. 2, p. 416–441, 2022. Cited 8 times in pages vii, x, 2, 60, 61, 105, 123, and 124.

SINGH, K. K.; YU, H.; SARMASI, A.; PRADEEP, G.; LEE, Y. J. Hide-and-seek: A data augmentation technique for weakly-supervised localization and beyond. *arXiv preprint arXiv:1811.02545*, 2018. Cited in page 54.

SMITH, S. L.; KINDERMANS, P.-J.; YING, C.; LE, Q. V. Don't decay the learning rate, increase the batch size. *arXiv preprint arXiv:1711.00489*, 2017. Cited in page 51.

SPEARMAN, C. nthe proof and measurement of association between two things, oamerican j. *Psychol*, 1904. Cited in page 89.

SRIVASTAVA, N.; HINTON, G.; KRIZHEVSKY, A.; SUTSKEVER, I.; SALAKHUTDINOV, R. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, JMLR. org, v. 15, n. 1, p. 1929–1958, 2014. Cited in page 50.

SUMMERS, C.; DINNEEN, M. J. Improved mixed-example data augmentation. In: IEEE. *2019 IEEE winter conference on applications of computer vision (WACV)*. [S.l.], 2019. p. 1262–1270. Cited 2 times in pages 54 and 55.

SUTSKEVER, I.; MARTENS, J.; DAHL, G.; HINTON, G. On the importance of initialization and momentum in deep learning. In: PMLR. *International conference on machine learning*. [S.l.], 2013. p. 1139–1147. Cited in page 52.

TAKAHASHI, R.; MATSUBARA, T.; UEHARA, K. Data augmentation using random image cropping and patching for deep cnns. *IEEE Transactions on Circuits and Systems for Video Technology*, IEEE, v. 30, n. 9, p. 2917–2931, 2019. Cited 2 times in pages 54 and 55.

TAN, M.; LE, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In: PMLR. *International conference on machine learning*. [S.l.], 2019. p. 6105–6114. Cited 8 times in pages 2, 9, 37, 55, 56, 57, 114, and 115.

TAN, M.; LE, Q. Efficientnetv2: Smaller models and faster training. In: PMLR. *International conference on machine learning*. [S.l.], 2021. p. 10096–10106. Cited in page 57.

TERVEN, J.; CORDOVA-ESPARZA, D. M.; RAMIREZ-PEDRAZA, A.; CHAVEZ-URBIOLA, E. A.; ROMERO-GONZALEZ, J. A. *Loss Functions and Metrics in Deep Learning*. 2024. Disponível em: <https://arxiv.org/abs/2307.02694>. Cited in page 53.

TOMAR, N. K.; SHERGILL, A.; RIEDERS, B.; BAGCI, U.; JHA, D. Transresu-net: Transformer based resu-net for real-time colonoscopy polyp segmentation. *arXiv preprint arXiv:2206.08985*, 2022. Cited in page 45.

TOMPSON, J.; GOROSHIN, R.; JAIN, A.; LECUN, Y.; BREGLER, C. Efficient object localization using convolutional networks. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2015. p. 648–656. Cited in page 50.

TUMMALA, S.; KADRY, S.; BUKHARI, S. A. C.; RAUF, H. T. Classification of brain tumor from magnetic resonance imaging using vision transformers ensembling. *Current Oncology*, MDPI, v. 29, n. 10, p. 7498–7511, 2022. Cited 3 times in pages 1, 9, and 10.

ULYANOV, D.; VEDALDI, A.; LEMPITSKY, V. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. Cited in page 35.

UPADHYAY, N.; WALDMAN, A. Conventional mri evaluation of gliomas. *The British journal of radiology*, The British Institute of Radiology. 36 Portland Place, London, W1B 1AT, v. 84, n. special_issue_2, p. S107–S111, 2011. Cited in page 11.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, Ł.; POLOSUKHIN, I. Attention is all you need. *Advances in neural information processing systems*, v. 30, 2017. Cited 7 times in pages vi, 31, 37, 38, 39, 40, and 41.

VERAART, J.; FIEREMANS, E.; JELESCU, I. O.; KNOLL, F.; NOVIKOV, D. S. Gibbs ringing in diffusion mri. *Magnetic resonance in medicine*, Wiley Online Library, v. 76, n. 1, p. 301–314, 2016. Cited in page 18.

VERMA, V.; LAMB, A.; BECKHAM, C.; NAJAFI, A.; MITLIAGKAS, I.; LOPEZ-PAZ, D.; BENGIO, Y. Manifold mixup: Better representations by interpolating hidden states. In: PMLR. *International conference on machine learning*. [S.l.], 2019. p. 6438–6447. Cited 2 times in pages 54 and 55.

VINCENT, P.; LAROCHELLE, H.; BENGIO, Y.; MANZAGOL, P.-A. Extracting and composing robust features with denoising autoencoders. In: *Proceedings of the 25th international conference on Machine learning*. [S.l.: s.n.], 2008. p. 1096–1103. Cited in page 45.

WALLACE, G. The jpeg still picture compression standard. *IEEE Transactions on Consumer Electronics*, v. 38, n. 1, p. xviii–xxxiv, 1992. Cited in page 23.

WANG, W.; XIE, E.; LI, X.; FAN, D.-P.; SONG, K.; LIANG, D.; LU, T.; LUO, P.; SHAO, L. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In: *Proceedings of the IEEE/CVF international conference on computer vision*. [S.l.: s.n.], 2021. p. 568–578. Cited 2 times in pages 42 and 43.

WANG, Z.; BOVIK, A. C.; SHEIKH, H. R.; SIMONCELLI, E. P. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, IEEE, v. 13, n. 4, p. 600–612, 2004. Cited in page 26.

WANG, Z.; CUN, X.; BAO, J.; ZHOU, W.; LIU, J.; LI, H. Uformer: A general u-shaped transformer for image restoration. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2022. p. 17683–17693. Cited 13 times in pages vi, 2, 7, 8, 45, 46, 54, 55, 57, 58, 83, 102, and 103.

WATSON, A. B.; AHUMADA, A. J. Blur clarified: A review and synthesis of blur discrimination. *Journal of vision*, The Association for Research in Vision and Ophthalmology, v. 11, n. 5, p. 10–10, 2011. Cited in page 21.

WESTBROOK, C. *MRI at a Glance*. [S.l.]: John Wiley & Sons, 2016. Cited in page 13.

WILBRAHAM, H. On a certain periodic function. *The Cambridge and Dublin Mathematical Journal*, v. 3, p. 198–201, 1848. Cited in page 18.

WU, Y.; HE, K. Group normalization. In: *Proceedings of the European conference on computer vision (ECCV)*. [S.l.: s.n.], 2018. p. 3–19. Cited 3 times in pages vi, 34, and 35.

XIE, Q.; LUONG, M.-T.; HOVY, E.; LE, Q. V. Self-training with noisy student improves imagenet classification. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. [S.l.: s.n.], 2020. p. 10687–10698. Cited in page 57.

XIE, S.; GIRSHICK, R.; DOLLÁR, P.; TU, Z.; HE, K. Aggregated residual transformations for deep neural networks. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2017. p. 1492–1500. Cited in page 42.

YANG, L.; SHAMI, A. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, v. 415, p. 295–316, 2020. ISSN 0925-2312. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0925231220311693>. Cited in page 46.

YAZDAN, S. A.; AHMAD, R.; IQBAL, N.; RIZWAN, A.; KHAN, A. N.; KIM, D.-H. An efficient multi-scale convolutional neural network based multi-class brain mri classification for samd. *Tomography*, MDPI, v. 8, n. 4, p. 1905–1927, 2022. Cited 3 times in pages 1, 9, and 10.

YIM, D.; KIM, B.; LEE, S. A deep convolutional neural network for simultaneous denoising and deblurring in computed tomography. *Journal of Instrumentation*, IOP Publishing, v. 15, n. 12, p. P12001, 2020. Cited 3 times in pages 2, 7, and 8.

YING, X. An overview of overfitting and its solutions. In: IOP PUBLISHING. *Journal of physics: Conference series*. [S.l.], 2019. v. 1168, p. 022022. Cited 3 times in pages 2, 47, and 49.

YUN, S.; HAN, D.; OH, S. J.; CHUN, S.; CHOE, J.; YOO, Y. Cutmix: Regularization strategy to train strong classifiers with localizable features. In: *Proceedings of the IEEE/CVF international conference on computer vision*. [S.l.: s.n.], 2019. p. 6023–6032. Cited 2 times in pages 54 and 55.

ZAGORUYKO, S.; KOMODAKIS, N. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. Cited in page 37.

ZAMIR, S. W.; ARORA, A.; KHAN, S.; HAYAT, M.; KHAN, F. S.; YANG, M.-H.; SHAO, L. Learning enriched features for real image restoration and enhancement. In: SPRINGER. *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXV 16*. [S.l.], 2020. p. 492–511. Cited in page 54.

ZAMIR, S. W.; ARORA, A.; KHAN, S.; HAYAT, M.; KHAN, F. S.; YANG, M.-H. Restormer: Efficient transformer for high-resolution image restoration. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. [S.l.: s.n.], 2022. p. 5728–5739. Cited 4 times in pages 7, 8, 45, and 46.

ZAR, J. H. Spearman rank correlation. *Encyclopedia of Biostatistics*, Wiley Online Library, v. 7, 2005. Cited in page 89.

ZEILER, M. D.; KRISHNAN, D.; TAYLOR, G. W.; FERGUS, R. Deconvolutional networks. In: IEEE. *2010 IEEE Computer Society Conference on computer vision and pattern recognition*. [S.l.], 2010. p. 2528–2535. Cited in page 36.

ZHANG, H. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. Cited 2 times in pages 54 and 55.

ZHAO, H.; GALLO, O.; FROSIO, I.; KAUTZ, J. Loss functions for neural networks for image processing. *arXiv preprint arXiv:1511.08861*, 2015. Cited in page 53.

ZHAO, Y.; SHEN, X.; CHEN, J.; QIAN, W.; MA, H.; SANG, L. loss for low-contrast medical image segmentation. *Machine Learning: Science and Technology*, IOP Publishing, v. 5, n. 1, p. 015013, 2024. Cited in page 21.

ZHONG, A.; LI, B.; LUO, N.; XU, Y.; ZHOU, L.; ZHEN, X. Image restoration for low-dose ct via transfer learning and residual network. *IEEE Access*, IEEE, v. 8, p. 112078–112091, 2020. Cited 5 times in pages 2, 6, 8, 102, and 103.

ZHONG, Z.; ZHENG, L.; KANG, G.; LI, S.; YANG, Y. Random erasing data augmentation. In: *Proceedings of the AAAI conference on artificial intelligence*. [S.l.: s.n.], 2020. v. 34, n. 07, p. 13001–13008. Cited in page 54.

ZHOU, Z.; SIDDIQUEE, M. M. R.; TAJBAKHSH, N.; LIANG, J. Unet++: A nested u-net architecture for medical image segmentation. In: SPRINGER. *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support: 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, Proceedings 4*. [S.l.], 2018. p. 3–11. Cited in page 45.