Missing MRI Pulse Sequence Synthesis using Multi-Modal Generative Adversarial Network

by

Anmol Sharma

B.Tech., Punjab Technical University, 2016

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

in the
School of Computing Science
Faculty of Applied Sciences

SIMON FRASER UNIVERSITY
Summer 2019

Approval

Name:	**Anmol Sharma**
Degree:	**Master of Science (Computing Science)**
Title:	**Missing MRI Pulse Sequence Synthesis using Multi-Modal Generative Adversarial Network**
Examining Committee:	**Chair:** Ze-Nian Li Professor **Ghassan Hamarneh** Senior Supervisor Professor **Maxwell Libbrecht** Supervisor Assistant Professor **Roger Tam** External Examiner Associate Professor Department of Radiology University of British Columbia
Date Defended:	**22 August, 2019**

Abstract

Magnetic resonance imaging (MRI) is being increasingly utilized to assess, diagnose, and plan treatment for a variety of diseases. The ability to visualize tissue in varied contrasts in the form of MR pulse sequences in a single scan provides valuable insights to physicians, as well as enabling automated systems performing downstream analysis. However many issues like prohibitive scan time, image corruption, different acquisition protocols, or allergies to certain contrast materials may hinder the process of acquiring multiple sequences for a patient. This poses challenges to both physicians and automated systems since complementary information provided by the missing sequences is lost. In this paper, we propose a variant of generative adversarial network (GAN) capable of leveraging redundant information contained within multiple available sequences in order to generate one or more missing sequences for a patient scan. The proposed network is designed as a multi-input, multi-output network which combines information from all the available pulse sequences and synthesizes the missing ones in a single forward pass. We demonstrate and validate our method on two brain MRI datasets each with four sequences, and show the applicability of the proposed method in simultaneously synthesizing all missing sequences in any possible scenario where either one, two, or three of the four sequences may be missing. We compare our approach with competing unimodal and multi-modal methods, and show that we outperform both quantitatively and qualitatively.

Keywords: generative adversarial networks, multi-modal, missing modality, pulse sequences, MRI, synthesis

Dedication

Dedicated to my loving family and friends.

Acknowledgements

First and foremost I would like to thank my senior supervisor Prof. Ghassan Hamarneh for putting his trust in me to undertake graduate studies at SFU. His incredible insights, passion for his field, and patience throughout our interactions made me appreciate research in medical image analysis even more than I originally did before joining. I'd also like to thank Dr. Mostafa Fatehi (VGH) with whom I had elaborate discussions about the intricacies of medical software and research. I cherished the amazing opportunities I got for clinical collaboration with VGH, which helped me form an informed perspective about the healthcare system, and allowed me to identify and one day possibly solve some of the problems that clinicians face in their practice. The inspiration for this work (MM-GAN) came from working on clinical data from VGH, where MR sequences for some patients were missing due to a variety of reasons. I'd also like to thank my examining committee Dr. Maxwell Libbrecht, and Dr. Roger Tam whose valuable comments improved my work.

I would further like to thank one of my closest friends and labmate, Ben Cardoen, for his patience, support and long intellectual discussions which made my time at the lab much more enjoyable. I'd also like to thank my other labmates Saeid Asgari, Hanene Ben Yedder, Saeed Izadi, and Darren Sutton for all the discussions and chats, and also to all other members of the Medical Image Analysis Lab for their support - it was a pleasure sharing this experience with all of you.

I would like to acknowledge NSERC-CREATE Bioinformatics Scholarship for their support during the 2018-2019 calendar year, which made this research possible, and also for introducing me to the extremely intriguing field of bioinformatics through two of their mandatory courses at BC Cancer Agency. I would also like to thank Dr. Jayasree Chakraborty and Dr. Abhishek Midya, who nurtured me throughout my bachelors to become a researcher that I am today. I will forever be grateful and indebted to them for their unconditional support during that time.

Finally, I'd like to thank my parents and my brother for their unconditional love and support throughout my journey in not just graduate school, but throughout life. I'd also like to thank my partner Harman, whose positivity and joyfulness never went dull, and picked me up whenever I needed it the most. Without them, I wouldn't be where I am today.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation and Context

Medical imaging forms the backbone of the modern healthcare systems, providing means to assess, diagnose, and plan treatments for a variety of diseases. Imaging techniques like computed tomography (CT), magnetic resonance imaging (MRI), X-Rays have been in use for over many decades. Magnetic resonance imaging (MRI) out of these is particularly interesting in the sense that a single MRI scan is a grouping of multiple pulse sequences, each of which provides varying tissue contrast views and spatial resolutions, without the use of radiation. These sequences are acquired by varying the spin echo and repetition times during scanning, and are widely used to show pathological changes in internal organs and muscoskeletal system. Some of the commonly acquired sequences are T_1-weighted, T_2-weighted, T_1-with-contrast-enhanced (T_{1c}), and T_2-fluid-attenuated inversion recovery (T_{2flair}), though there exist many more [1]. The common sequences are shown in Figure 1.1.

A combination of sequences provide both redundant and complimentary information to the physician about the imaged tissue, and certain diagnosis are best performed when a particular sequence is observed. For example, T_1 and T_{2flair} sequences provide clear delin-

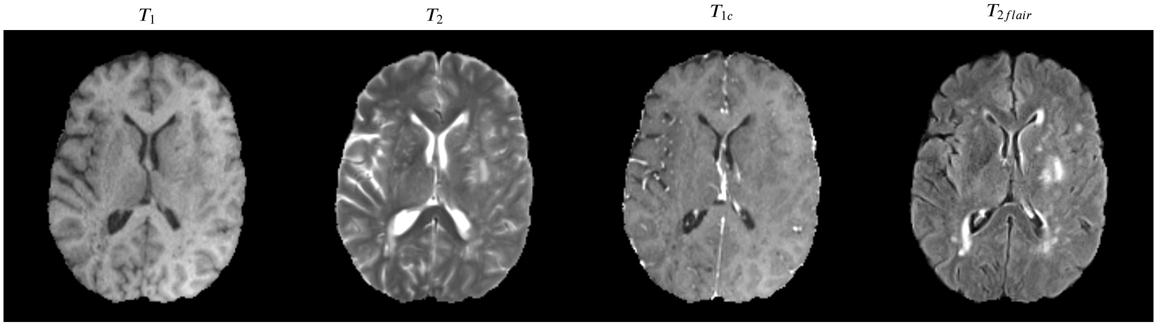

Figure 1.1: Common MR sequences acquired for a patient. Axial slices from a high grade glioma (HGG) patient from BraTS2018 are shown.

eations of the edema region of tumor in case of glioblastoma, T_{1c} provides clear demarcation of enhancing region around the tumor used as an indicator to assess growth/shrinkage, and T_{2flair} sequence is used to detect white matter hyperintensities for diagnosing vascular dementia (VD) [2].

In clinical settings, however, it is common to have MRI scans acquired using varying protocols, and hence varying sets of sequences per patient. Sequences which are routinely acquired may be unusable or missing altogether due to scan corruption, artifacts, incorrect machine settings, allergies to certain contrast agents and limited available scan time [3–5]. This phenomenon is problematic for many downstream data analysis pipelines that assume presence of a certain set of pulse sequences to perform their task. For instance, most of the segmentation methods [6–10] proposed for brain MRI scans depend implicitly on the availability of a certain set of sequences in their input in order to perform the task. Most of these methods are not designed to handle missing inputs, and hence may fail in the event where some or most of the sequences may be absent. This is also illustrated in Figure 1.2.

Modifying existing pipelines in order to handle missing sequences is hard, and may lead to performance degradation. Also, the option of redoing a scan to acquire the missing/corrupted sequence is impractical due to the expensive nature of the acquisition, longer wait times for patients with non-life-threatening cases, need for registration between old and new scans, and rapid changes in anatomy of area in-between scan times due to highly active abnormalities such as glioblastoma. Hence there is a clear advantage in retrieving any missing sequence or an estimate thereof, without having to redo the scan or changing the downstream pipelines.

To this end, we propose a multi-modal generative adversarial network (MM-GAN) which is capable of synthesizing missing sequences by combining information from all available sequences. The proposed method exhibits the ability to synthesize, with high accuracy, all the required sequences which are deemed missing in a single forward pass through the network. The term "multi-modal" simply refers to the fact that the GAN can take multiple-modalities of available information as input, which in this case represents different pulse sequences. Similar to the input being multi-modal, our method generates multi-modal output containing synthesized versions of the missing sequences. Since most of the downstream analysis pipelines commonly target $C = 4$ pulse sequences $S = \{T_1, T_{1c}, T_2, T_{2flair}\}$ as their input [7, 11, 12], we design our method around the same number of sequences, although we note that our method can be generalized to any number C and set S of sequences. The input to our network is a 4-channel (corresponding to $C = 4$ sequences) 2D axial slice, where a zero image is imputed for channels corresponding to missing sequences. The output of the network is a 4-channel 2D axial slice, in which the originally missing sequences are synthesized by the network.

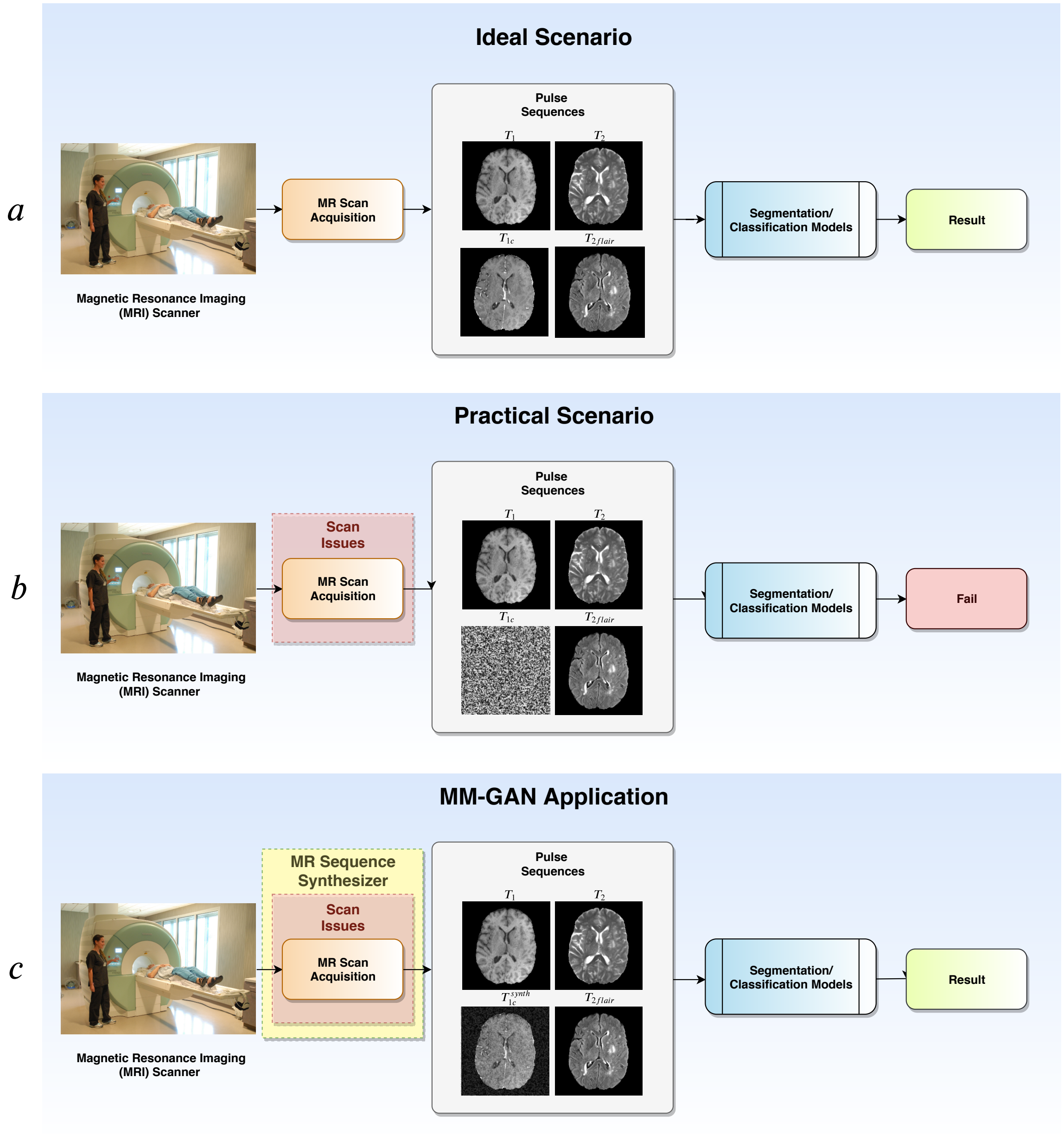

Figure 1.2: Illustration of scenarios where MR pulse sequences may be missing. (a) shows an ideal scenario where MR scan is acquired consisting of all the required sequences (T_1, T_2, T_{1c}, T_{2flair}). (b) shows practical scenario where the scan may be interrupted or the sequences get missing due to other reasons, leading to downstream deployed algorithms to fail due to their hard reliance on the particular set of sequences. (c) a valid use-case where MR pulse sequence synthesizer like MM-GAN may be inserted in the pipeline to generate missing sequences, and allowing the deployed downstream algorithms to still function without failing.

1.2 Related Work

There has been an increased amount of interest in developing methods for synthesizing MR pulse sequences [2, 5, 13–26]. We present a brief overview of previous work in this field by covering them in two sections: Unimodal, where both the input and output of the system is a single pulse sequence (one-to-one); and multimodal, where methods are able to leverage multiple input sequences to synthesize a single (many-to-one) or multiple sequences (many-to-many).

1.2.1 Unimodal Synthesis

In unimodal synthesis (one-to-one), a common strategy includes building an atlas or a database that maps intensity values between given sequences. Jog et al. [15] used a bagged ensemble of regression trees trained from an atlas. The training data $(\mathcal{A}_1, \mathcal{A}_2)$ consisted of multiple image patches $\mathcal{A}_1$ around a voxel i in a source sequence, and a single intensity value at the same voxel in a target sequence, as $\mathcal{A}_2$. The use of image patches to predict the intensity value of a single voxel in output sequence allows representing many-to-many relationship between intensity values of input and target sequences. Ye et al. [14] propose an inverse method, which performs a local patch-based search in a database for every voxel in the target pulse sequence. Once the patches are found, they are "fused" together using a data-driven regularization approach. Another atlas based method was proposed in [19] where T_2 whole-head sequence (including skull, eyes etc.) is synthesized from the available T_1 images. The synthesized T_2 sequence is used to correct distortion in diffusion-weighted MR images by using it as template for registration, in the absence of a real T_2 sequence. Yawen et al. [20] leverage joint dictionary learning (JDL) for synthesizing any unavailable MRI sequence from available MRI data. JDL is performed by minimizing the inconsistency between statistical distributions of the dictionary codes for input MRI sequences while preserving the geometrical structure of the input image.

Supervised machine learning and deep learning (DL) based methods have also been employed in sequence synthesis pipelines. A 3D continuous-valued conditional random field (CRF) is proposed in [18] to synthesize T_2 images from T_1. The synthesis step is encoded as a maximum a-posterior (MAP) estimate of Gaussian distribution parameters built from a learnt regression tree. Nguyen et al. [27] was one of the first to employ DL in the form of location-sensitive deep network (LSDN) for sequence synthesis. LSDN predicts the intensity value of the target voxel by using voxel-centered patches extracted from an input sequence. The network models the responses of hidden nodes as a product of feature and spatial responses. Similarly, Bowleset et al. [2] generate "pseudo-healthy" images by performing voxel-wise kernel regression instead of deep networks to learn local relationships between intensities in T_1 and T_{2flair} sequences of healthy subjects. Since most of the methods were based on local features in the form of patches and did not leverage global features of the input sequence, Sevetlidis et al. [21] proposed an encoder-decoder style deep neural network trained layer-wise using restricted Boltzmann machine (RBM) based training. The method utilized global context of the input sequence by taking a full slice as input. Recently, Jog et al. [22] propose a random forest based method that learns intensity mapping between input patches centered around a voxel extracted from a single pulse sequence, and the intensity of corresponding voxel in target sequence. The method utilized multi-resolution patches by building a Gaussian pyramid of the input sequence. Yu et al. [26] propose a unimodal GAN architecture to synthesize missing pulse sequences in a one-to-one setting. The approach uses an edge detection module that tries to preserve the high-frequency edge features of the input sequence, in the synthesized sequence. Recently, Ul Hassan Dar et al. [28] propose to use a conditional GAN to synthesize missing MR pulse sequences in a unimodal setting for two sequences T_1 and T_2.

1.2.2 Multimodal Synthesis

Multimodal synthesis has been a relatively new and unexplored avenue in MR synthesis literature. One of the first multi-input, single-output (many-to-one) method was proposed by Jog et al. [16]; a regression based approach to reconstruct T_{2flair} sequence using combined information from T_1, T_2, and proton density (PD) sequences. Reconstruction is performed by a bagged ensemble of regression trees predicting the T_{2flair} voxel intensities. Chartsias et al. [5] were one of the first to propose a multi-input, multi-output (many-to-many) encoder-decoder based architecture to perform many-to-many sequence synthesis, although their multimodal method is tested only using a single-output (T_{2flair}) (many-to-one setting). Their network is trained using a combination of three loss functions, and uses a feature fusion step in the middle that separates the encoders and decoders. Olut et al. [23] present a GAN based framework to generate magnetic resonance angiography (MRA) sequence from available T_1, and T_2 sequences. The method uses a novel loss function formulation, which preserves and reproduces vascularities in the generated images. Although for a different application, Mehta et al. [24] proposed a multi-task, multi-input, multi-output 3D CNN that outputs a segmentation mask of the tumor, as well as a synthesized version of T_{2flair} sequence. The main aim remains to predict tumor segmentation mask from three available sequences T_1, T_2, and T_{1c}, and no quantitative results for T_{2flair} synthesis using T_1, T_2, and T_{1c} are provided.

Though all the methods discussed above propose a multi-input method, none of the methods have been proposed to synthesize multiple missing sequences (multi-output), and in one single pass. All three methods [16], [5], and [24] synthesize only one sequence (either T_{2flair} or T_2, many-to-one setting) in the presence of varying number of input sequences, while [23] only synthesizes MRA using information from multiple inputs (many-to-one). Although the work presented in [23] is close to our proposed method, theirs is not a truly multimodal network (many-to-many), since there is no empirical evidence that their method will generalize to multiple scenarios. Similarly, the framework proposed in [5] can theoretically work in a many-to-many setting, but no empirical results are given to demonstrate its scalability and applicability in a variety of different scenarios, as we do in this work. The authors briefly touch upon this by adding a new decoder to already trained many-to-one network, but do not explore it any further. To the best of our knowledge, we are the first to propose a method that is capable of synthesizing multiple missing sequences using a combination of various input sequences (many-to-many), and demonstrate the method on the complete set of scenarios (i.e., all combinations of missing sequences).

The main motivation for most synthesis methods is to retain the ability to meaningfully use some downstream analysis pipelines like segmentation or classification despite the partially missing input. However, there have been efforts by researchers working on those analysis pipelines to bypass any synthesis step by making the analysis methods themselves robust to missing sequences. Most notably, Havaei et al. [3] and Varsavsky et al. [4] provide methods for tumor segmentation using brain MRI that are robust to missing sequences [3], or to missing sequence labels [4]. Although the methods bypass the requirement of having a synthesis step before actual downstream analysis, the performance of these robust versions of analysis pipelines often do not match the state-of-the-art performance of other non-robust methods in the case when all sequences are present. This is due to the fact that the methods not only have to learn how to perform the task (segmentation/classification) well, but also to handle any missing input data. This two-fold objective for a single network raises a trade-off between robustness and performance.

1.3 Thesis Contributions

The following are the key contributions of this work:

1. We propose the first multi-input multi-output MR pulse sequence synthesizer capable of synthesizing missing pulse sequences using *any combination of available sequences as input* without the need for tuning or retraining of models, in a many-to-many setting.

2. The proposed method is capable of synthesizing *any combination of target missing sequences as output* in one single forward pass, and requires only a single trained model for synthesis. This provides significant savings in terms of computational overhead during training time compared to training multiple models in the case of unimodal and multi-input single-output methods.

3. We propose to use implicit conditioning (IC), a combination of three design choices, namely imputation in place of missing sequences for input to generator, sequence-selective loss computation in the generator, and sequence-selective discrimination.

We show that IC improves overall quantitative synthesis performance of generator compared to the baseline approach without IC.

4. To the best of our knowledge, we are the first to incorporate curriculum learning based training for GAN by varying the difficulty of examples shown to the network during training.

5. Through experiments, we show that we outperform both the current state-of-art in unimodal (pGAN [28]), as well as the multi-input single-output synthesis (REPLICA [22] and MM-Synthesis [5]) method. We also set up new benchmarks on a complete set of scenarios using the BraTS2018 dataset.

1.4 Thesis Outline

The rest of the thesis is organized as follows: Chapter 2 presents detailed description of the proposed method, along with implementation details, datasets used, as well as outlines experimental setup for the current work. Chapter 3 discusses the results and observations for the proposed method, and finally the thesis is concluded in Chapter 4.

Chapter 2

Methodology

2.1 Background

Generative adversarial networks (GANs) were first proposed by Goodfellow et al. [29] in order to generate realistic looking images. A GAN is typically built using a combination of two networks: generator ($\mathcal{G}$) and discriminator ($\mathcal{D}$). The generator network is tasked with generating realistic data, typically by learning a mapping from a random vector z to an image I:

$$\mathcal{G} : z \rightarrow I, \tag{2.1}$$

where I is said to belong to the generator's distribution $p_{\mathcal{G}}$.

The discriminator:

$$\mathcal{D} : I \rightarrow t \tag{2.2}$$

maps its input I to a target label $t \in \{0, 1\}$, where $t = 0$ if $I \in p_{\mathcal{G}}$, i.e. a fake image generated by $\mathcal{G}$ and $t = 1$ if $I \in p_r$ where p_r is the distribution of real images. A variant of GANs, called conditional-GAN (cGAN) [30], proposes a generator that learns a mapping from a random vector z and a class label y to an output image I:

$$\mathcal{G} : (z, y) \rightarrow I, I \in p_{\mathcal{G}} \tag{2.3}$$

Another variant of cGAN called Pix2Pix [31] develops a GAN in which the generator learns a mapping from an input image $x \in p_r$ to output image $I \in p_{\mathcal{G}}$, $\mathcal{G} : x \rightarrow I$, and the discriminator learns a mapping from two input images, x_1 and x_2, to T, $\mathcal{D} : (x_1, x_2) \rightarrow T$. x_1 and x_2 may belong to either p_r (real) or $p_{\mathcal{G}}$ (fake). The output T in this case is a not a single class label, but a binary prediction tensor representing whether each $N \times N$ patch in the input image is real or fake [31].

A GAN is trained in an adversarial setting, where the generator (parameterized by $\theta_{\mathcal{G}}$) is trained to synthesize realistic output that can ”fool” the discriminator into classifying them as real, and the discriminator (parameterized by $\theta_{\mathcal{D}}$) is trained to accurately distinguish between real data and fake data synthesized by the generator. GAN input/outputs can be images [31], text [32] or even music [33]. Both the generator and discriminator act as adversaries to each other, hence the training formulation forces both networks to continuously get better at their tasks. GANs found tremendous success in a variety of different tasks, ranging from face-image synthesis [34], image stylization [35], future frame prediction in videos [36], text-to-image synthesis [32] and synthesizing scene images using scene attributes and semantic layout [37]. GANs have also been utilized in medical image analysis [38], particularly for image segmentation [39–41], normalization [42], synthesis [23, 26, 28] as well as image registration [43].

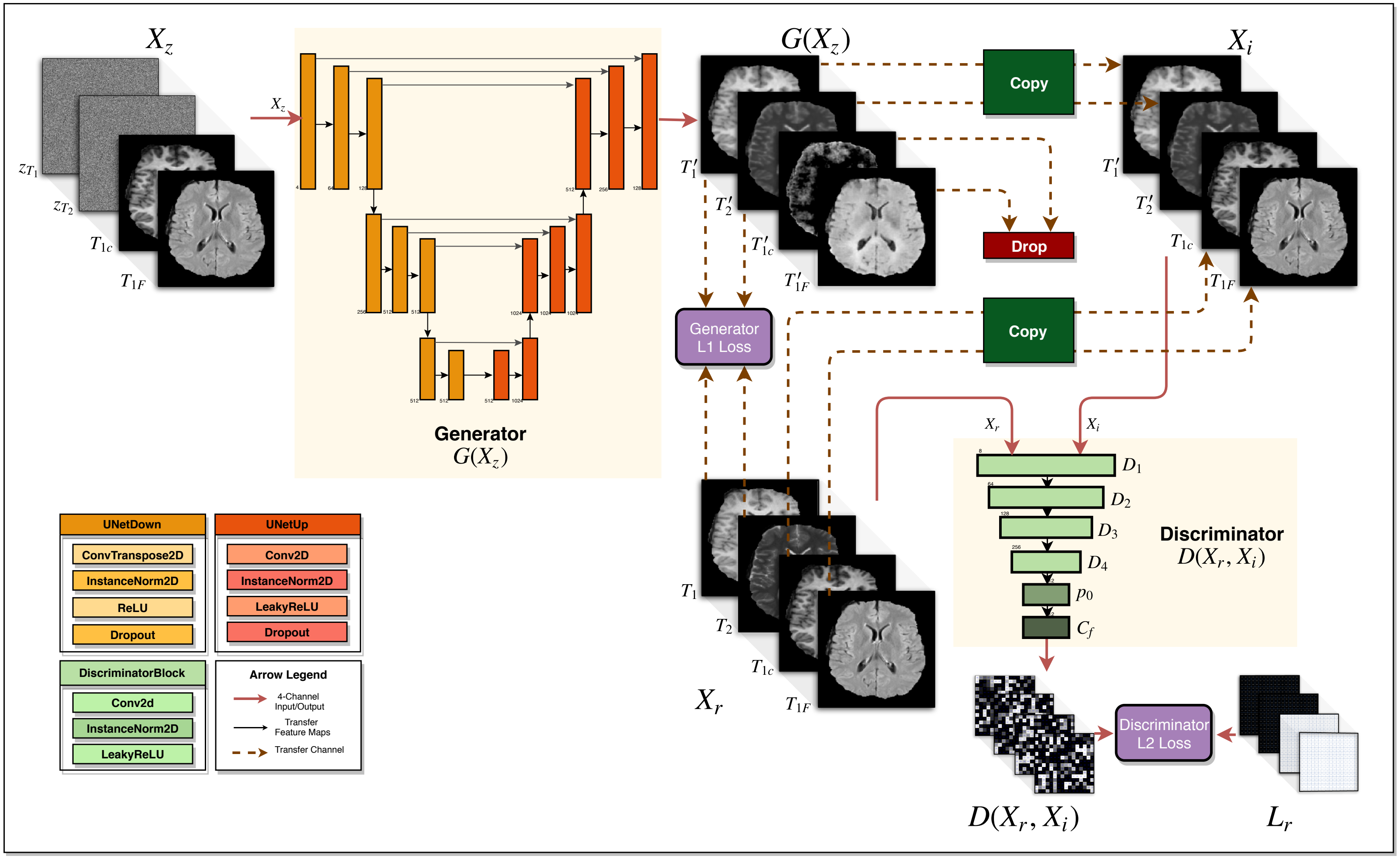

Figure 2.1: Proposed multimodal generative adversarial network (MM-GAN) training process. The generator is a UNet architecture, while the discriminator is a PatchGAN architecture with L2 loss (least squares GAN). The green "Copy" blocks transfer the input channels as is to its output, while the red "Drop" block deletes its input channels. The generator L1 loss is computed only between the synthesized versions of the missing sequences (here T_1 and T_2). The discriminator takes X_r and X_i as input and produces a 4-channel 2D output representing whether each $N \times N$ patch in X_i is either fake or real.

2.2 Proposed Method

We propose a variant of Pix2Pix architecture [31] called Multi-Modal Generative Adversarial Network (MM-GAN) for the task of synthesizing missing MR pulse sequences in a single forward pass while leveraging all available sequences. The following subsections would outline the detailed architecture of our model.

2.2.1 Generator

The generator of the proposed method is a UNet [44], which has proven useful in a variety of segmentation and synthesis tasks due to its contracting and expanding paths in the form of encoder and decoder subnetworks. The architecture is illustrated in Figure 2.1. The convolution kernel sizes for each layer in the generator is set to 4×4. The generator network is a combination of `UNetUp` and `UNetDown` blocks. The input to the generator is a 2D axial slice from a patient scan with $C = 4$ channels representing four pulse sequences, and spatial size of 256×256 pixels. The network is designed with a fixed input size of 4-channels, where channel $C = 0, 1, 2, \text{and } 3$ corresponds to T_1, T_2, T_{1c}, and T_{2flair}, respectively.

In order to synthesize missing sequences, the channels corresponding to each missing sequence are imputed with zeros. The imputed version (along with the real sequences) becomes the input to the generator and is represented by X_z. For instance, if sequences T_1 and T_2 are missing, channels $C = 0$ and $C = 1$ in the input image are imputed with a zero image of size 256×256. The output of the generator is given by $\mathcal{G}(X_z|\theta_{\mathcal{G}})$ and is of the same size as the input. Due to design, the generator always outputs 4 channels, however, as we outline in the subsequent text the output channels corresponding to the existing real sequences are not used for loss computation and are replaced with the real sequences before relaying them as input to the discriminator. During training the ground truth image X_r, short for “real”, which is of the same size as X_z contains all ground truth sequences at their respective channel indices. We use the term “image” for a single 2D slice with 4 channels.

MM-GAN observes both an input image and imputed zeros z in the form of X_z, in contrast to vanilla Pix2Pix where the generator is conditioned just by an observed image x. The reasons behind this design choice are discussed in subsection 2.2.3. We also investigate different imputation strategies in subsection 3.3.1, and found that zero based imputation performs the best quantitatively.

To optimize $\theta_{\mathcal{G}}$, our generator adopts the general form of the generator loss in Pix2Pix, which is a combination of a reconstruction loss $\mathcal{L}_1$ and an adversarial loss $\mathcal{L}_2$ used to train the generator to fool the discriminator, i.e.

$$\begin{aligned} \theta_{\mathcal{G}}^* = \underset{\theta_{\mathcal{G}}}{\arg\min}\ & \lambda \mathcal{L}_1(\mathcal{G}(X_z|\theta_{\mathcal{G}}), X_r) + \\ & (1 - \lambda)\ \mathcal{L}_2(\mathcal{D}(X_i, X_r|\theta_{\mathcal{D}}), L_{ar}). \end{aligned} \tag{2.4}$$

To calculate $\mathcal{L}_1$, we select synthesized sequences from $\mathcal{G}(X_z|\theta_\mathcal{G})$, that were originally missing, and compute the L1 norm of the difference between the synthesized versions of the sequence and the available ground truth from X_r. Mathematically, given the set K containing the indices of missing sequences (e.g. $K = \{0, 2\}$ when T_1 and T_{1c} are missing) in the current input, we calculate $\mathcal{L}_1$ only for the sequences that are missing ($k = 0, 2$), and sum the values.

To calculate $\mathcal{L}_2$, we compute the squared L2 norm of the difference between the discriminator's predictions $\mathcal{D}(X_i, X_r|\theta_\mathcal{D}))$ and a dummy ground truth tensor L_{ar} of the same size as the output of $\mathcal{D}$. In order to encourage the generator to synthesize sequences that confuse or "fool" the discriminator into predicting they are real, we set all entries of L_{ar} to ones, masquerading all generated sequences as real. X_i is introduced in the next section.

The choice of L1 as a reconstruction loss term for the generator is motivated by its ability to prevent too much blurring in the final synthesized sequences, as compared to using an L2 loss (similar to [31]).

2.2.2 Discriminator

We use the PatchGAN architecture [31] for the discriminator part of our MM-GAN. PatchGAN architecture learns to take into account the local characteristics of its input, by predicting a real/fake class for every $N \times N$ patch of its input, compared to classic GANs where the discriminator outputs a single real/fake prediction for the whole input image. This encourages the generator to synthesize images not just with proper global features (shape, size), but also with accurate local features (texture, distribution, high-frequency details). In our case we set $N = 16$.

The discriminator is built using four blocks followed by a zero padding layer and a final convolutional layer (Figure 2.1). The convolutional kernel sizes, stride and padding is identical to the values used in the generator (subsection 2.2.1). Due to the possibility of having a varying number of sequences missing, instead of providing just the synthesized sequences and their real counterparts as input to the discriminator, we first create a modified version of $\mathcal{G}(X_z|\theta_\mathcal{G})$ by dropping the reconstructed sequences that were originally present, and replacing them with the original sequences from X_r. The modified version of $\mathcal{G}(X_z|\theta_\mathcal{G})$ is represented by X_i, short for "imputed". The input to the discriminator is a concatenation of X_i and X_r. This is also illustrated in Figure 2.1.

The discriminator is trained to output a 2D patch of size 16×16 pixels, with 4 channels corresponding to each sequence. In order to supervise the discriminator during training, we use a 4-channel 2D image based target, in which each channel corresponds to a sequence. More specifically, given missing sequences K (e.g., $K = \{0, 2\}$, T_1 and T_{1c} missing), the target (i.e. ground truth) variable for $\mathcal{D}$ is:

$$L_r^k = \begin{cases} 0.0 \text{ (fake)} & \text{if } k \in K \\ 1.0 \text{ (real)} & \text{otherwise} \end{cases} \tag{2.5}$$

Note that L_r^k is a 2D tensor of size 16×16 (since each 256×256 image is divided into 16×16 patches) yet the assignment of 0.0 or 1.0 represents an assignment to the whole 16×16 L_r^k tensor (since the whole image is either real or fake and not patch-specific). This is also illustrated in Figure 2.1.

Between the output of discriminator $\mathcal{D}(X_i, X_r|\theta_{\mathcal{D}})$ and L_r, an L2 loss is computed. The final discriminator loss becomes:

$$\begin{aligned} \theta_{\mathcal{D}}^* = \arg\min_{\theta_{\mathcal{D}}} \, & \mathcal{L}_2(\mathcal{D}(X_r, X_r|\theta_{\mathcal{D}}), L_{ar}) \\ & + \mathcal{L}_2(\mathcal{D}(X_r, X_i|\theta_{\mathcal{D}}), L_r). \end{aligned} \tag{2.6}$$

This is equivalent to a least-squares GAN since the loss function incorporates an L2 loss.

2.2.3 Implicit conditioning

Due to the inherent design of deep learning architectures, the input as well as output of a convolutional neural network model has to have a fixed channel dimension. In our use case however, both the input and output channel dimensions vary (since the number of available sequences can vary).

In order to address this problem, we propose a combination of three design choices, which we collectively refer to as implicit conditioning (IC). In IC, the varying input channels problem is solved by imputing a fixed value (zeros) to the input channels where the sequences are missing. For the problem of generator output size being fixed in channel dimension, one possible approach can be to synthesize all four input sequences. The loss function can be calculated between four ground truth sequences, and the four synthesized sequences. However, this poses a challenge for the generator, as its burdened with generating all sequences, including the reconstruction of the ones that were provided as input. In order to address this, we proposed selective loss computation in $\mathcal{G}$, where the loss is only calculated between the ground truth sequences that were missing, and the corresponding output channels of the generators. In conjunction, we also propose selective discrimination in $\mathcal{D}$, which ensured stability during training by preventing the discriminator from overpowering the generator. We also show that IC-based training outperforms the baseline training methodology of generating and penalizing inaccurately synthesizing all sequences (subsection 3.3.2). The design choices are individually summarized below.

2.2.3.1 Input Imputation

The input X_z of the generator always contains an imputed value (z = zeros) in place of the missing sequences which acts as a way to condition the generator and informs which sequence(s) to synthesize.

2.2.3.2 Selective Loss Computation in Generator ($\mathcal{G}$)

In conjunction, the $\mathcal{L}_1(\mathcal{G})$ loss that is computed only between the synthesized sequences for the generator, and then backpropagated, allows the generator to align itself towards only synthesizing the actual missing sequences properly while ignoring the performance in synthesizing the ones that were already present.

2.2.3.3 Selective Discrimination in Discriminator ($\mathcal{D}$)

Imputing real sequences at the generator output (i.e. X_i) before providing it as discriminator input forces the discriminator to accurately learn to delineate only between the synthesized sequences and their real counterparts. Since the generator loss function also has a term that tries to fool the discriminator, this allows selective backpropagation into the generator where it is penalized only for incorrectly synthesizing the missing sequences, and not for incorrectly synthesizing the sequences that were already present. This relieves the generator of the difficult task of synthesizing all sequences in the presence of some sequences.

2.2.4 Curriculum Learning

In order to train our proposed method we use a curriculum learning (CL) [45] based approach. In CL based training, the network is initially shown easier examples followed by increasingly difficult examples as the training progresses. We hypothesized that CL can benefit in training of MM-GAN due to an ordering in the level of difficulty across the various scenarios that the network has to handle. If some cases are “easier” than others, it might be useful if the easier cases are shown first to the network in order to allow the network to effectively learn when ample supervision is available. As the network trains, “harder” cases can be introduced so that the network adapts without diverging.

In our work, scenarios with 1 sequence missing are considered “easy”, followed by a “moderate” set of scenarios with 2 missing sequences, and lastly, the scenarios with 3 missing sequences are considered “hard”. We adopted this ordering in our work, and showed the network easier examples first, followed by moderate and finally hard examples. After a threshold of 30 epochs, we show every scenario with uniform probability.

2.3 Experimental Setup

In this section we describe different aspects of the experiments that are performed in this work.

2.3.1 Datasets

In order to validate our method we use brain MRI datasets from two sources, namely the Ischemic Stroke Lesion Segmentation Challenge 2015 (ISLES2015) [46] and the Multimodal Brain Tumor Segmentation Challenge 2018 (BraTS2018) [47].

2.3.1.1 Ischemic Stroke Lesion Segmentation Challenge 2015

ISLES2015 dataset is a publicly available database with multi-spectral MR images [46]. We choose the sub-acute ischemic stroke lesion segmentation (SISS) cohort of patients, which contains 28 training and 36 testing cases. The patient scans are skull stripped using BET2 [48], and resampled to an isotropic resolution of 1 mm^3. Each scan consists of four sequences namely T_1, T_2, DWI, and T_{2flair}, and are rigidly co-registered to the T_{2flair} sequence using elastix tool-box [49]. More information about the preprocessing steps can be found in the original publication [46]. We use 22 patients from the SISS training set for experiments.

2.3.1.2 Brain Tumor Segmentation Challenge 2018

BraTS2018 consists of a total of 285 patient MR scans acquired from 19 different institutions, divided into two cohorts: glioblastoma/high grade glioma (GBM/HGG) and low grade glioma (LGG). The patient scans contains four pulse sequence T_1, T_2, T_{1c}, and T_{2flair}. All scans are resampled to 1 mm^3 isotropic resolution using a linear interpolator, skull stripped, and co-registered with a single anatomical template using rigid registration model with mutual information similarity metric. Detailed preprocessing information can be found in [47].

In order to demonstrate our method's ability in synthesizing sequences with both high grade and low grade glioma tumors present, we use a total of 210 patients from HGG and 75 patients from LGG cohort. 195 patients are reserved for training for HGG cohort, while 65 are used for training in LGG experiments. For validation, we use 5 patients for both HGG and LGG cohorts. In order to test our trained models, we use 10 patients from HGG cohort (due to larger data available), while we report results using 5 patients for LGG cohort as testing. To enable direct comparisons by future works, we release the list of patient images used as test set throughout this study in the Table 2.1. Patients other than those specified in Table 2.1 are used for training.

Table 2.1: Patient images used for testing from BraTS2018 HGG and LGG cohorts.

HGG Patients	**LGG Patients**
Brats18_2013_12_1	Brats18_2013_9_1
Brats18_CBICA_AAP_1	Brats18_TCIA10_449_1
Brats18_CBICA_APR_1	Brats18_TCIA12_466_1
Brats18_CBICA_ASA_1	Brats18_TCIA12_470_1
Brats18_CBICA_ASK_1	Brats18_TCIA13_654_1
Brats18_TCIA01_390_1	
Brats18_TCIA02_226_1	
Brats18_TCIA02_473_1	
Brats18_TCIA02_491_1	
Brats18_TCIA08_469_1	

2.3.2 Preprocessing

Each patient scan is normalized by dividing each sequence by its mean intensity value. This ensures that distribution of intensity values is preserved [5]. Normalization by mean is less sensitive to high or low intensity outliers as compared to min-max normalization procedures, which can be greatly exacerbated by the presence of just a single high or low intensity voxel in a sequence. This is especially common in the presence of various pathologies, like tumors as in BraTS2018 datasets, which tend to have very high intensity values in some sequences (T_2, T_{2flair}) and recessed intensities in others (T_1, T_{1c}). In practice, we observed this for BraTS2018 HGG cohort, where some voxels had an unusually high intensity value due to a pathology. On performing min-max normalization to scale intensities between [0,1], we found that the presence of very high intensity voxel squashed the pixel range to always lie very close to zero. This artificially bumped the performance numbers for the generator since most voxels lied close to zero, and hence the generator could synthesize images with intensity values close to zero, and achieve a low L1 score easily. On the other hand, mean normalization was relatively unaffected due to a large number of voxels in a defined range (0-4000). The mean value was not strongly affected by the presence of one or more high/low intensity voxels.

We also tested the method internally with zero mean and unit variance based standardization, and found the results to be at par with mean normalization. In order to crop out the brain region from each sequence, we calculate the largest bounding box that can accommodate each brain in the whole dataset, and then use the coordinates to crop each sequence in every patient scan. The final cropped size of a single patient scan with all sequences contains 148 axial slices of size 194×155. Each slice in every sequence is resized to a spatial resolution of 256×256, using bilinear interpolation, in order to maintain compatibility with UNet architecture of the generator.

We note that avoiding resampling twice (once during registration performed by the original authors of the dataset, and once during resampling to 256 × 256 in this work) may preserve some information in the scans that may otherwise be lost. However, it is a necessary preprocessing step in order to maintain compatibility with various network architectures that we utilize, which includes inherent assumptions that the input size would be a power of two to allow successive contracting and expanding steps used in many encoder-decoder style architectures. In order to fully avoid the second resampling step (to 256 × 256 in this work), a different network architecture may be used without the encoder-decoder setup, though the performance of those networks may not be at par with the modern encoder-decoder style networks as established in synthesis field [28, 31, 50]

2.3.3 Benchmark Methods

We compare our method with three competing methods, one unimodal and two multimodal. The unimodal (single-input, single-output, one-to-one) method we compare against is pGAN [28], while the multimodal (many-to-one) models being REPLICA [22] (in a multi-input setting), and that of Chartsias et al. [5], called MM-Synthesis hereafter. Both pGAN and MM-Synthesis were recently published (2019 and 2018), and they outperform all other methods before them (MM-Synthesis outperforms LSDN [27], Modality Propagation [14], and REPLICA [22], while pGAN outperforms both REPLICA and MM-Synthesis in one-to-one synthesis). To the best of our knowledge, we did not find any other methods that claimed to outperform either pGAN or MM-Synthesis, and so decided to evaluate our method against them.

For comparison with pGAN [28], we reimplement the method using the open source code provided with the publication, and train both pGAN and our method on a randomly chosen subset of data from BRATS2018 LGG cohort. We also compare with a standard baseline which is a vanilla Pix2Pix [31] model trained and tested in a one-to-one setting. For our multimodal (many-to-one) experiments, we report mean squared error (MSE) results for both REPLICA and MM-Synthesis directly from [5], as we recreate the exact same testbed for comparison with MM-GAN, as used in MM-Synthesis. We adopt the same testing strategy (5-fold cross validation), database (ISLES2015), and scenarios (7 scenarios where T_{2flair} is always missing and is the only one that is synthesized). As highlighted in [5], the multi-input version of REPLICA required seven models each for each of the seven valid scenarios in many-to-one setting synthesizing T_{2flair} sequence. MM-Synthesis and our proposed MM-GAN only required a single multimodal (many-to-one) network which generalized to all seven scenarios.

For our final extended set of experiments, we demonstrate the effectiveness of our method in a multi-input multi-output (many-to-many) setting, where we perform testing on the HGG and LGG cohorts of BRaTS2018 dataset for which we report results of all 14 valid scenarios (=16−2, as scenario when all sequences are missing/present are invalid for our experiments) instead of just 7. The results showcase our method's generalizability on different use-cases with varying input and output subsets of sequences, and different difficulty levels. We use a fixed order of sequences (T_1, T_2, T_{1c}, T_{2flair}) throughout this thesis, and represent each scenario as a 4-bit string, where a zero (0) represents the absence of a sequence at that location, while a one (1) represents its presence.

2.3.4 Training and Implementation Details

In order to optimize our networks, we use Adam [51] optimizer with learning rate $\eta = 0.0002$, $\beta_1 = 0.5$ and $\beta_2 = 0.999$. Both the generator and discriminator networks are initialized with weights sampled from a Gaussian distribution with $\mu = 0, \sigma = 0.02$.

We perform four experiments, first for establishing that multi-input synthesis is better than single-input (many-to-one vs one-to-one respectively), second for T_{2flair} synthesis using multiple inputs (many-to-one) (called MISO, short for multi-input single-output) using ISLES2015 dataset to compare with REPLICA and MM-Synthesis. The third set of experiments encompasses validation of multiple key components proposed throughout this thesis, in terms of their contribution towards overall network performance. We test different imputation strategies ($z = \{average, noise, zeros\}$), as well as the effect of curriculum learning (CL) and implicit conditioning (IC). The final set of experiments pertain to multimodal synthesis (MIMO, short for multi-input multi-output), which sets a new benchmark for many-to-many synthesis models using BraTS2018 HGG and LGG cohorts. We refer to the second and fourth experiments as MISO and MIMO, respectively, hereafter. We use a batch size of 4 slices to train models, except for MISO, where we use batch size of 2. We train the models for 30 epochs in MISO and 60 epochs for MIMO sets of experiments, with no data augmentation.

We choose $\lambda = 0.9$ for the generator loss given in equation 2.4, while we multiply the discriminator loss by 0.5 which essentially slows down the rate at which discriminator learns compared to generator. During each epoch, we alternate between a single gradient descent step on the generator, and one single step on the discriminator.

For our MIMO experiments, we use the original PatchGAN [31] discriminator. However for our MISO experiments, due to lack of training data, we used a smaller version of the PatchGAN discriminator with just two discriminator blocks, followed by a zero padding and final convolution layer. Also, random noise was added to both X_r and X_i inputs of the discriminator in MISO experiments. This was done to reduce the complexity of the discriminator to prevent it from overpowering the generator, which we observed when original PatchGAN with no noise imputation in its inputs was used for this set of experiments. The generator's final activation was set to ReLU for MIMO and linear for MISO experiments due to the latter having negative intensity values for some patients.

For our implementation we used Python as our main programming language. We implemented Pix2Pix architecture in PyTorch. The computing hardware consisted of an i7 CPU with 64 GB RAM and GTX1080Ti 12 GB VRAM GPU. Throughout our experiments we use random seeding in order to ensure reproducibility of our experiments. For our MIMO experiments, we use curriculum learning by raising the difficulty of scenarios every 10 epochs (starting from one missing sequence) that are shown to the network until epoch 30 (shown examples with three missing sequences), after which the scenarios are shown randomly with uniform probability until epoch 60. For MISO experiments we train the model without curriculum learning, and show all scenarios with uniform probability to the network for 30 epochs. MM-GAN takes an average time of 0.1536 ± 0.0070 seconds per patient as it works in constant time at test-time w.r.t number of sequences missing.

2.3.5 Evaluation Metrics

Evaluating the quality of synthesized images should ideally take into account both the quantitative aspect (per pixel synthesis error) as well as qualitative differences mimicking human perception. In order to cover this spectrum, we report results using three metrics, namely mean squared error (MSE), peak signal-to-noise ratio (PSNR) and structural similarity index metric (SSIM).

The MSE is given as:

$$MSE = \frac{1}{n}\sum_{i=1}^{N}(y_i - y_i^{'})^2, \tag{2.7}$$

where y_i is the original sequence and $y_i^{'}$ is the synthesized version. MSE however depends heavily on the scale of intensities, hence for fair comparison, similar intensity normalization procedures were followed. In this work, we adopt the normalization procedure used in [5]. We report all results except in Section 3.2 after normalizing both ground truth and synthesized image in range [0, 1]. We do this in order to maintain consistency across the study, and allow all future methods to easily compare with our reported values regardless of the standardization/normalization procedure used in network training. We note that the generator successfully learns to synthesize images that lie in the same normalized range as the ground truth and input training images, and hence there is no need for re-normalization after synthesis. Re-normalization in our case was only applied before evaluation to ensure fair comparison for current and future works. For Section 3.2, in order to directly compare with the results reported in [5], we report results without re-normalizing.

In order to still provide a normalization agnostic metric, we report PSNR, which takes into account both the MSE and the largest possible intensity value of the image, given as:

$$PSNR = 10\log_{10}\left(I_{max}^2/\text{MSE}\right), \tag{2.8}$$

where I_{max} is the maximum intensity value that the image supports, which depends on the datatype.

We also report SSIM, which tries to capture the human perceived quality of images by comparing two images. SSIM is given as:

$$\frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}, \tag{2.9}$$

where x, y are two images to be compared, and μ =mean intensity, σ^2 =variance of image, and σ_{xy} =covariance of x, y.

Chapter 3

Results and Discussion

In this section we present the results for our experiments validating our method and comparison with competing unimodal (one-to-one) (Section 3.1), and multi-input single-output (many-to-one) methods (Section 3.2) methods. We also present a validation study (Section 3.3) to investigate quantitative performance improvements by various key components proposed in this work. Finally we present benchmark results in multi-input multi-output (MIMO) synthesis (Section 3.4).

3.1 Single-Input VS Multi-Input Synthesis

In order to understand the performance difference between using a single sequence versus using information from multiple sequences to synthesize a missing sequence, we set up an experiment evaluating the two approaches. Our hypothesis for this experiment is that multiple sequences provide complimentary information about the imaged tissue, and hence should be objectively better than just using one sequence for the task of synthesizing a missing one. We set up an experiment to compare multi-input single-output model with two single-input single-output models in two tasks, namely synthesizing missing T_1 and T_2 sequences respectively. For single-input single-output models, we set up a Pix2Pix [31] model as baseline, called P2P. We also compare with a state-of-art single-input single-output synthesis method called pGAN [28]. pGAN adopts it's generator architecture from [50] and proposes a combination of L1 reconstruction loss and perceptual loss using VGG16 as the loss-network in a generative adversarial network framework. The discriminator for pGAN was adopted from [31]. We use the official pGAN implementation[1] for training and testing on the pGAN (k=3) model. Finally, for our multi-input single-input model, we implement a multi-input single-output variant of our proposed MM-GAN called MI-GAN (multi-input GAN).

[1] https://github.com/icon-lab/pGAN-cGAN

We call the baseline Pix2Pix models P2P$_{T_1}$ (synthesizing T_1 from T_2) and P2P$_{T_2}$ (synthesizing T_2 from T_1). Similarly, pGAN models are named as pGAN$_{T_1}$ and pGAN$_{T_2}$. For our multi-input variants, the two variants of MI-GAN are named as MI-GAN$_{T_1}$ and MI-GAN$_{T_2}$, which synthesize T_1 from (T_2, T_{1c}, T_{2flair}) and T_2 from (T_1, T_{1c}, T_{2flair}) respectively. For P2P and MI-GAN models training was performed for 60 epochs, using consistent set of network hyperparameters used throughout this thesis. For pGAN, training was performed as outlined in the original paper [28] for 100 epochs, with k=3 slices as input to the network. All networks were trained on 70 patients from LGG cohort of BraTS2018 dataset, and tested on 5 patients. Although input normalization between the P2P/MI-GAN and pGAN differ, all metrics were calculated after normalizing both ground truth and synthesized image in range [0, 1]. We also perform Wilcoxon signed-rank tests across all test patients and report p-values wherever the performance difference is statistically significant ($p < 0.05$).

Table 3.1: Comparison between P2P, pGAN and MI-GAN. Values in boldface represent best performance values. Reported values are mean ± std.

Model	**MSE**	**PSNR**	**SSIM**
P2P$_{T_1}$	0.0135±0.0044	22.1168±2.1001	0.8864±0.0180
pGAN$_{T_1}$	0.0107±0.0048	23.8645±2.8851	0.8992±0.0203
MI-GAN$_{T_1}$	**0.0052±0.0026**	**26.6057±1.3801**	**0.9276±0.0118**
P2P$_{T_2}$	0.0050±0.0019	25.0606±1.2020	0.8931±0.0176
pGAN$_{T_2}$	0.0050±0.0033	25.4511±1.6773	0.9008±0.0250
MI-GAN$_{T_2}$	**0.0049±0.0041**	**26.1233±2.6630**	**0.9078±0.0324**

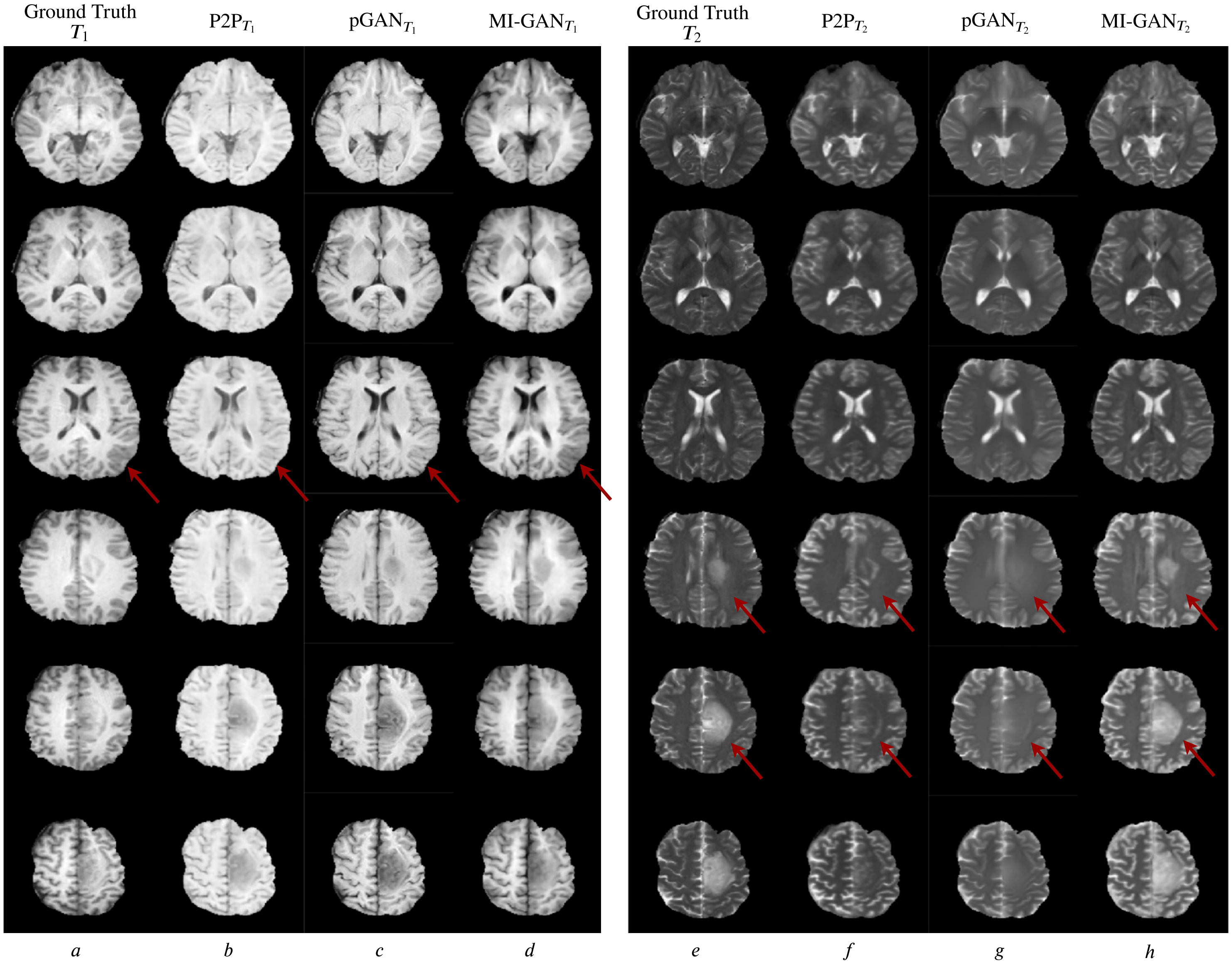

Figure 3.1: Qualitative results for P2P, pGAN, and MI-GAN. Left subfigure (columns (a) through (d)) visualize the performance of three models in synthesis of T_1 sequence. The second subfigure (columns (e) through (h)) show synthesized T_2 sequences for each of the tested models. Red arrows indicate areas where tumor regions were present, and successfully synthesized by MI-GAN in both T_1 and T_2 synthesis.

Table 3.1 presents MSE, PSNR and SSIM results for all three model architectures and their variants. We observe that both variants of MI-GAN outperform both P2P and pGAN models in all metrics.

Comparing MI-GAN$_{T_1}$ and the baseline P2P$_{T_1}$, MI-GAN$_{T_1}$ outperformed by 61.48% in terms of MSE ($p < 0.05$), 20.29% in PSNR ($p < 0.05$) and 4.64% in SSIM ($p < 0.05$). MI-GAN$_{T_1}$ also outperformed the state-of-art single-input single-output model pGAN$_{T_1}$, in all metrics, with improvements of 51.40% in MSE ($p < 0.01$), 11.48% in PSNR ($p < 0.05$) and 3.15% in SSIM ($p < 0.05$).

Similarly for T_2 synthesis, MI-GAN$_{T_2}$ outperforms both P2P$_{T_2}$, pGAN$_{T_2}$. With respect to P2P$_{T_2}$, MI-GAN$_{T_2}$ performs better by 2% in MSE, 4.24% in PSNR and 1.64% in terms of SSIM. Compared to pGAN$_{T_2}$, MI-GAN$_{T_2}$ shows improvement of 2% in MSE, 2.64% in PSNR and 0.77% in SSIM. Although the quantitative improvement in T_2 synthesis was modest compared to both P2P$_{T_2}$ and pGAN$_{T_2}$, we notice interesting qualitative improvements, visible in Figure 3.1, where MI-GAN$_{T_2}$ managed to synthesize the pathology region which other methods failed to generate.

These improvements of MI-GAN over P2P and pGAN models can be attributed to the availability of multiple sequences as input, which the network utilizes to synthesize missing sequences. The qualitative results showing axial slices from a test patient are provided in Figure 3.1 in which red arrow points to the successful synthesis of tumor regions in the case of MI-GAN, which was possible due to tumor specific information present in the available three sequences about the various tumor sub-regions (edema, enhancing and necrotic core) in the input sequences, which is not available in its entirety to the single-input single-output methods. We also notice that MI-GAN performs consistently for a single patient, without showing significant deviation from the ground truth intensity distributions in both T_1 and T_2.

Superior quantitative and qualitative results showing MI-GAN outperforming P2P and pGAN reinforce the hypothesis that using multiple input sequences for the synthesis of a missing sequence is objectively better than using just one input sequence. Moreover, using multi-input methods reduces the number of required models by an order of magnitude, where for a multi-input single-output (many-to-one) only 4 models would be required, compared to 12 for single-input single-output (one-to-one) model ($C(C-1)$ when C = number of sequences = 4). A multi-input multi-output (many-to-many) model which we explore in this work, improves this further by just requiring a single model to perform all possible synthesis tasks for a given C, leading to enormous computational savings during training time.

3.2 T_{2flair} Synthesis (MISO)

In this second set of experiments we train our MM-GAN model to synthesize T_{2flair} sequence in the presence of a varied number of input sequences (one, two or three). Contrasting from the MI-GAN models, this model is trained to generalize on number of different scenarios depending on the available input sequences. In this case, the number of valid scenarios are 7. We perform validation on the ISLES2015 dataset in order to directly compare with REPLICA [22] and MM-Synthesis [5]. The quantitative results are given in Table 3.2. We note that the proposed MM-GAN (0.226±0.046) clearly outperforms REPLICA's unimodal synthesis models (0.271±0.10) in all scenarios, as well as MM-Synthesis (0.236±0.08) in majority (4/7) scenarios. Our method also demonstrates an overall lower MSE standard deviation throughout testing (ranging between [0.03, 0.07], compared to REPLICA [0.08, 0.16] and MM-Synthesis [0.02, 0.13]) in all scenarios but one (T_2 missing). The qualitative results for ISLES2015 are shown in Figure 3.2. Compared to MM-Synthesis (from qualitative results shown in their original paper [5]), our results are objectively sharper, with lower blurring artifacts. MM-GAN also preserves high frequency details of the synthesized sequence, while MM-Synthesis and REPLICA seem to miss most of these details. We request the readers to refer to the original MM-Synthesis [28] manuscript's Figures 5 and 6 for comparison with our proposed MM-GAN's qualitative results given in Figure 3.3 of the current manuscript. Qualitatively from Figure 3.2, MM-GAN follows the intensity distribution of the real T_{2flair} sequence in its synthesized version of T_{2flair}.

Table 3.2: Comparison with unimodal method REPLICA and multimodal method MM-Synthesis. The reported values are mean squared error (MSE). Boldface values represent lowest values of the three methods for a particular scenario.

Scenarios T_1 T_2 DW	**REPLICA**	**MM-Synthesis**	**MM-GAN** (Proposed)
- - ✓	0.278±0.09	0.285±0.13	**0.210±0.057**
- ✓ -	0.374±0.16	0.321±0.12	**0.279±0.055**
- ✓ ✓	0.235±0.08	0.214±0.09	**0.182±0.033**
✓ - -	0.301±0.11	**0.249±0.09**	0.281±0.071
✓ - ✓	0.225±0.08	0.198±0.02	**0.191±0.039**
✓ ✓ -	0.271±0.12	**0.214±0.08**	0.254±0.066
✓ ✓ ✓	0.210±0.08	**0.171±0.06**	0.182±0.041
Mean	0.271±0.10	0.236±0.08	**0.226±0.046**

We found that using CL based learning did not help in MISO experiments, as the presence of more sequences does not necessarily increase the amount of information available. For example, the presence of both T_1 and T_2 does not result in better T_{2flair} synthesis (MSE 0.2541) compared to the presence of DW alone (MSE 0.2109). This is because, for every missing sequence, there tends to be some "highly informative" sequences that, if absent,

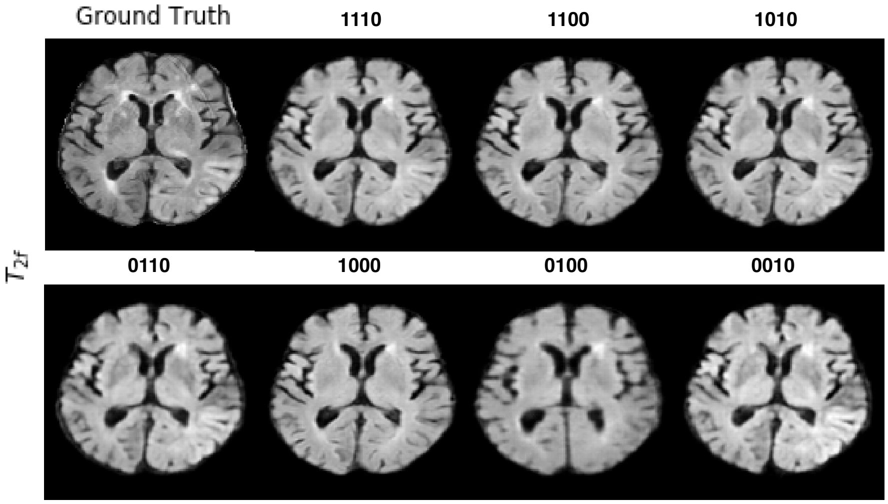

Figure 3.2: Qualitative results from T_{2flair} synthesis experiments with ISLES2015 dataset. The order of scenario bit-string is T_1, T_2, DW, T_{2flair} , where a zero (0) represents missing sequence, while a one (1) represents an available sequence. We note that the presence of multiple sequences (scenario 1110) qualitatively matches the closest to the ground truth, with sharp boundaries between white and grey matter, as well as less blurring artifacts compared to scenarios with two or just one available sequences. Patient images shown here are from VSD ID 70668 from ISLES2015 SISS cohort.

reduces the synthesis performance by a larger margin. On the other hand, the presence of these highly informative sequences can dramatically boost performance, even in cases where no other sequence is present. Due to this, the assumption that leveraging a higher number of present sequences implies an easier case (i.e. more accurate synthesis) does not hold, and thus it becomes problematic to rank scenarios based on how easy they are, which renders CL useless in this case. Globally (for all valid scenarios, presented in next subsection), however, this assumption tends to hold due to the complex nature of interactions between sequences. CL helps tremendously in achieving a stable training of the network in the subsequent experiments (MIMO). For MISO, every scenario was shown to the network with uniform probability, throughout training.

3.3 Validation of Key Components

3.3.1 Imputation Value and Curriculum Learning

In our proposed multi-input multi-output MM-GAN model, the missing sequences during both training and test time has to be imputed with some value. In order to identify the best approach for imputing missing slices, we test MM-GAN with three imputation strategies, where z = average of available sequences, z = noise, and z = zeros.

We also test curriculum learning (CL) based training strategy where the network is initially shown easier examples followed by increasingly difficult examples as the training progresses. We compare this approach with random sampling (RS) method, where scenarios are uniformly sampled during training regardless of the number of missing sequences. In order to test the effectiveness of CL based learning, we report results with CL-based learning as well as random sampling (RS) based learning for each imputation strategy.

The results for our experiments are given in Table 3.3. The imputation value $z =$ zeros performed the best when CL based training strategy is adopted (MSE 0.0095 and PSNR 24.2487), and was slightly outperformed in SSIM by $z =$ zeros with RS sampling (SSIM 0.8955). $z =$ noise performs slightly worse than zero imputation, but better than the $z =$ average imputation for both CL and RS strategies. This may be due to the fact that imputing with a fixed value (zero vs noise/average) provides network with a clear indication about a missing sequence (implicit conditioning), as well as relieves it from learning weights that have to generalize for variable values imputed at test time. Imputing with zeros also removes the non-deterministic behaviour that was prevalent in $z =$ noise, where each test run would yield slightly different quantitative (though mostly similar qualitative) results. For our final benchmark results on LGG and HGG cohorts in BraTS2018, we set $z =$ zeros and use CL based learning strategy to train the networks.

Table 3.3: Quantitative experiments testing three different imputation methods $z \in$ {average, noise, zeros}, as well as two training strategies curriculum learning (CL) and random sampling (RS). The combination $z =$ zeros and curriculum learning based training performs the best in terms of MSE and PSNR, while $z =$ zeros and random sampling performs slightly better in SSIM. Bit strings 0001 to 1110 denote absence (0) or presence (1) of sequences in order T_1, T_2, T_{1c}, T_{2flair} respectively.

z	Strategy	Metrics	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	Mean
average	**CL**	**MSE**	0.0102 ± 0.0041	0.0072 ± 0.0034	0.0043 ± 0.0024	0.0136 ± 0.0049	0.0178 ± 0.0080	0.0088 ± 0.0040	0.0072 ± 0.0040	0.0135 ± 0.0064	0.0107 ± 0.0043	0.0104 ± 0.0039	0.0058 ± 0.0050	0.0148 ± 0.0058	0.0121 ± 0.0031	0.0125 ± 0.0037	0.0106 ± 0.0045
		PSNR	24.3600 ± 3.1202	25.8186 ± 2.8329	26.5299 ± 2.2198	22.3548 ± 3.2600	21.5038 ± 2.6834	24.3666 ± 2.4599	25.3814 ± 2.2801	23.1930 ± 3.6652	23.8125 ± 2.1223	23.7930 ± 2.2222	25.5398 ± 2.7827	21.5824 ± 2.2552	22.5795 ± 1.2743	22.5779 ± 1.3326	23.8138 ± 2.4650
		SSIM	0.8688 ± 0.0707	0.8890 ± 0.0495	0.9136 ± 0.0169	0.8642 ± 0.0655	0.8740 ± 0.0398	0.8982 ± 0.0174	0.9273 ± 0.0128	0.8853 ± 0.0507	0.9007 ± 0.0243	0.8748 ± 0.0186	0.9023 ± 0.0303	0.8844 ± 0.0190	0.9078 ± 0.0236	0.8676 ± 0.0198	0.8899 ± 0.0328
	RS	**MSE**	0.0091 ± 0.0036	0.0071 ± 0.0034	0.0047 ± 0.0028	0.0125 ± 0.0053	0.0153 ± 0.0069	0.0093 ± 0.0044	0.0082 ± 0.0044	0.0134 ± 0.0068	0.0113 ± 0.0047	0.0107 ± 0.0038	0.0061 ± 0.0052	0.0147 ± 0.0054	0.0129 ± 0.0038	0.0124 ± 0.0039	0.0105 ± 0.0046
		PSNR	24.7807 ± 3.0408	25.8981 ± 3.0860	26.4938 ± 2.3945	22.9278 ± 3.3844	22.2024 ± 2.7340	24.1413 ± 2.5214	24.8613 ± 2.4021	23.2776 ± 3.8887	23.7212 ± 2.3412	23.7417 ± 2.2673	25.5211 ± 2.8685	21.5878 ± 2.1707	22.4448 ± 1.2573	22.5763 ± 1.3752	23.8697 ± 2.5523
		SSIM	0.8707 ± 0.0674	0.8892 ± 0.0497	0.9130 ± 0.0194	0.8709 ± 0.0609	0.8806 ± 0.0361	0.8969 ± 0.0193	0.9219 ± 0.0157	0.8835 ± 0.0544	0.8992 ± 0.0269	0.8751 ± 0.0184	0.9027 ± 0.0308	0.8844 ± 0.0186	0.9055 ± 0.0266	0.8691 ± 0.0209	0.8902 ± 0.0332
noise	**CL**	**MSE**	0.0087 ± 0.0039	0.0078 ± 0.0033	0.0042 ± 0.0025	0.0108 ± 0.0049	0.0142 ± 0.0072	0.0092 ± 0.0049	0.0073 ± 0.0040	0.0124 ± 0.0065	0.0114 ± 0.0053	0.0110 ± 0.0036	0.0057 ± 0.0049	0.0153 ± 0.0063	0.0157 ± 0.0022	0.0118 ± 0.0051	0.0104 ± 0.0046
		PSNR	24.9639 ± 3.1463	25.4739 ± 2.8858	26.7597 ± 2.3988	23.9253 ± 3.5588	22.8694 ± 3.0571	24.4041 ± 2.4803	25.3652 ± 2.1933	23.4336 ± 3.7283	23.7107 ± 2.3808	23.5981 ± 2.2295	25.4479 ± 2.7345	21.7608 ± 2.4545	21.6432 ± 1.1993	22.9348 ± 1.6894	24.0208 ± 2.5812
		SSIM	0.8747 ± 0.0647	0.8877 ± 0.0485	0.9133 ± 0.0192	0.8795 ± 0.0587	0.8858 ± 0.0346	0.8984 ± 0.0165	0.9238 ± 0.0119	0.8860 ± 0.0520	0.8997 ± 0.0266	0.8754 ± 0.0167	0.9020 ± 0.0289	0.8831 ± 0.0192	0.9008 ± 0.0262	0.8694 ± 0.0226	0.8914 ± 0.0319
	RS	**MSE**	0.0094 ± 0.0034	0.0087 ± 0.0034	0.0039 ± 0.0019	0.0121 ± 0.0050	0.0141 ± 0.0061	0.0102 ± 0.0040	0.0057 ± 0.0036	0.0138 ± 0.0069	0.0116 ± 0.0040	0.0123 ± 0.0032	0.0059 ± 0.0057	0.0151 ± 0.0053	0.0158 ± 0.0014	0.0129 ± 0.0055	0.0108 ± 0.0042
		PSNR	24.7128 ± 3.1711	25.1588 ± 3.1052	26.7751 ± 2.2401	23.3274 ± 3.5124	22.7595 ± 2.6671	23.9286 ± 2.2215	26.2113 ± 1.8840	23.1377 ± 3.8143	23.5335 ± 1.9258	23.3185 ± 1.9840	25.5491 ± 2.8145	21.4729 ± 1.8973	21.7065 ± 0.8774	22.6359 ± 1.6579	23.8734 ± 2.4123
		SSIM	0.8663 ± 0.0683	0.8838 ± 0.0535	0.9140 ± 0.0159	0.8740 ± 0.0600	0.8831 ± 0.0345	0.8942 ± 0.0144	0.9300 ± 0.0115	0.8855 ± 0.0515	0.8999 ± 0.0220	0.8742 ± 0.0178	0.9023 ± 0.0301	0.8832 ± 0.0167	0.9001 ± 0.0227	0.8691 ± 0.0227	0.8900 ± 0.0315
zeros	**CL**	**MSE**	0.0092 ± 0.0037	0.0072 ± 0.0032	0.0040 ± 0.0022	0.0108 ± 0.0044	0.0129 ± 0.0054	0.0084 ± 0.0036	0.0061 ± 0.0035	0.0120 ± 0.0063	0.0109 ± 0.0040	0.0102 ± 0.0030	0.0057 ± 0.0046	0.0128 ± 0.0048	0.0120 ± 0.0025	0.0113 ± 0.0040	**0.0095 ± 0.0039**
		PSNR	24.7832 ± 3.3197	25.7012 ± 2.9797	26.9985 ± 2.4075	23.6527 ± 3.4296	22.9238 ± 2.6554	24.7581 ± 2.2131	25.9841 ± 2.1926	23.6018 ± 3.8153	23.8408 ± 2.1715	23.9202 ± 2.0746	25.6005 ± 2.7909	22.1330 ± 1.8389	22.4980 ± 1.1684	23.0852 ± 1.5142	**24.2487 ± 2.4694**
		SSIM	0.8758 ± 0.0651	0.8925 ± 0.0486	0.9179 ± 0.0178	0.8811 ± 0.0560	0.8873 ± 0.0337	0.8984 ± 0.0144	0.9288 ± 0.0137	0.8908 ± 0.0509	0.9028 ± 0.0244	0.8792 ± 0.0178	0.9030 ± 0.0303	0.8885 ± 0.0164	0.9086 ± 0.0253	0.8692 ± 0.0224	0.8946 ± 0.0312
	RS	**MSE**	0.0083 ± 0.0034	0.0078 ± 0.0025	0.0039 ± 0.0021	0.0114 ± 0.0047	0.0141 ± 0.0070	0.0094 ± 0.0046	0.0070 ± 0.0037	0.0115 ± 0.0053	0.0104 ± 0.0044	0.0102 ± 0.0029	0.0051 ± 0.0035	0.0142 ± 0.0052	0.0136 ± 0.0037	0.0122 ± 0.0043	0.0099 ± 0.0041
		PSNR	25.1025 ± 2.9154	25.5322 ± 2.7620	27.0696 ± 2.2400	23.4269 ± 3.2486	22.7479 ± 2.9075	24.2726 ± 2.3542	25.4209 ± 2.1961	23.6656 ± 3.5530	23.9555 ± 2.1310	23.9078 ± 1.9425	25.7566 ± 2.3658	21.7955 ± 2.1545	22.0829 ± 1.4224	22.6741 ± 1.5116	24.1008 ± 2.4075
		SSIM	0.8814 ± 0.0608	0.8925 ± 0.0456	0.9193 ± 0.0173	0.8850 ± 0.0546	0.8879 ± 0.0352	0.9010 ± 0.0157	0.9264 ± 0.0125	0.8909 ± 0.0498	0.9042 ± 0.0256	0.8803 ± 0.0155	0.9050 ± 0.0281	0.8873 ± 0.0179	0.9063 ± 0.0258	0.8699 ± 0.0237	**0.8955 ± 0.0306**

3.3.2 Effect of Implicit Conditioning (IC)

We proposed to use a combination of zero imputation, selective loss computation in $\mathcal{G}$, and selective discrimination in $\mathcal{D}$, collectively referred to as implicit conditioning (IC) in order to better train the network to learn to synthesize only the missing sequences, without penalizing it for inaccurate synthesis of other available sequences. In this set of experiments we test the effectiveness of IC based training by training an MM-GAN model on the LGG cohort, with and without IC enabled. In the model with IC disabled, we penalize the generator for inaccurately synthesizing all sequences (C = 4), as well as let the discriminator take the whole $\mathcal{G}$ output ($G(X_z)$) as input, instead of imputing real sequences in them (X_i). We keep the zero imputation part intact due to the network architectural constraints.

The results are shown in Table 3.4. We observe that training a model with implicit conditioning turned on outperforms the one which is trained without it in all three metrics. IC-based trained model achieves a MSE of 0.0095, compared to 0.0100 for non-IC model. Similarly, IC-based trained model outperforms in PSNR (24.2487 vs 23.6626), and in SSIM (0.8946 vs 0.8796) ($p < 0.01$). This performance improvement can be attributed to the fact that the generator has to work harder in case where IC is not used, since it has to synthesize all the sequence with high accuracy in order to achieve a lower loss value. This leads to a trade-off as the limited capacity of the generator would have to synthesize all input sequences, instead of a subset of the input which is the case in IC-based training. Moreover, selective discrimination in IC helps the generator by making it harder for the discriminator to discriminate between real and fake sequences by imputing real sequences in its input. This prevents the discriminator to overpower the generator, thereby leading to a stable training process with an even learning ground. This phenomenon was not observed while training MM-GAN without IC, in which without the selective discrimination in $\mathcal{D}$, it quickly overpowered the generator by easily detecting fakes and lead to overall poorer performance of the generator which was exacerbated by lack of selective loss computation in $\mathcal{G}$. In the final benchmark presented in subsection 3.4 of the thesis, we train networks with implicit conditioning enabled.

Table 3.4: Quantitative results for evaluating effectiveness of implicit conditioning (IC) based training strategy.

Scenario				**Strategies**					
				No-IC			**IC**		
T_1	T_2	T_{1c}	T_{2f}	**MSE**	**PSNR**	**SSIM**	**MSE**	**PSNR**	**SSIM**
-	-	-	✓	0.0120±0.0050	22.3268±3.0867	0.8282±0.0812	0.0092±0.0037	24.7832±3.3197	0.8758±0.0651
-	-	✓	-	0.0086±0.0033	23.2450±2.3221	0.8456±0.0646	0.0072±0.0032	25.7012±2.9797	0.8925±0.0486
-	-	✓	✓	0.0036±0.0016	26.8574±1.9684	0.9092±0.0168	0.0040±0.0022	26.9985±2.4075	0.9179±0.0178
-	✓	-	-	0.0089±0.0027	22.3974±2.3134	0.8294±0.0716	0.0108±0.0044	23.6527±3.4296	0.8811±0.0560
-	✓	-	✓	0.0152±0.0051	22.2603±2.4174	0.8749±0.0337	0.0129±0.0054	22.9238±2.6554	0.8873±0.0337
-	✓	✓	-	0.0076±0.0030	24.4887±2.2431	0.8912±0.0167	0.0084±0.0036	24.7581±2.2131	0.8984±0.0144
-	✓	✓	✓	0.0054±0.0034	26.3621±1.8637	0.9304±0.0080	0.0061±0.0035	25.9841±2.1926	0.9288±0.0137
✓	-	-	-	0.0116±0.0058	23.7413±3.6682	0.8858±0.0534	0.0120±0.0063	23.6018±3.8153	0.8908±0.0509
✓	-	-	✓	0.0119±0.0034	23.5427±1.8551	0.8999±0.0223	0.0109±0.0040	23.8408±2.1715	0.9028±0.0244
✓	-	✓	-	0.0090±0.0021	24.0322±1.6683	0.8698±0.0208	0.0102±0.0030	23.9202±2.0746	0.8792±0.0178
✓	-	✓	✓	0.0050±0.0036	25.7482±2.2597	0.9013±0.0249	0.0057±0.0046	25.6005±2.7909	0.9030±0.0303
✓	✓	-	-	0.0133±0.0036	21.9508±1.5660	0.8855±0.0162	0.0128±0.0048	22.1330±1.8389	0.8885±0.0164
✓	✓	-	✓	0.0188±0.0035	20.8677±1.3094	0.8957±0.0214	0.0120±0.0025	22.4980±1.1684	0.9086±0.0253
✓	✓	✓	-	0.0094±0.0030	23.4557±1.3986	0.8671±0.0198	0.0113±0.0040	23.0852±1.5142	0.8692±0.0224
mean±std				0.0100±0.0035	23.6626±2.1386	0.8796±0.0337	**0.0095±0.0039**	**24.2487±2.4694**	**0.8946±0.0312**

3.3.3 Reconstruction Error in Other Planes

Due to the design of our proposed method in which 2D axial slices are used in training/testing, we ran another experiment in order to quantify whether any inconsistency in terms of reconstruction error (mean squared error) exists in other planes (sagittal and coronal). We calculated reconstruction errors for 5 test patients in LGG cohort for all three planes (axial, sagittal, and coronal). Then, we compared the error distributions from each plane using the Mann-Whitney U statistical test, which tests statistical significance between two unpaired sample distributions without the assumption that they are originally sampled from the Normal distribution. The null hypothesis for this test is chosen as follows: if an error value is randomly chosen from the first sample distribution, the value is equally likely to be either greater than or less than another random value chosen from the second sample distribution. In our case, we perform two tests; between axial and sagittal planes, and between axial and coronal planes. We maintain our choice of confidence threshold of 0.05 in this test as well.

We report the results of the test in Table 3.5. We observe that out of the 10 observed p-values, all but one were significantly higher than our chosen confidence threshold of 0.05. Through these tests, we confirmed that the null hypothesis cannot be rejected due to high p-values in both the tests (axial versus sagittal and axial versus coronal), for all tested patients. Hence the test strongly suggests that there are no significant differences or inconsistency between reconstruction error distributions in different planes.

Table 3.5: Results for Mann-Whitney U statistical test on 5 test patients from LGG cohort. *p*-values from two tests (axial vs coronal and axial vs sagittal) are reported.

Patient Name	**p-values**	
	Coronal	**Sagittal**
Brats18_2013_9_1	0.2814	0.2714
Brats18_TCIA10_449_1	0.4980	0.4670
Brats18_TCIA09_451_1	0.4848	0.3592
Brats18_TCIA12_470_1	0.4353	0.0730
Brats18_TCIA12_466_1	0.3061	0.4213

3.4 Multimodal Synthesis (MIMO)

We present results for our experiments on BRaTS2018's HGG and LGG cohorts in Table 3.6 and 3.7. We set $z =$ zeros for imputation, and train the networks with implicit conditioning (IC) and curriculum learning (CL). In this experiment we train our proposed MM-GAN model on all 14 valid scenarios, in order to synthesize any missing sequence from any number or combination of available sequences. We observe that the proposed MM-GAN model performs consistently well when synthesizing just one sequence, with high overall SSIM (>0.90 in most cases except one in LGG), PSNR (>22.0) values, and low MSE (<0.015). As more sequences start missing, the task of synthesizing missing sequences gets harder.

During the initial epochs of training, MM-GAN tends to learn the general global structure of the brain, without considering the local level details. This seems to be enough for the generator to fool the discriminator initially. However, as the training progresses and the discriminator becomes stronger, the generator is forced to learn the local features of the slice, which includes small details, especially the boundaries between the grey and white matter visible in the sequence. The qualitative results shown in Figure 3.3 show how MM-GAN effectively synthesizes the missing sequence in various scenarios, while preserving high frequency details that delineate between grey and white matter of the brain, as well as recreating the tumor region in the frontal lobe by combining information from available sequences. The synthesis of the tumor in the final images depend heavily on the available sequences. For example, the contrast sequence T_{1c} provides clear delineation of enhancing ring-like region around the necrotic mass, which is an important indicator of the size of the tumor.

Table 3.6: Performance on BraTS2018 High Grade Glioma (HGG) Cohort

Scenarios						
T_1	T_2	T_{1c}	T_{2f}	**MSE**	**PSNR**	**SSIM**
-	-	-	✓	0.0143±0.0086	23.196±4.2908	0.8973±0.0668
-	-	✓	-	0.0072±0.0065	24.524±4.0671	0.8984±0.0726
-	-	✓	✓	0.0060±0.0061	25.863±3.2218	0.9166±0.0339
-	✓	-	-	0.0102±0.0065	23.469±4.1744	0.9074±0.0680
-	✓	-	✓	0.0136±0.0048	22.900±2.1989	0.9156±0.0260
-	✓	✓	-	0.0073±0.0070	24.792±2.9524	0.9140±0.0311
-	✓	✓	✓	0.0091±0.0053	24.173±3.2754	0.9228±0.0190
✓	-	-	-	0.0072±0.0056	24.879±3.8216	0.9091±0.0651
✓	-	-	✓	0.0073±0.0041	26.189±2.1337	0.9264±0.0328
✓	-	✓	-	0.0040±0.0032	26.150±1.8470	0.9107±0.0275
✓	-	✓	✓	0.0017±0.0026	28.678±2.3290	0.9349±0.0262
✓	✓	-	-	0.0068±0.0041	25.242±2.0339	0.9175±0.0275
✓	✓	-	✓	0.0098±0.0066	24.372±2.2792	0.9239±0.0375
✓	✓	✓	-	0.0033±0.0040	26.397±1.9733	0.9150±0.0275
mean±std				0.0082±0.0054	24.789±2.8999	0.9120±0.0401

Table 3.7: Performance on BraTS2018 Low Grade Glioma (LGG) Cohort

Scenarios						
T_1	T_2	T_{1c}	T_{2f}	**MSE**	**PSNR**	**SSIM**
-	-	-	✓	0.0092±0.0037	24.7832±3.3197	0.8758±0.0651
-	-	✓	-	0.0072±0.0032	25.7012±2.9797	0.8925±0.0486
-	-	✓	✓	0.0040±0.0022	26.9985±2.4075	0.9179±0.0178
-	✓	-	-	0.0108±0.0044	23.6527±3.4296	0.8811±0.0560
-	✓	-	✓	0.0129±0.0054	22.9238±2.6554	0.8873±0.0337
-	✓	✓	-	0.0084±0.0036	24.7581±2.2131	0.8984±0.0144
-	✓	✓	✓	0.0061±0.0035	25.9841±2.1926	0.9288±0.0137
✓	-	-	-	0.0120±0.0063	23.6018±3.8153	0.8908±0.0509
✓	-	-	✓	0.0109±0.0040	23.8408±2.1715	0.9028±0.0244
✓	-	✓	-	0.0102±0.0030	23.9202±2.0746	0.8792±0.0178
✓	-	✓	✓	0.0057±0.0046	25.6005±2.7909	0.9030±0.0303
✓	✓	-	-	0.0128±0.0048	22.1330±1.8389	0.8885±0.0164
✓	✓	-	✓	0.0120±0.0025	22.4980±1.1684	0.9086±0.0253
✓	✓	✓	-	0.0113±0.0040	23.0852±1.5142	0.8692±0.0224
mean±std				0.0095±0.0039	24.2487±2.4694	0.8946±0.0312

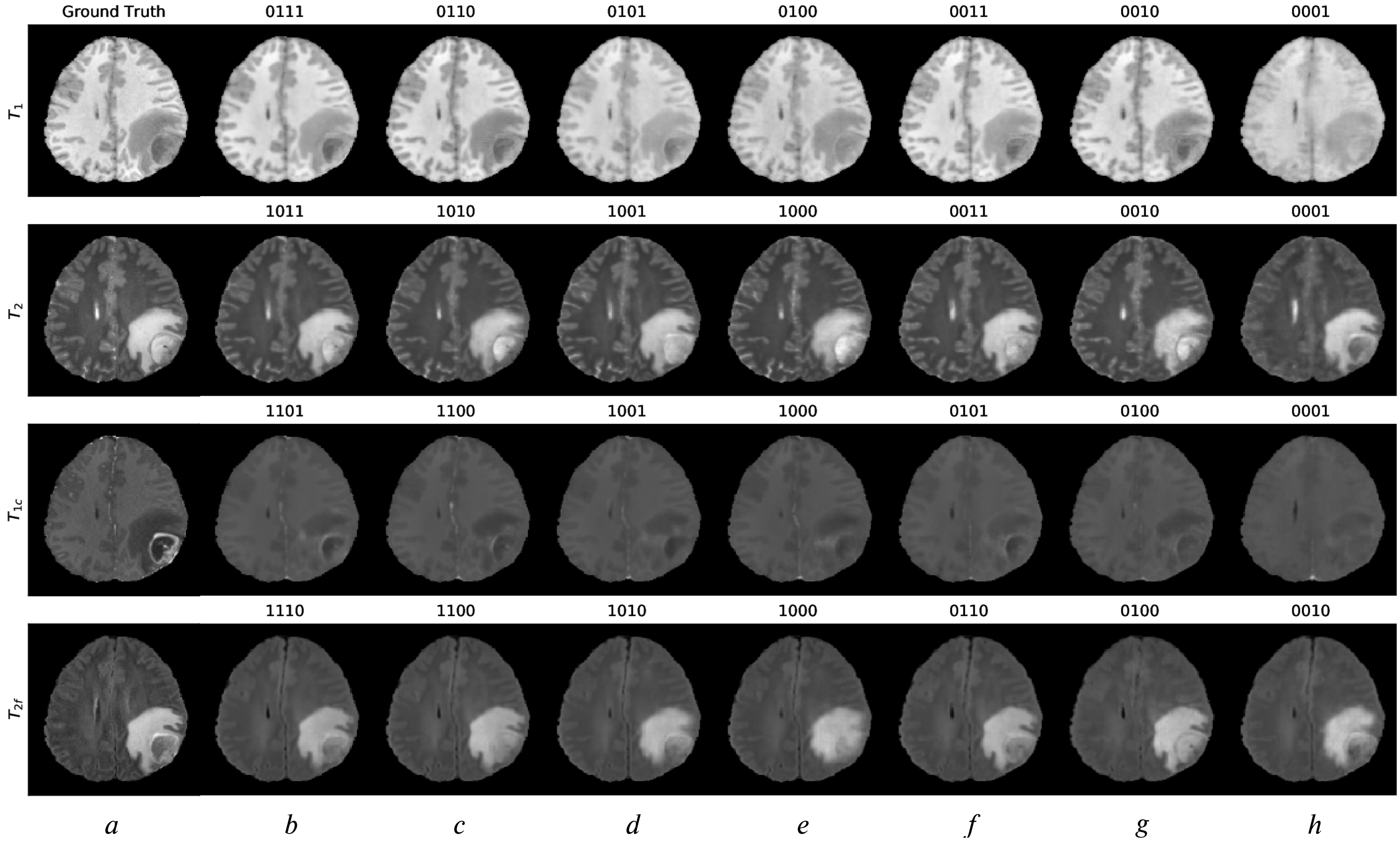

Figure 3.3: Qualitative results from the multimodal synthesis experiments with BRaTS2018 dataset. Each row corresponds to a particular sequence (row names on the left in order T_1, T_2, T_{1c} and T_{2flair}). Columns are indexed at the bottom of the figure by alphabets (a) through (h), and have a column name written on top of each slice. Column names are 4-bit strings where a zero (0) represents missing sequence that was synthesized, and one (1) represents presence of sequence. Column (a) of each row shows the ground truth slice, and the subsequent columns ((b) through (h)) show synthesized versions of that slice in different scenarios. The order of scenario bit-string is T_1, T_2, T_{1c}, T_{2flair}. For instance, the string 0011 indicates that sequences T_1 and T_2 were synthesized from T_{1c} and T_{2flair} sequences. Patient images shown here are from *Brats18_CBICA_AAP_1* from HGG cohort.

Presence of T_1 and/or T_{2flair} sequence leads to improved synthesis of edema features. The contrast sequence T_{1c} provides unique information about the enhancing region around the tumor, which is usually not visible in any other sequence.

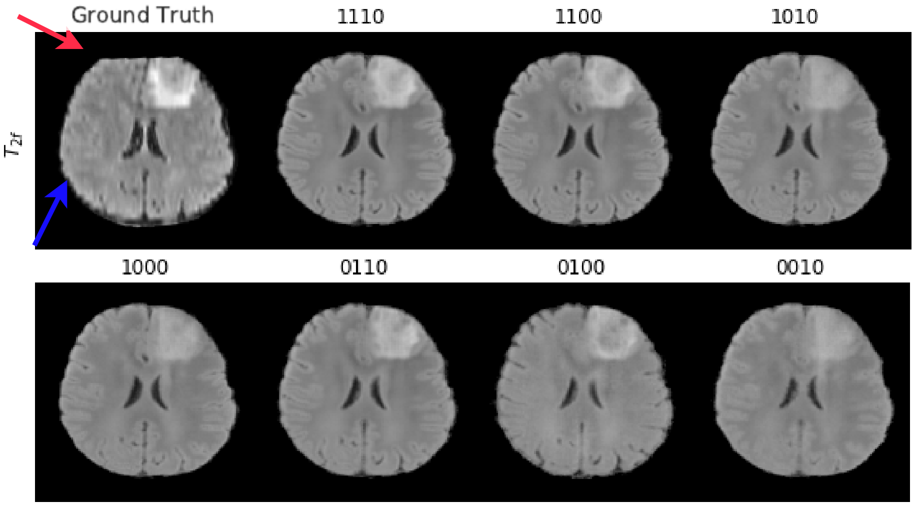

Figure 3.4: Illustration of MM-GAN filling up parts in the scan that are originally missing in the ground truth. Red arrow shows the top part of brain that is missing, and is synthesized in all scenarios. Blue arrow highlights the high-frequency details that are missing in ground truth, but are synthesized in most images. We also notice that when all sequences are present, the T_{2flair} synthesis is the most accurate with respect to the pathology visible in the frontal cortex. The order of scenario bit-string is T_1, T_2, T_{1c}, T_{2flair}. Patient images shown here are from *Brats18_2013_9_1* from LGG cohort.

As shown in Figure 3.4, we also observe that the method fills up lost details as can be seen in T_{2flair} sequence. The original ground truth sequence has the frontal lobe part cut off, probably due to patient movement or miss-registration. However MM-GAN recreates that part by using information from the available sequences. Another interesting side-effect of our approach is visible in T_{2flair} synthesis, where the synthesized versions of T_{2flair} exhibit higher quality details (Figure 3.4) than the original sequence, which was acquired in a very low resolution. This effect is the consequence of the method using high-resolution input sequences (all sequences except T_{2flair} are acquired at higher resolution) to synthesize the missing T_{2flair} sequence. This also suggests that our method may be used for improving or upscaling resolution of available sequences. However we do not investigate this further here, and leave it as future work.

We observe that the generators perform really well when they are constrained using a non-linear activation function at the final layer. However in the case of MISO, the limitation of the normalization type (dividing by mean value of sequence) used in MM-Synthesis prevents us from using a non-linear activation at the end of generator. This is due to the fact that some patients' data in SISS cohort from ISLES2015 contain negative intensity values, which after normalization stay negative. It can be seen that the MSE values reported

Table 3.8: MM-GAN performance variation with respect to number of sequences missing for HGG and LGG cohort.

Dataset	Missing	MSE	PSNR	SSIM
HGG	1	0.0539 ± 0.0215	29.1162 ± 0.9716	0.9268 ± 0.0115
	2	0.0602 ± 0.0170	28.9064 ± 0.8095	0.9223 ± 0.0090
	3	0.0752 ± 0.0125	28.0801 ± 0.4110	0.9087 ± 0.0071
LGG	1	0.1296 ± 0.0485	26.1362 ± 2.8872	0.9080 ± 0.0383
	2	0.1499 ± 0.0208	25.5641 ± 1.3161	0.8976 ± 0.0157
	3	0.1914 ± 0.0358	24.8987 ± 0.8363	0.8732 ± 0.0205

in Table 3.2 tend to be higher than the ones reported in Tables 3.6 and 3.7, due to the latter set of experiments using ReLU activation at the end of generator.

Although MM-GAN observes different scenarios and hence different fake sequences in each iteration, which may affect stability during training, we did not observe any unstable behaviour during the training process. The use of implicit conditioning (IC) assisted in ensuring stable training of networks by making the task challenging for the discriminator, preventing it from overpowering the generator, which in turn lead to the generator converging to a good local minima.

We also observe that the proposed method shows graceful degradation as the number of sequences missing start increasing, which is apparent both qualitative and quantitatively in Figure 3.3 and Table 3.8. For instance, in HGG experiments, compared to having one sequence missing, the performance of MM-GAN drops on average by 27.1%, 2.7% and 0.7% in MSE, PSNR and SSIM respectively for scenarios where two sequences are missing. For scenarios where three sequences are missing, the performance drops on average by 39.1%, 7.2% and 2.2% in terms of MSE, PSNR and SSIM respectively compared to one sequence missing, and 29.3%, 4.6% and 1.5% when compared to scenarios where two sequences are missing. We observe that the method holds up well in generating sequences with high fidelity in terms of PSNR and SSIM even in harder scenarios where multiple sequences may be missing.

Qualitatively, we also investigated the question as to which sequences are the most valuable for the synthesis for each of the four sequences in BraTS2018 HGG cohort. For every sequence that is synthesized, we list a ranking based upon our investigation of the results for each of the remaining sequences. For synthesizing T_{2flair}, we found that T_{1c} sequence, followed by T_2 and T_1 sequences were important. This is also apparent in Figure 3.3, where the removal of T_{1c} in column (c) lead to the synthesized sequence missing necrotic part of tumor completely, while the removal of T_1 (columns (b) and (f)) and T_2 (columns (b) and (d)) did not affect the performance dramatically. For the synthesis of T_{1c}, we found that T_2 sequence held the highest significance, followed by T_1 and T_{2flair} (comparing columns (b) with (d), (f), (c)). This is also evident from row 3 column (d) in Figure 3, when removal of T_2 sequence lead to increased blurring artifacts in the synthesized version of T_{1c}, which were not as pronounced when T_1 or T_{2flair} were removed. For T_2 synthesis, we found that T_{1c} sequence contributed the most towards accurate synthesis (comparing columns (b) and (d)), with T_1 sequence also playing an important role (columns (b) and (f)), lastly followed by T_{2flair}. Finally for T_1 synthesis, we found that T_{1c} was the most important sequence (columns (b) and (d)) enabling accurate synthesis, followed closely by T_2 (columns (b) and (f)) and T_{2flair} (columns (b) and (c)).

Due to MM-GAN being a single unified model (MM-GAN), it relieves the end-user from the difficult task of choosing the right model for synthesis during inference time. For instance, in the case where sequences (T_1,T_2,T_{1c}) are present, and T_{2flair} to be synthesized, a single-input single-output method would have three networks capable of synthesizing T_{2flair} from T_1, T_2 and T_{1c} respectively. The decision as to which network should be chosen for this problem is hard, since each unimodal network would provide trade-offs in terms synthesis quality, especially in tumorous areas where individual sequences do not provide full information. This decision problem is mitigated in multi-input models (MM-Synthesis and MI-GAN), but there still exists the computational overhead during training time in order to train multiple models for each output sequence (total 4 for both MM-Synthesis and MI-GAN). MM-GAN on the other hand, is completely multimodal and only requires training for just one model, which provide computational savings during training time by eliminating the need for training multiple models (if number of sequences C=4, then 12 models in case of unimodal, 4 models in case of multi-input multi-output architectures).

Chapter 4

Conclusions and Future Work

We propose a multi-modal generative adversarial network (MM-GAN) capable of synthesizing missing MR pulse sequences using combined information from the available sequences. Most approaches so far in this domain had been either unimodal, or partially multi-modal (multi-input, single-output). We present a truly multi-modal method that is multi-input and multi-output, generalizing to any combination of available and missing sequences. The synthesis process runs in a single forward pass of the network regardless of the number of sequences missing, and the run time is constant w.r.t number of missing sequences.

The first variant of our proposed MM-GAN, called MI-GAN outperformed the unimodal version pGAN in all three metrics (Table 3.1). We also show that MM-GAN outperforms the best multimodal synthesis method REPLICA [22], as well as MM-Synthesis [5] in multi-input single-input synthesis of T_{2flair} sequence (Table 3.2), and produces objectively sharper and more accurate results. In another set of experiments, we train our method on BraTS2018 dataset to set up a new benchmark in terms of MSE, PSNR and SSIM (Tables 3.6 and 3.7), and show qualitative results for the same (Figure 3.3). MM-GAN performance degrades as a function of number of missing sequences in Table 3.8 but exhibits robustness in maintaining high PSNR and SSIM values even in harder scenarios. We also investigate the performance improvement imparted by our proposed components (imputation type, implicit conditioning, use of curriculum learning), and show that they outperform models trained with respective baseline approaches (Tables 3.3, 3.4). Finally, we show that our method is capable of filling in details missing from the original ground truth sequences, and also capable of improving quality of the synthesized sequences (Figure 3.4).

Although our approach qualitatively and quantitatively performs better than all other competing methods, we note that it has problems in synthesizing the enhancing subregion in T_{1c} sequence properly. This, however, is expected since T_{1c} sequence contains highly specific information about the enhancing region of the tumor that is not present in any other sequences. An inherent limitation of all synthesis methods stems from the fact that MR sequences provide both redundant and unique information. This creates challenges for all synthesis methods, unimodal and multimodal alike. Unimodal methods provide a one-to-one mapping between sequences, but each such model (12 total for 4 sequences) would raise tradeoffs between the synthesis accuracy. For instance, in the experiments given in subsection 3.2, we found that synthesizing T_{2flair} from DWI tends to be more accurate (MSE 0.2109) than synthesizing from T_1 (MSE 0.2813) or T_2 (MSE 0.2799). This reinforces the fact that there are some inherent characteristics to each sequence, which can only be faithfully synthesized if another sequence that more or less captures similar characteristics is present. The sequences provide complementary visual information for a human reader, though there are underlying correlations imperceptible to the naked eye, since they all originate due to common underlying physics and from the same subject. Multi-input methods like ours can exploit the correlations between available sequences, and synthesize the missing sequence by leveraging information from all input sequences. This is evident from the quantitative results in Tables 3.2, 3.6, 3.7, 3.8, and summarized in Table 3.8 where more available sequences allow better synthesis of missing ones. For future work, we note that the inherent design of our method is 2D, and an extension of the work which can take either 2.5D or 3D images into account may perform better both quantitatively and qualitatively.

Another area of investigation would be to explore the up-scaling capabilities of the MM-GAN, where given a low-quality ground truth scan with missing scan areas, the method can generate a higher quality version with filled in missing details. It would also be interesting to test MM-GAN by deploying it as part of the pipeline for downstream analysis, for example segmentation. This natural placement in the pipeline would allow the downstream methods to become robust to missing pulse sequences. Compared to HeMIS [3] and PIMMS [4], this can be another approach to make segmentation algorithms robust to missing pulse sequences.

Bibliography

[1] E. F. Jackson, L. E. Ginsberg, D. F. Schomer, and N. E. Leeds, “A review of MRI pulse sequences and techniques in neuroimaging,” *Surgical Neurology*, vol. 47, pp. 185–199, Feb 1997.

[2] C. Bowles, C. Qin, C. Ledig, R. Guerrero, R. Gunn, A. Hammers, E. Sakka, D. A. Dickie, M. V. Hernandez, N. Royle, J. Wardlaw, H. Rhodius-Meester, B. Tijms, A. W. Lemstra, W. van der Flier, F. Barkhof, P. Scheltens, and D. Rueckert, “Pseudo-healthy image synthesis for white matter lesion segmentation,” in *Simulation and Synthesis in Medical Imaging*, pp. 87–96, Springer International Publishing, Oct 2016.

[3] M. Havaei, N. Guizard, N. Chapados, and Y. Bengio, “HeMIS: Hetero-Modal Image Segmentation,” in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016*, pp. 469–477, Springer International Publishing, 2016.

[4] T. Varsavsky, Z. Eaton-Rosen, C. H. Sudre, P. Nachev, and M. J. Cardoso, “PIMMS: Permutation Invariant Multi-modal Segmentation,” in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pp. 201–209, Springer International Publishing, 2018.

[5] A. Chartsias, T. Joyce, M. V. Giuffrida, and S. A. Tsaftaris, “Multimodal MR Synthesis via Modality-Invariant Latent Representation,” *IEEE Transactions on Medical Imaging*, vol. 37, pp. 803–814, Mar 2018.

[6] H. Chen, Q. Dou, L. Yu, J. Qin, and P.-A. Heng, “VoxResNet: Deep voxelwise residual networks for brain segmentation from 3D MR images,” *NeuroImage*, vol. 170, pp. 446–455, 2018.

[7] S. Bakas, B. Menze, C. Davatzikos, M. Reyes, and K. Farahani, “2017 International MICCAI BraTS Challenge,” 2017.

[8] F. Isensee, P. Kickingereder, W. Wick, M. Bendszus, and K. H. Maier-hein, “Brain Tumor Segmentation and Radiomics Survival Prediction : Contribution to the BRATS 2017 Challenge,” in *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*, pp. 287–297, Springer International Publishing, 2018.

[9] G. Wang, W. Li, S. Ourselin, and T. Vercauteren, “Automatic brain tumor segmentation using cascaded anisotropic convolutional neural networks,” in *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*, pp. 178–190, Springer International Publishing, 2018.

[10] J. Dolz, K. Gopinath, J. Yuan, H. Lombaert, C. Desrosiers, and I. B. Ayed, "HyperDense-Net: A hyper-densely connected CNN for multi-modal image segmentation," *IEEE Transactions on Medical Imaging*, Apr 2018.

[11] S. Pereira, A. Pinto, V. Alves, and C. A. Silva, "Brain tumor segmentation using convolutional neural networks in MRI images," *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1240–1251, 2016.

[12] M. Havaei, A. Davy, D. Warde-Farley, A. Biard, A. Courville, Y. Bengio, C. Pal, P.-M. Jodoin, and H. Larochelle, "Brain tumor segmentation with Deep Neural Networks," *Medical Image Analysis*, vol. 35, pp. 18–31, Jan 2017.

[13] J. E. Iglesias, E. Konukoglu, D. Zikic, B. Glocker, K. Van Leemput, and B. Fischl, "Is synthesizing MRI contrast useful for inter-modality analysis?," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 631–638, Springer, 2013.

[14] D. H. Ye, D. Zikic, B. Glocker, A. Criminisi, and E. Konukoglu, "Modality propagation: coherent synthesis of subject-specific scans with data-driven regularization," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 606–613, Springer, 2013.

[15] A. Jog, S. Roy, A. Carass, and J. L. Prince, "Magnetic resonance image synthesis through patch regression," in *Proceedings - International Symposium on Biomedical Imaging*, vol. 2013, pp. 350–353, NIH Public Access, Dec 2013.

[16] A. Jog, A. Carass, D. L. Pham, and J. L. Prince, "Random forest FLAIR reconstruction from T1, T2, and Pd-weighted MRI," in *Proceedings - International Symposium on Biomedical Imaging*, pp. 1079–1082, IEEE, 2014.

[17] G. van Tulder and M. de Bruijne, "Why does synthesized data improve multi-sequence classification?," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 531–538, Springer, 2015.

[18] A. Jog, A. Carass, D. L. Pham, and J. L. Prince, "Tree-encoded conditional random fields for image synthesis," in *International Conference on Information Processing in Medical Imaging*, pp. 733–745, Springer, 2015.

[19] S. Roy, Y.-Y. Chou, A. Jog, J. A. Butman, and D. L. Pham, "Patch based synthesis of whole head MR images: Application to EPI distortion correction," in *International Workshop on Simulation and Synthesis in Medical Imaging*, pp. 146–156, Springer, 2016.

[20] Y. Huang, L. Beltrachini, L. Shao, and A. F. Frangi, "Geometry regularized joint dictionary learning for cross-modality image synthesis in magnetic resonance imaging," in *International Workshop on Simulation and Synthesis in Medical Imaging*, pp. 118–126, Springer, 2016.

[21] V. Sevetlidis, M. V. Giuffrida, and S. A. Tsaftaris, "Whole image synthesis using a deep encoder-decoder network," in *International Workshop on Simulation and Synthesis in Medical Imaging*, pp. 127–137, Springer, 2016.

[22] A. Jog, A. Carass, S. Roy, D. L. Pham, and J. L. Prince, "Random forest regression for magnetic resonance image synthesis," *Medical Image Analysis*, vol. 35, pp. 475–488, Jan 2017.

[23] S. Olut, Y. H. Sahin, U. Demir, and G. Unal, "Generative adversarial training for MRA image synthesis using multi-contrast MRI," in *International Workshop on PRedictive Intelligence In MEdicine*, pp. 147–154, Springer, 2018.

[24] R. Mehta and T. Arbel, "RS-Net: Regression-Segmentation 3D CNN for Synthesis of Full Resolution Missing Brain MRI in the Presence of Tumours," in *International Workshop on Simulation and Synthesis in Medical Imaging*, pp. 119–129, Springer, 2018.

[25] M. Salem, S. Valverde, M. Cabezas, D. Pareto, A. Oliver, J. Salvi, A. Rovira, and X. Lladó, "Multiple Sclerosis Lesion Synthesis in MRI using an encoder-decoder U-NET," *IEEE Access*, vol. 7, pp. 25171–25184, 2019.

[26] B. Yu, L. Zhou, L. Wang, Y. Shi, J. Fripp, and P. Bourgeat, "Ea-GANs: Edge-aware Generative Adversarial Networks for Cross-modality MR Image Synthesis," *IEEE Transactions on Medical Imaging*, pp. 1–1, 2019.

[27] H. Van Nguyen, K. Zhou, and R. Vemulapalli, "Cross-domain synthesis of medical images using efficient location-sensitive deep network," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 677–684, Springer, 2015.

[28] S. Ul, H. Dar, M. Yurt, L. Karacan, A. Erdem, E. Erdem, and T. Çukur, "Image Synthesis in Multi-Contrast MRI with Conditional Generative Adversarial Networks," *IEEE Transactions on Medical Imaging*, pp. 1–1, 2019.

[29] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, pp. 2672–2680, 2014.

[30] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[31] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1125–1134, 2017.

[32] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," *Proceedings of The 33rd International Conference on Machine Learning*, pp. 1060–1069, 2016.

[33] L.-C. Yang, S.-Y. Chou, and Y.-H. Yang, "MidiNet: A convolutional generative adversarial network for symbolic-domain music generation," *arXiv preprint arXiv:1703.10847*, 2017.

[34] J. Gauthier, "Conditional generative adversarial nets for convolutional face generation," *Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, Winter semester*, vol. 2014, no. 5, p. 2, 2014.

[35] D. Ulyanov, A. Vedaldi, and V. Lempitsky, “Instance normalization: The missing ingredient for fast stylization,” *arXiv preprint arXiv:1607.08022*, 2016.

[36] M. Mathieu, C. Couprie, and Y. LeCun, “Deep multi-scale video prediction beyond mean square error,” *arXiv preprint arXiv:1511.05440*, 2015.

[37] L. Karacan, Z. Akata, A. Erdem, and E. Erdem, “Learning to generate images of outdoor scenes from attributes and semantic layouts,” *arXiv preprint arXiv:1612.00215*, 2016.

[38] S. Kazeminia, C. Baur, A. Kuijper, B. van Ginneken, N. Navab, S. Albarqouni, and A. Mukhopadhyay, “GANs for medical image analysis,” *arXiv preprint arXiv:1809.06222*, 2018.

[39] P. Moeskops, M. Veta, M. W. Lafarge, K. A. J. Eppenhof, and J. P. W. Pluim, “Adversarial Training and Dilated Convolutions for Brain MRI Segmentation,” in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pp. 56–64, Springer International Publishing, 2017.

[40] Y. Xue, T. Xu, H. Zhang, L. R. Long, and X. Huang, “SegAN: Adversarial network with multi-scale L1 loss for medical image segmentation,” *Neuroinformatics*, vol. 16, no. 3-4, pp. 383–392, 2018.

[41] S. Izadi, Z. Mirikharaji, J. Kawahara, and G. Hamarneh, “Generative adversarial networks to segment skin lesions,” in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, pp. 881–884, IEEE, 2018.

[42] A. Bentaieb and G. Hamarneh, “Adversarial Stain Transfer for Histopathology Image Analysis,” *IEEE Transactions on Medical Imaging*, vol. 37, pp. 792–802, Mar 2018.

[43] C. Tanner, F. Ozdemir, R. Profanter, V. Vishnevsky, E. Konukoglu, and O. Goksel, “Generative adversarial networks for MR-CT deformable image registration,” *arXiv preprint arXiv:1807.07349*, 2018.

[44] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation,” in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, Springer, May 2015.

[45] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, “Curriculum learning,” in *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 41–48, ACM, 2009.

[46] O. Maier, B. H. Menze, J. von der Gablentz, L. Häni, M. P. Heinrich, M. Liebrand, S. Winzeck, A. Basit, P. Bentley, L. Chen, and Others, “ISLES 2015-A public evaluation benchmark for ischemic stroke lesion segmentation from multispectral MRI,” *Medical Image Analysis*, vol. 35, pp. 250–269, 2017.

[47] B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, Y. Burren, N. Porz, J. Slotboom, R. Wiest, L. Lanczi, E. Gerstner, M. A. Weber, T. Arbel, B. B. Avants, N. Ayache, P. Buendia, D. L. Collins, N. Cordier, J. J. Corso, A. Criminisi, T. Das, H. Delingette, Ç. Demiralp, C. R. Durst, M. Dojat, S. Doyle, J. Festa,

F. Forbes, E. Geremia, B. Glocker, P. Golland, X. Guo, A. Hamamci, K. M. Iftekharuddin, R. Jena, N. M. John, E. Konukoglu, D. Lashkari, J. A. Mariz, R. Meier, S. Pereira, D. Precup, S. J. Price, T. R. Raviv, S. M. Reza, M. Ryan, D. Sarikaya, L. Schwartz, H. C. Shin, J. Shotton, C. A. Silva, N. Sousa, N. K. Subbanna, G. Szekely, T. J. Taylor, O. M. Thomas, N. J. Tustison, G. Unal, F. Vasseur, M. Wintermark, D. H. Ye, L. Zhao, B. Zhao, D. Zikic, M. Prastawa, M. Reyes, and K. Van Leemput, "The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS)," *IEEE Transactions on Medical Imaging*, vol. 34, pp. 1993–2024, Oct 2015.

[48] M. Jenkinson, M. Pechaud, S. Smith, and Others, "BET2: MR-based estimation of brain, skull and scalp surfaces," in *Eleventh annual meeting of the organization for human brain mapping*, vol. 17, p. 167, Toronto., 2005.

[49] S. Klein, M. Staring, K. Murphy, M. A. Viergever, and J. P. W. Pluim, "Elastix: a toolbox for intensity-based medical image registration," *IEEE Transactions on Medical Imaging*, vol. 29, no. 1, pp. 196–205, 2010.

[50] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European Conference on Computer Vision*, pp. 694–711, Springer, 2016.

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