UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA IN BIOINGEGNERIA

Time-efficient Diffusion Models for Semantic Segmentation of Choroid Plexus from brain MRI

Relatore
Prof. Castellaro Marco

Laureando
Giupponi Alessandro

ANNO ACCADEMICO 2023-2024

Data di laurea 03/09/2024

Abstract

Since their introduction, diffusion models have achieved significant success in various applications, demonstrating superiority over other generative models. This work explores the feasibility of applying diffusion models to the segmentation of the choroid plexus, a crucial brain structure for the diagnosis and study of various pathologies. Using magnetic resonance imaging (MRI) and manual segmentations performed by expert radiologists on healthy subjects and patients with relapsing-remitting multiple sclerosis (RRMS), several models were trained to generate accurate and consistent segmentations from the input images. In addition to the direct training of Denoising Diffusion Probabilistic Models, advanced deep learning techniques were implemented to enhance performance and make the application of diffusion models in the medical field more feasible. Specifically, transfer learning and self-supervised learning techniques were adopted to assess whether efficient pre-training could reduce the number of images required for model training, given the limited availability of freely usable medical data. Additionally, patch-based techniques were employed to segment complete three-dimensional volumes, aiming to reduce the time needed to generate segmented volumes, which is a crucial factor in the medical context. Our results indicate that, while diffusion models were not initially designed for segmentation tasks, they can be successfully adapted for this purpose. However, further research is necessary to achieve state-of-the-art performance and fully leverage deep learning techniques such as data augmentation, transfer learning, and self-supervised learning. Nevertheless, our findings provide valuable insights for future research, guiding efforts toward improved performance and generalization. In conclusion, among various generative models, diffusion models present a promising alternative to traditional methodologies used in medical segmentation, with potential improvements achievable through the combination with other deep learning techniques and the ability to provide quick results even for three-dimensional segmentations.

Contents

Chapter 1

Introduction

1.1 Diffusion Models

Generative modelling using neural network has been a driving force in the past decade of deep learning, significantly impacting various domains such as images [1], audio [2] and text [3]. From a probabilistic point of view, generative models are designed to synthesize samples $\tilde{x} \sim p_\theta(\tilde{x})$ that come from the same distribution as the training data $x \sim p_d(x)$ [4].
In recent years, advancements in general deep learning architectures as well as in computation power have enabled the training of larger models on more extensive datasets, leading to a resurgence of interest in generative models and unveiling improved visual fidelity and sampling speed. Specifically, generative adversarial networks(GANs) [5], variational autoencoders (VAEs) [6], and normalizing flows (NFs) [7] have emerged.

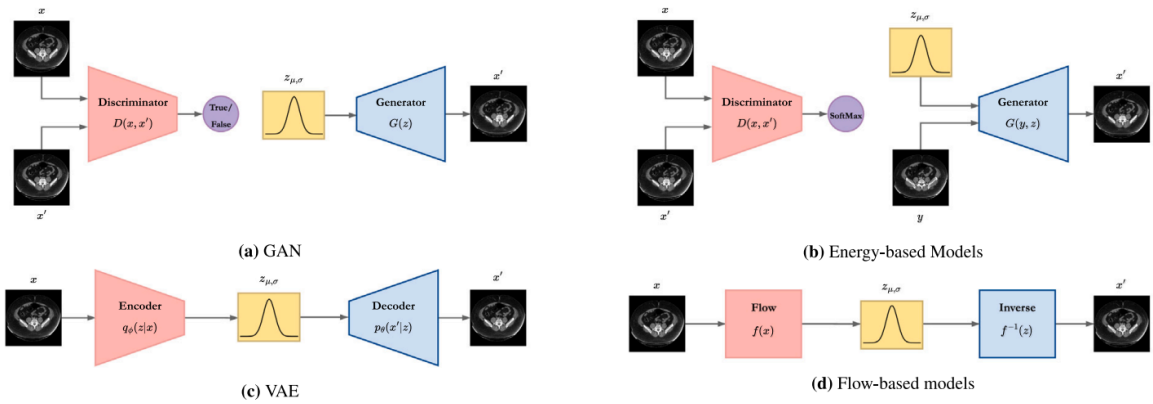

Fig. 1.1. This figure highlights various generative models and their principles: **(a)** GAN trains a generator and discriminator adversarially; **(b)** EBM uses an energy-based function and prior input for generation; **(c)** VAE projects data to a latent space and samples via a decoder; **(d)** NF employs an invertible flow function for transformation and generation. [4]

In addition to these, generative models based on diffusion processes have gained traction. These models do not require aligning posterior distribution, estimating intractable partition functions, introducing of additional discriminator networks or imposing network constraints. To date, diffusion models have proven useful in a wide variety of areas, ranging from generative modelling tasks such as image generation [8], [9], image super-resolution [10], image inpainting [11], [12] to discriminative tasks such as image segmentation [13], classification [14], contrast harmonization [15], and anomaly detection [16]. Moreover, they are versatile, applicable to bi-dimensional, tri-dimensional, and even four-dimensional data, where the fourth dimension refers to time.

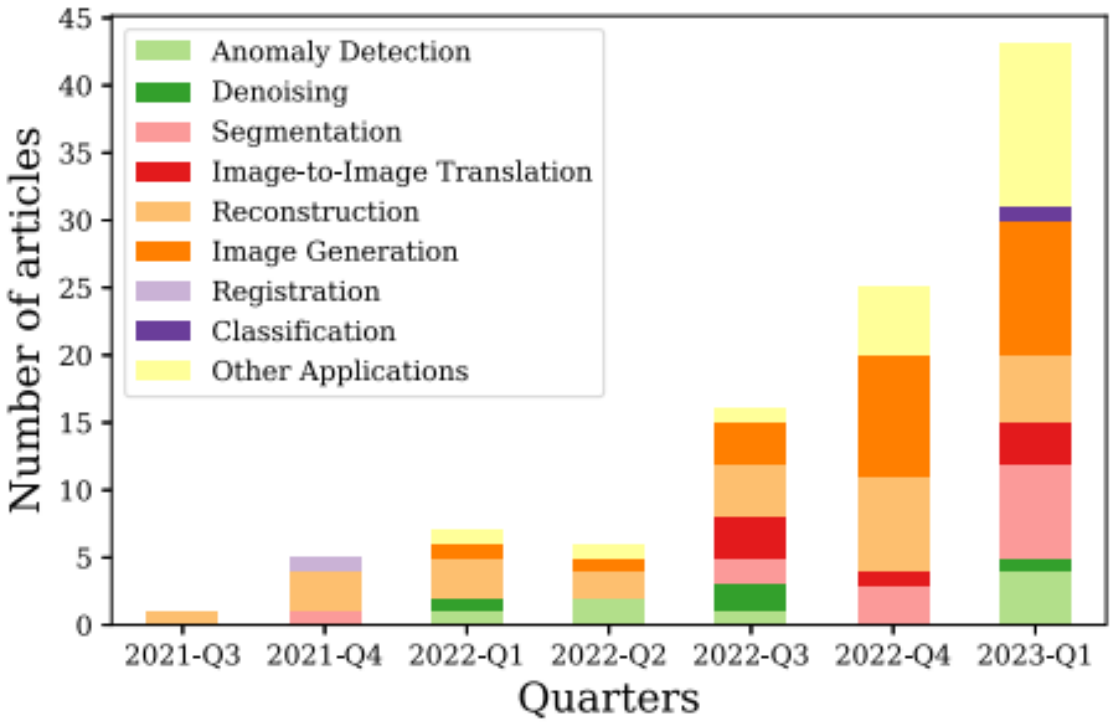

Fig. 1.2. The diagram shows the number of diffusion-based research papers published in the medical domain, highlighting an annual growth rate and emphasizing the significance of diffusion models for future research. The total number of papers is 103. [4]

Denoising diffusion probabilistic models (DDPMs) [17] have recently outperformed other approaches in modelling the distribution of natural images, excelling in both realism and diversity of samples. These advantages have enabled DDPMs to achieve impressive results, often surpassing GANs in various applications [18].

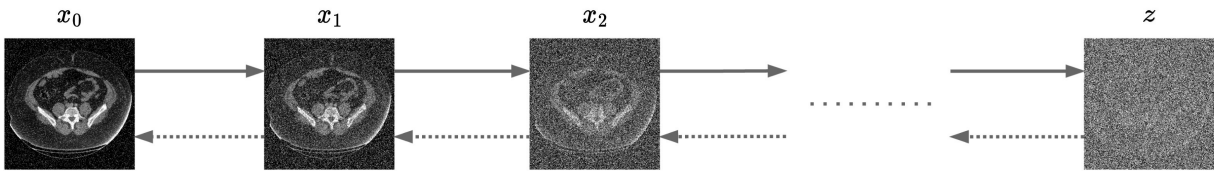

Fig. 1.3. This figure illustrates diffusion models, which progressively add noise to the input until it becomes a noise distribution, then apply a reverse process to remove the noise step by step during sampling. [4]

The scientific community's interest in diffusion models has led to the development of various DDPM variants in recent years, specifically aimed at achieving better performance and enhancing the models' efficiency both in terms of time and computational resources required for training and sampling. Among these are denoising diffusion implicit models (DDIMs) [19] and latent diffusion models (LDMs) [20]. In addition to these structural and methodological variants of diffusion models, alternative training techniques have also been developed in recent years to make the training process more effective for applying models to three-dimensional volumes, such as patch-based techniques [21].

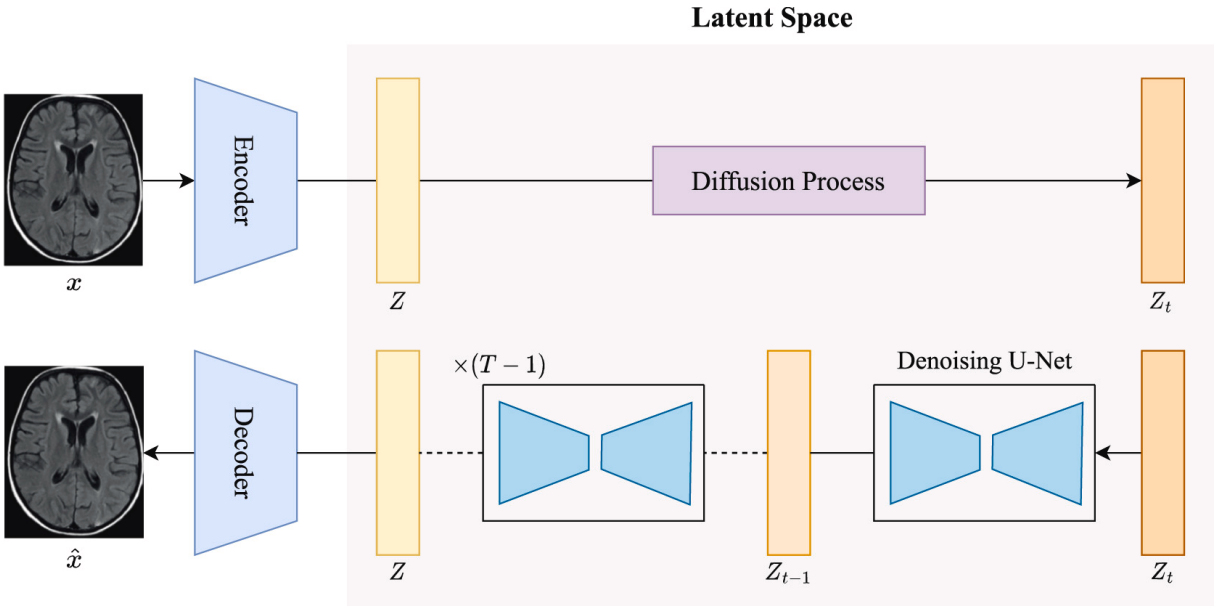

Fig. 1.4. The diagram shows a schematic representation of latent diffusion models functioning. [4]

Overall, deep generative models of all kinds have produced high-quality samples across various data modalities. GANs, autoregressive models, flows, and VAEs have generated remarkable images and audio samples, while advances in energy-based modeling and score matching have produced images comparable to those of GANs. Diffusion models, in particular, are straightfor-

ward to define, efficient to train, and capable of generating high-quality samples, demonstrating superiority over other generative architectures even in discriminative tasks.

1.2 Training Strategies

1.2.1 Transfer Learning

In recent years, deep neural networks, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), graph neural networks (GNNs), and attention neural networks, have been extensively utilized for a variety of artificial intelligence tasks. Unlike traditional non-neural models that heavily depend on hand-crafted features and statistical methods, neural models can automatically learn low-dimensional continuous vectors from data, which serve as task-specific features. Despite their success, deep neural networks face a significant challenge: they are data hungry. With their large number of parameters, these models are prone to overfitting and have poor generalization ability when trained on insufficient data [22].

To address this issue, substantial efforts have been made to create high-quality datasets for AI tasks, facilitating the learning of effective neural models that outperform conventional non-neural models. However, investing in data gathering is increasingly impractical due to the rarity, inaccessibility, expense, time consumption and difficulty of compiling these datasets [23].

A breakthrough in tackling this problem is the introduction of transfer learning (TL). Transfer learning is a machine learning technique where a model developed for a particular task is reused as the starting point for a model on a second task. It leverages the knowledge gained from a pre-trained model on a large dataset to improve the performance of a related task with limited data. This approach is particularly useful in scenarios where labeled data is scarce or expensive to obtain.

Transfer learning has become increasingly popular due to its effectiveness in improving model accuracy, reducing training time, and minimizing the need for extensive labeled data. It is widely applied in various fields such as natural language processing, computer vision, and speech recognition.

The concept of transfer learning is based on the hypothesis that many tasks share common underlying features. For instance, in computer vision, the initial layers of a convolutional neural network typically capture generic features like edges and textures, which are useful for many different tasks. By reusing these layers, we can save computational resources and improve model performance. However, it is important to note that transferred knowledge does not always positively impact on new tasks; if the target learner is negatively affected by the transferred knowledge, the phenomenon is termed negative transfer [24].

The study of TL is heavily motivated by the fact that humans can rely on previously learned

knowledge to solve new problems and even achieve better results. More formally, transfer learning aims to capture important knowledge from multiple source tasks and then apply the knowledge to a target task. The primary components of TL include [25]:

- Source Domain (D_s): The domain from which the model is initially trained.
- Target Domain (D_t): The domain to which the model is transferred.
- Source Task (T_s): The task associated with the source domain.
- Target Task (T_t): The task associated with the target domain.

and the final goal is to improve the performance on T_t by leveraging knowledge from D_s and T_s.

In transfer learning, source tasks and target tasks may have completely different data domains and task settings, yet the knowledge required to handle these tasks is consistent [22]. It is thus important to select a feasible method to transfer knowledge from source tasks to target tasks. To this end, various methods have been proposed to facilitate the transfer of knowledge from source tasks to target tasks. These methods can be broadly categorized into:

- Feature Extraction: Feature extraction methods pre-train effective feature representations to pre-encode knowledge across domains and tasks [26]. The pre-trained model is then used to extract useful feature from the target domain data. Typically, the last few layers of the model are replaced with layers specific to the target task, while the rest of the model remains unchanged. For instance, a convolutional neural network (CNN) pre-trained on ImageNet can be used to extract features from medical images, which are then fine-tuned on a smaller medical dataset to improve diagnostic accuracy.
- Fine-Tuning: This method follow an intuitive assumption that source tasks and target tasks can share model parameters or prior distributions of hyper-parameters. Fine-tuning involes taking a pre-trained model and updating the weights for all or some layers based on the target task data [27]. This requires careful tuning of hyperparameters, such as the learning rate, and the selection of the pre-trained layers to unfreeze in order to avoid catastrophic forgetting of the pre-traiend knowledge. An example is using a language model pre-trained on a large corpus of text and fine-tuning it on a specific downstream task like sentiment analysis or question answering.
- Domain Adaptation: Domain adaptation addresses situation where the D_s and D_t have different distributions [28]. Techniques like adversarial training or domain adversarial neural networks are used to minimize the discrepancy between the source and target domains, ensuring that the knowledge transfer is effective even when the domains differ significantly.

- Multi-task Learning: In multi-task learning, the model is trained simultaneously on multiple related tasks, allowing the model to learn shared representation that benefit all tasks [29]. This approach can be seen as a form of transfer learning where the transfer happens during the training process.

- Self-supervised Learning: Self-supervised learning involves training a model on pretext task for which labels are automatically generated [30]. The pre-trained model can then be fine-tuned on the target task, leveraging the representations learned from the pretext task.

Transfer learning applied in the context of deep learning offers several significant advantages:

- Reduced training time: By starting with a pre-trained model, the number of epochs required to achieve good performance on the target task is significantly reduced.

- Improved model performance: Transfer learning often results in higher accuracy and better generalization, particularly when the target dataset is small.

- Lower data requirements: Effective learning can occur even with limited labeled data in the target domain, as the pre-trained model provides a strong initialization.

- Increased versatility: Models pre-trained on large and diverse datasets can be adapted to a wide range of tasks, making transfer learning highly versatile.

Diffusion models usually employ a U-Net-like neural network for the denoising. Training such a model from scratch is challenging due to the scarcity of labeled images, particularly in the medical field. Transfer learning can be very useful here to pre-train the network on publicly available datasets and then fine-tune it on the target dataset to adapt the model weights to the specific characteristics of the target domain. During this fine-tuning phase, a lower learning rate is used to prevent overfitting.
Transfer learning offers significant advantages for enhancing diffusion models, including improved performance, reduced training time, and lower data requirements. By leveraging pre-trained models, diffusion models can achieve higher quality and efficiency, making them highly effective for tasks such as image generation, data augmentation, and beyond. The integration of transfer learning into diffusion models showcases the versatility and power of this technique in modern deep learning applications.

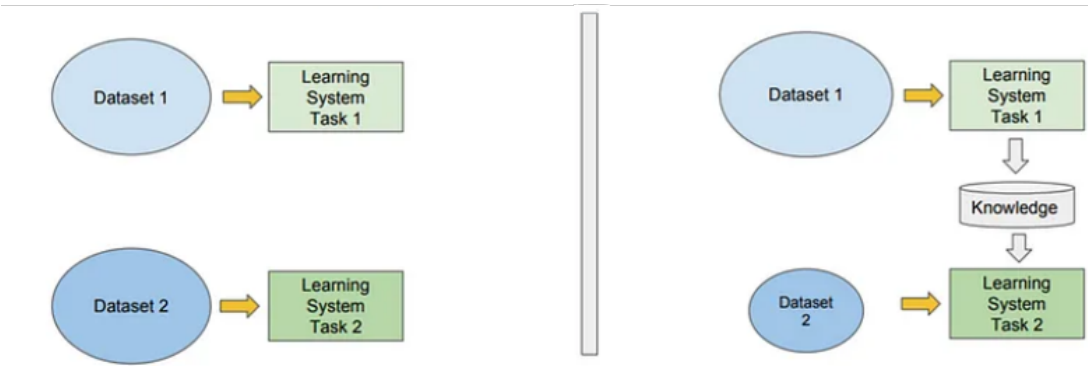

Fig. 1.5. Schematic comparison between traditional learning and transfer learning methods. (A Comprehensive Hands-on Guide to Transfer Learning with Real-World Applications in Deep Learning)

1.2.2 Self-supervised learning

Numerous real-world deep learning applications encounter the issue of having a large amount of unlabeled training data while only a small fraction is labeled. Obtaining labeled examples can be expensive, labor-intensive, and time-consuming, as it requires skilled human annotators with domain-specific expertise. For example, in the medical field, unlabeled data can be readily obtained from routine medical exams. However, assigning diagnoses to a vast number of cases imposes a significant burden on medical professionals. Additionally, biases and discrepancies among experts can introduce uncertainty into the ground truth labels [30].

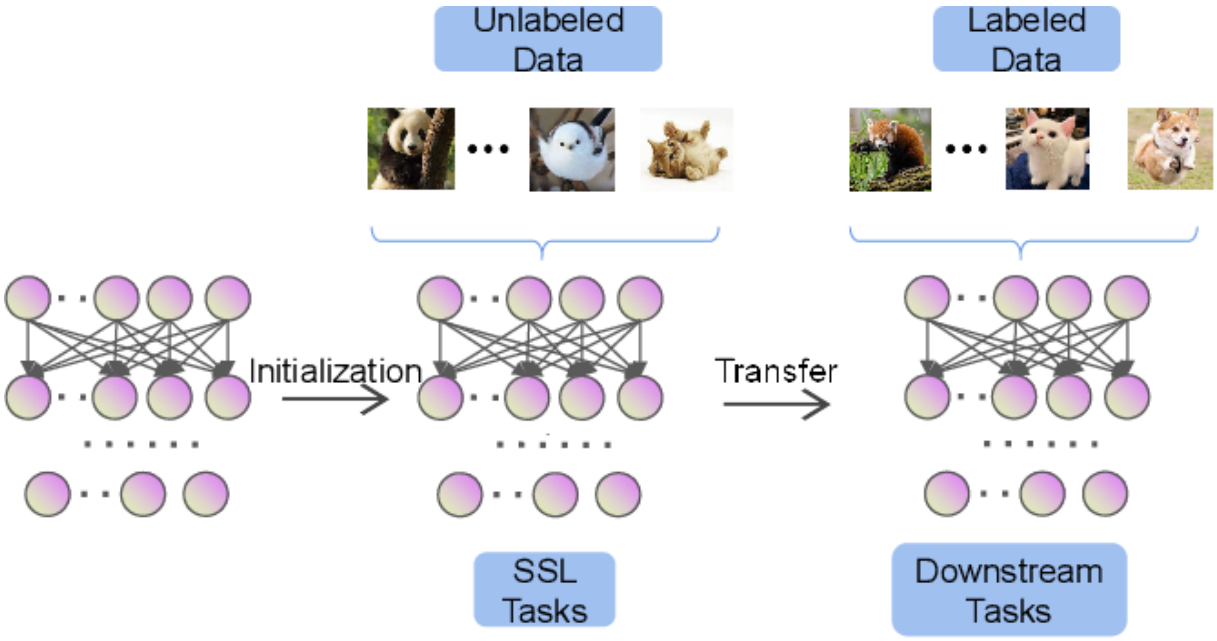

Fig. 1.6. Schematic representation of the SSL framework. [30]

To overcome these challenges of supervised learning, various machine learning paradigms have emerged, including self-supervised learning (SSL). Self-supervised learning [31] is a prominent example of a transfer learning method, particularly valued for its ability to operate without relying on labeled data. This makes it exceptionally useful in real-world applications where there is an abundance of unlabeled training instances and a limited number of labeled ones. SSL algorithms capitalize on extensive unlabeled data by generating pseudo-labels for pre-training, which allows models to leverage the vast amount of available data to learn meaningful representations.

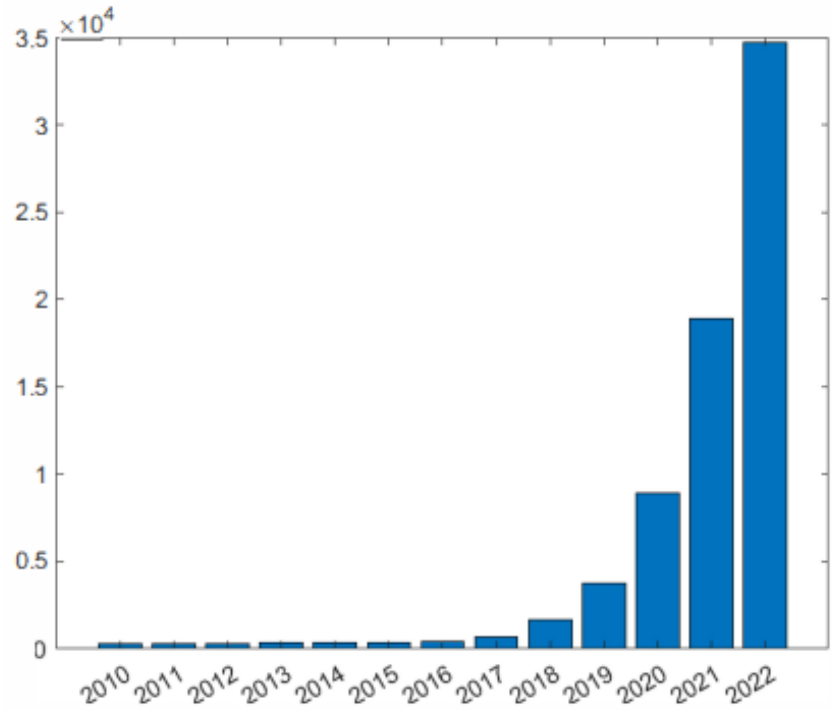

Fig. 1.7. Barchart depicting the number of SSL publications by year. [30]

In SSL, the process of generating output labels is intrinsic to the input data. These labels are derived directly from the data examples themselves by revealing the relationships between data components or various views of the data. An autoencoder (AE), for instance, can be viewed as an SSL algorithm where the output labels are the data itself. This intrinsic labeling allows the model to learn features that are beneficial for downstream tasks without needing external annotations.

The essence of SSL is akin to solving a puzzle where parts of the input are missing, and the goal is to predict those missing segments. This methodology can be generalized to encompass any approach that operates without human-annotated labels, thus broadening the scope of SSL.

Pretext tasks, also known as surrogate or proxy tasks, are central to SSL. These tasks are not the primary objectives but serve as a means to generate robust pre-trained models. Each pretext task requires distinct loss functions to achieve its goals. The fundamental idea is to create tasks that, when solved, force the model to learn useful representations of the data. Common categories of pretext tasks include:

- Context-based methods.
- Contrastive learning (CL).
- Generative algorithms.
- Contrastive generative methods.

Context-based methods rely on the inherent contextual relationships among data examples, including spatial structures and both local and global consistency. An example is rotation prediction, where a model learns to recognize geometric transformations applied to images. A deep neural network is trained to predict the rotation transform, thus learning valuable image representations in the process. Given an image, different rotated version of it are generated and classified accordingly to the rotation angle applied. Thus, by learning to recognize these transformations, the model captures features relevant to the image's spatial properties.

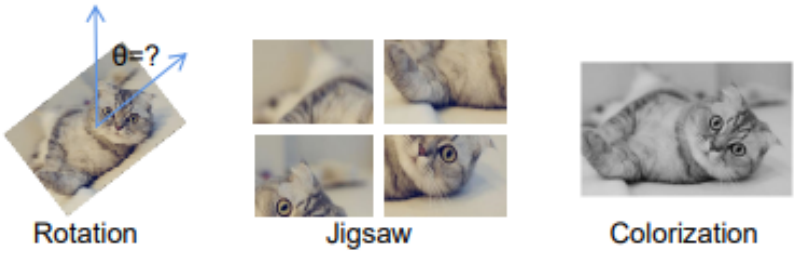

Fig. 1.8. Three common examples of input in context-based methods.

Another context-based method is image colorization. Color prediction offers the advantageous feature of requiring freely available training data. The deep neural network utilizes the lightness channel L of a color image as input, and predicts the color channels ab in the CIE Lab color space. The network is trained using a multinomial cross-entropy loss between the true and predicted color channels [32], allowing it to learn how to restore a grayscale image to its original colorful state by concatenating together L, a and b.
The jigsaw approach divides an image into patches that are shuffled. The model's task is to

reconstruct the original order of the patches, which encourages it to understand the contextual relationships within the image.

Contrastive SSL aims to learn invariant representations by contrasting different augmented views of the same image. These methods extract the information in unannotated images by treating each unannotated image as a positive pair with its counterpart supervisory signal obtained through some transformation and considering the supervisory signals of other unannotated images as negative pairs. In other words, views originating from the same instance are treated as positive examples for an anchor sample, while views from different instances serve as negative examples. The goal is to pull together representations of similar (positive) pairs and push apart those of dissimilar (negative) pairs within a latent space.

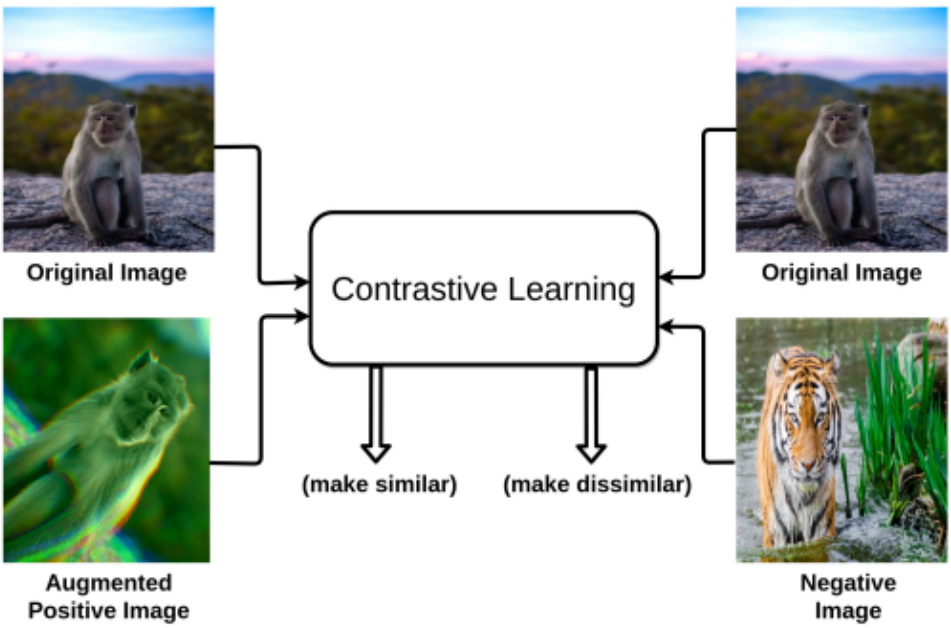

Fig. 1.9. Schematic representation of contrastive learning principle.[33]

For example, the Momentum Contrast (MoCo) [34] framework uses a query q and a dictionary of keys $k_0, k_1, k_2 ...$ to learn visual representations by optimizing a contrastive loss function. More in details, MoCo maintains the dictionary of encoded representations that is updated with a momentum mechanism. The model learns representations by contrasting the query with the keys in the dictionary, using a contrastive loss to align similar representations and separate dissimilar ones. The value of this function is low when q is similar to its positive key k_+ in the dictionary (which is unique) and dissimilar to all other negative keys. In the MoCo framework,

the InfoNCE loss function [35], a form of contrastive loss, is utilized:

$$L_q = -\log \frac{\exp q\frac{k_+}{\tau}}{\sum_{i=0}^{K} \exp q\frac{k_i}{\tau}}$$

where τ represents the temperature hyper-parameter.

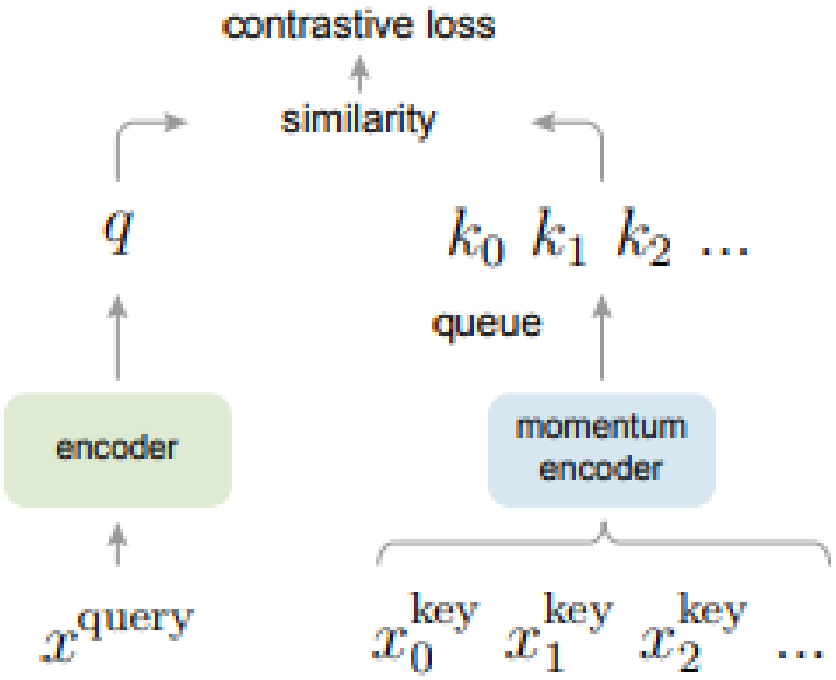

Fig. 1.10. Schematic representation of Momentum Contrast (MoCo). MoCo trains a visual representation encoder by matching an encoded query q to a dictionary of encoded keys using a contrastive loss. Keys are encoded by a momentum-updated encoder, providing a large and consistent dictionary for learning visual representations.[34]

Similarly, SimCLR [36] uses a mini-batch sampling strategy with N instances to form positive and negative pairs, leveraging a contrastive loss to learn useful representations. Each instance in the batch is augmented twice, generating a total of $2N$ instances, and the model learns to bring the augmented views of the same instance closer in the representation space while pushing apart views from different instances. Notably, SimCLR does not explicitly select negative instances. Instead, for a given positive pair, the remaining $2(N-1)$ augmented instances in the mini-batch are treated as negatives. Let $sim(u, v) = u^T v/(\|u\|\|v\|)$ represent the cosine similarity between two instances u and v. The loss function of SimCLR for a positive instance pair (i,j) is

defiend as:

$$l_{i,j} = \log \frac{\exp \frac{sim(z_i,z_j)}{\tau}}{\sum_{k=1}^{2N} 1_{[k \neq i]} \exp \frac{sim(z_i,z_k)}{\tau}}$$

where $1_{[k \neq i]} \in 0,1$ is an indicator function equal to 1 if $k \neq i$, and τ denotes the temperature hyper-parameter. The overall loss is computed across all positive pairs, including both (i,j) and (j,i), within the mini-batch.

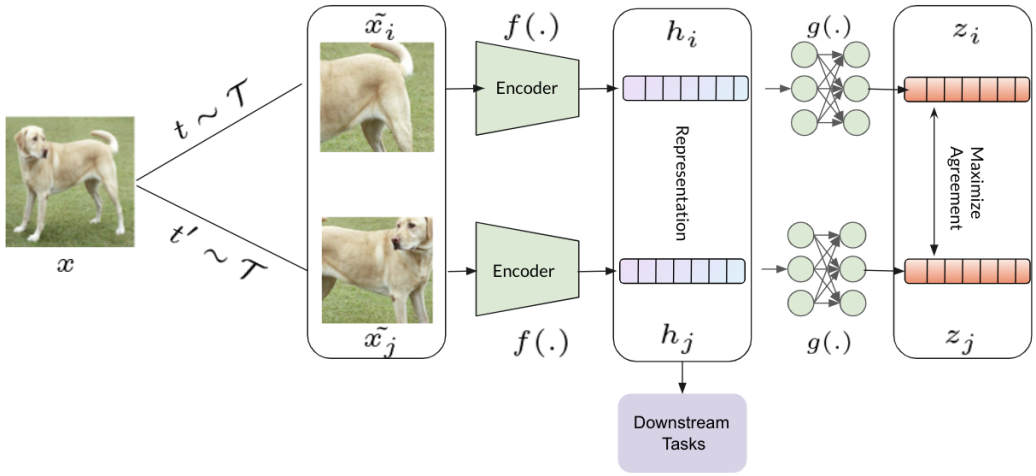

Fig. 1.11. Schematic representation of SimCLR functioning. Two data augmentation operators are sampled from the same family and applied to each data example to create two correlated views. A base encoder $f(\cdot)$ and a projection head $g(\cdot)$ are trained to maximize agreement using a contrastive loss. After training, the projection head is discarded, and the encoder and representation h are used for downstream tasks.(SimCLR Visually Explained)

Generative SSL methods involve generating plausible outputs from corrupted or incomplete inputs, obtained through different transformation. A common approach uses autoencoders where the encoder compresses the data into a latent representation and the decoder reconstructs the original input [37]. The model is trained using a reconstruction loss that evaluates how well the reconstructed output matches the original input, thus learning features that are useful for data reconstruction and other downstream tasks.

Contrastive generative methods combine elements of both contrastive and generative approaches. For example, a hybrid loss function might incorporate both a reconstruction loss and a contrastive loss. The encoder's latent representations are optimized using a contrastive loss, while the decoder's output is optimized using a reconstruction loss, providing a balanced learning approach. The latter ensure the decoder accurately rebuilds the input while the former

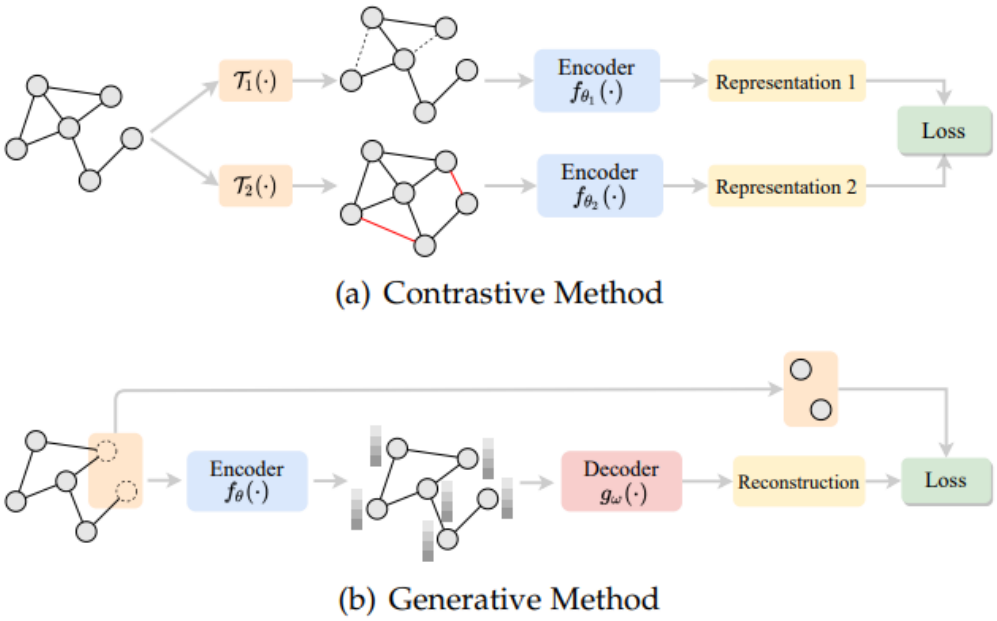

Fig. 1.12. Schematic comparison of contrastive **(a)** and generative **(b)** SSL method applied to graph data.[38]

ensure that similar inputs are close in the latent space. The overall function could be:

$$L = L_{rec} + L_{cl} L_{rec}$$

where L_{rec} is the reconstruction loss and L_{cl} is the contrastive loss. This scheme helps in balancing the influence of the two losses in a generative SSL framework.
Diffusion models can also be applied in SSL [39]. The forward process of these models can be used to add noise to the input image x_0 iteratively, producing a corrupted version x_t. A U-Net-based network then predicts the noise that takes x_0 to x_t, using the difference between predicted and actual noise to optimize the network. The pre-trained network, which learns to extract meaningful features through this process, can then be fine-tuned for specific downstream tasks. During the fine-tuning, is necessary to always impose $t = 0$ since the network has embedding layers for the pretext diffusion task.

In summary, self-supervised learning encompasses a broad range of techniques that allow models to learn from unlabeled data by creating and solving pretext tasks. These tasks force the model to uncover useful data representations that are beneficial for various downstream applications. By leveraging the inherent structure and relationships within the data, SSL methods can achieve remarkable performance without the need for extensive labeled datasets.

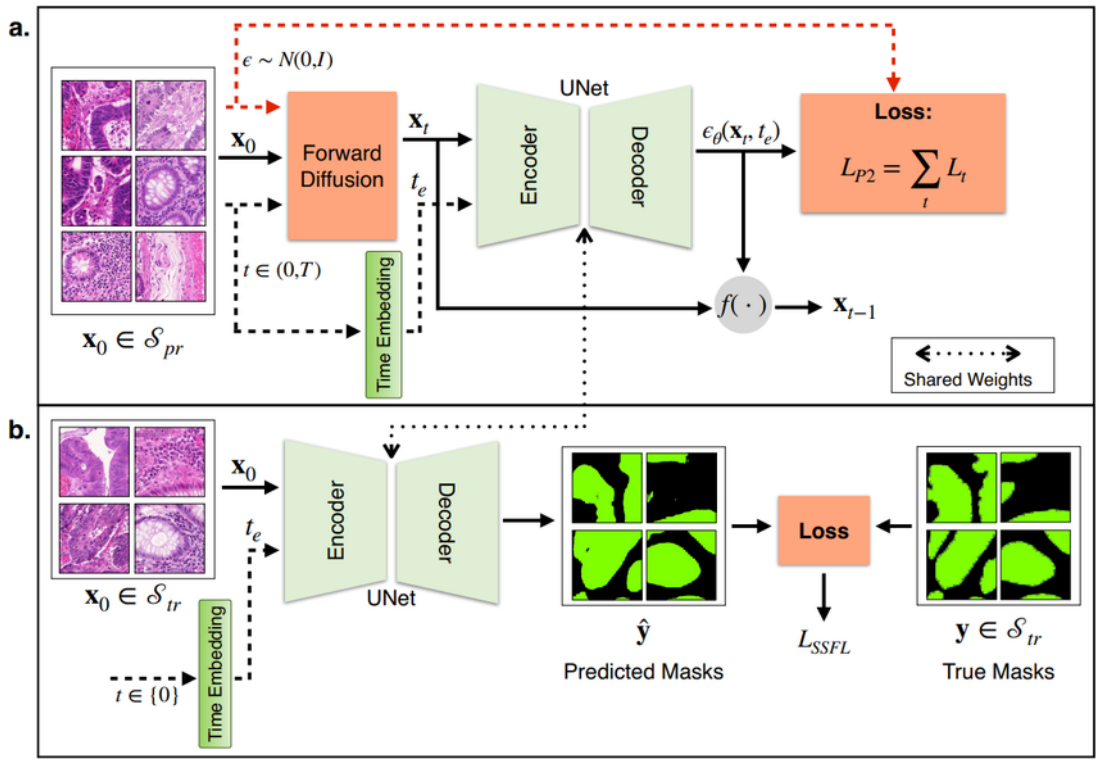

Fig. 1.13. Schematic representation of GenSelfDiff, an example of framework implementing diffusion for pre-training with histological images. **(a)** Self-supervised pre-training with diffusion: The UNet model (encoder-decoder) processes the corrupted image x_t along with the corresponding time embedding t_e to predict the noise that transforms x_0 into x_t, utilizing a P2 weighted loss function. $f(\cdot)$ represents a function that reconstructs x_{t-1} from x_t. **(b)** Downstream segmentation: The self-supervised pre-trained UNet is fine-tuned in a supervised manner to accurately predict segmentation masks. [39]

1.3 Medical image segmentation

Medical imaging is a cornerstone of modern healthcare, offering non-invasive diagnostic capabilities by generating structural and functional representations of the human body and its organs. These images are essential for diagnosing a multitude of conditions. The field of medical imaging can be broadly divided into two main components: image formation and image processing and analysis.

Image formation involves generating two-dimensional (2D) images from three-dimensional (3D) objects, often utilizing iterative algorithms to reconstruct images from projection data. On the other hand, image processing focuses on enhancing image properties, such as removing noise, while image analysis extracts quantitative information or features to identify or classify objects within the image.

Technological advancements have greatly facilitated image acquisition, leading to the production of vast quantities of high-resolution images at reduced costs. This surge in image data

has spurred significant progress in biomedical image processing algorithms, paving the way for automated image analysis techniques that can extract valuable information [40]. A critical step in automated analysis is segmentation, which divides an image into distinct regions based on visual characteristics that are meaningful for a given problem. These regions often have consistent gray levels, textures, or colors. Accurate segmentation is crucial for further analysis, such as assessing texture homogeneity or measuring layer thickness.

In the context of image segmentation, there is a fundamental distinction between instance and semantic segmentation. Sometimes the image may contain multiple objects of the same class, and the segmentation process may segregate regions containing these objects while ignoring other classes. This is known as instance segmentation. Conversely, semantic segmentation differentiates between classes but does not segregate objects of the same class. Semantic segmentation is a well-studied area in medical image analysis, with automated lesion segmentation using machine learning showing promising diagnostic results [13].

Another important distinction in the segmentation field is between manual, semi-automatic and fully automatic segmentation. Manual segmentation, where experts delineate regions of interest (ROI) by drawing precise boundaries to annotate each pixel, is vital for creating ground truth labeled images needed for developing semi-automatic and fully automatic segmentation methods. However, this approach is time-consuming, subjective, and impractical for large datasets. Moreover, manual segmentation is highly dependent on the expertise and experience of the annotator, leading to variability between different experts and even the same expert over time. Semi-automatic segmentation methods require limited user input combined with automated algorithms to achieve accurate segmentation. Typically, the user selects an initial ROI, which the algorithm then uses to segment the entire image. On the other hand, fully automatic segmentation methods eliminate user interaction but often rely on supervised learning approaches that require extensive labeled training data, usually obtained through manual segmentation. For this reason, they are affected by the same challenges and limitations mentioned before.

In recent years, deep learning techniques have become prominent in semantic segmentation for both natural and biomedical images. The rise of deep learning in this field has been driven by the availability of powerful central processing units (CPUs) and graphics processing units (GPUs), access to large datasets, and advancements in learning algorithms. Deep learning models, such as U-Net and its variants, have significantly improved the accuracy and efficiency of segmenting anatomical structures and abnormalities from imaging modalities like MRI and CT, becoming the state of the art architecture for medical segmentation. An example of this is given by nnU-Net, which is currently considered the best choice for biomedical image segmentation [41].

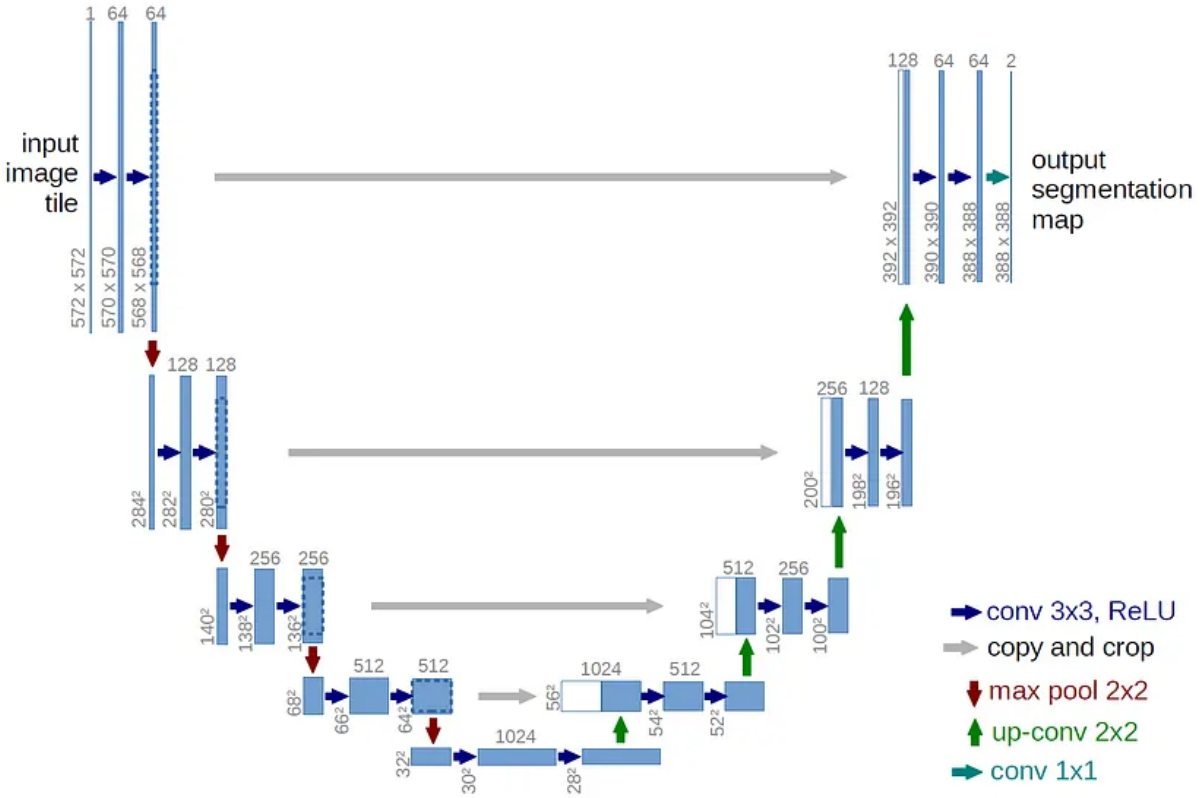

Fig. 1.14. U-Net architecture. [42]

Generative models such as GANs and VAEs have been applied as data augmentation techniques to improve the performance of segmentation models [43]. In the last years, diffusion models have also been successfully applied for medical image segmentation, demonstrating the potential to outperform conventional and current state-of-the-art models. This can be achieved by exploiting the conditional process of the models, which allows the use of images as anatomical prior.

In conclusion, image segmentation plays a vital role in medical image analysis by providing automated methods to precisely delineate and separate different anatomical structures and pathological entities. Automated segmentation allows clinicians to obtain detailed visualizations and quantitative assessments of lesions and other structural anomalies, thereby facilitating computer-aided diagnosis, treatment planning, and monitoring of disease progression. Therefore, multidisciplinary research efforts are essential to develop new fully automatic segmentation methods that are both computationally and temporally efficient, aiding clinicians in the medical field.

1.4 The Choroid plexus

The choroid plexus is a small, highly vascular structure located within the lateral, third, and fourth ventricles of the brain. It is primarily responsible for producing cerebrospinal fluid (CSF) through the activity of ependymal cells lining the ventricles. This production is essential for maintaining the brain's environment, providing cushioning, regulating intracranial pressure, and removing metabolic waste [44]. The choroid plexus also forms the blood-CSF barrier, a critical interface that controls the exchange of substances between the blood and CSF, thereby protecting the brain from toxins and pathogens. Additionally, it secretes various growth factors that maintain the stem cell pool in the subventricular zone, playing a role in brain development and repair [45].

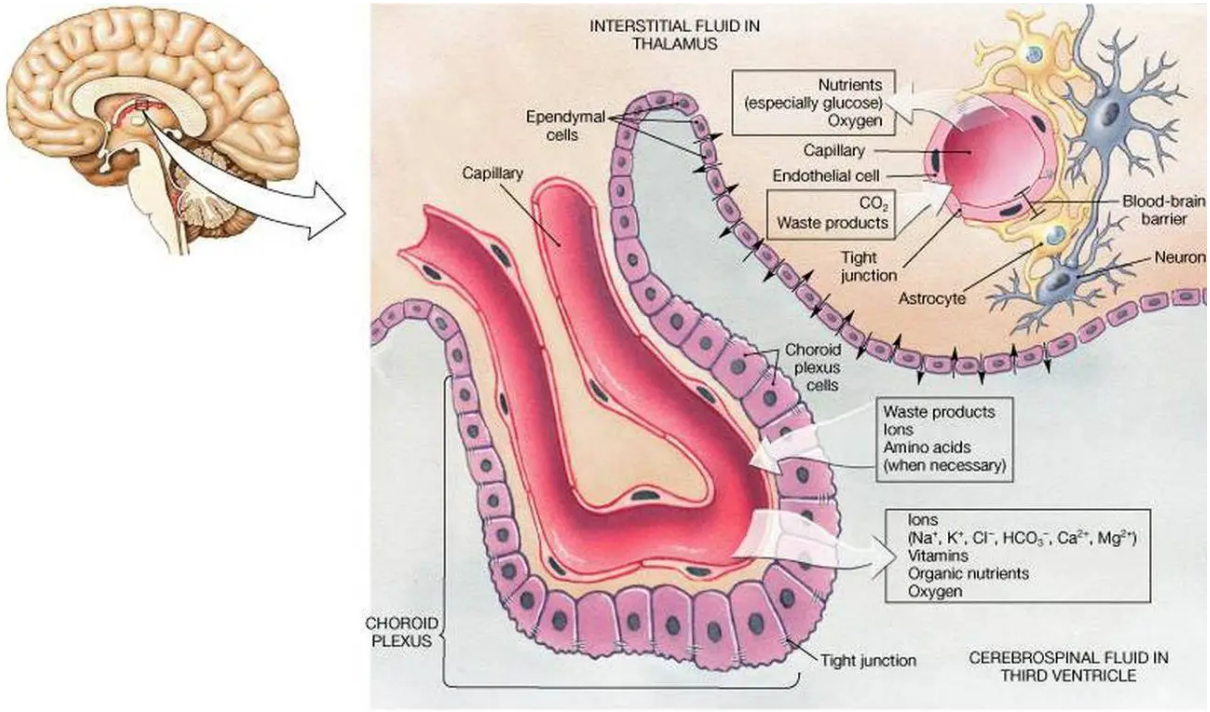

Fig. 1.15. Schematic representation of the choroid plexus and its functionalities. [46]

Pathological conditions affecting the choroid plexus include a spectrum ranging from benign cysts to malignant carcinomas and metastases. These abnormalities can manifest in different forms: focal enlargements, such as papillomas and carcinomas, present as localized growths, while diffuse enlargements, like diffuse villous hyperplasia of the choroid plexus (DVHCP), result in widespread tissue expansion. Changes in the choroid plexus structure and function have been linked to various neurological and psychiatric disorders. For instance, in Alzheimer's disease, the choroid plexus may contribute to impaired CSF circulation and amyloid-beta accumulation. Its role in immune regulation and surveillance is also significant in conditions

such as multiple sclerosis, where it influences immune cell trafficking into the central nervous system [47].

Given the choroid plexus's role in numerous pathophysiological conditions, accurate segmentation in MRI scans is crucial for both clinical and research purposes. MRI provides detailed images of the brain's soft tissues, making it an effective tool for visualizing the choroid plexus. However, segmentation is challenging due to the structure's small size, complex morphology, and individual variability. Advanced image processing techniques and deep learning algorithms are increasingly used to enhance segmentation accuracy and efficiency. These technological advancements have significant potential to improve the diagnosis and monitoring of conditions such as hydrocephalus, Alzheimer's disease, and other disorders involving CSF dynamics [48].

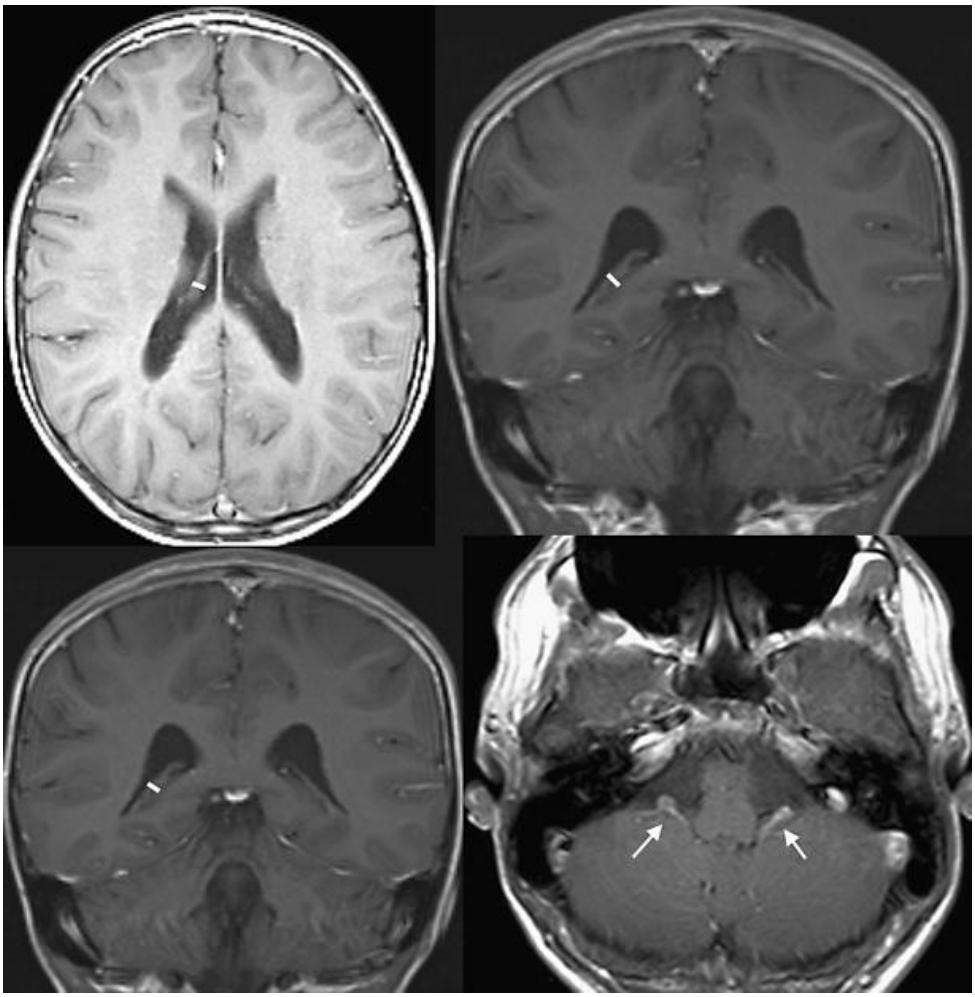

Fig. 1.16. Brain MRI scans with short white lines and arrows to highlight the choroid plexus. [47]

As mentioned before, the state-of-the-art in medical segmentation, including choroid plexus segmentation from MRI images, is currently represented by the U-Net and its variants. Eisma

et al. [49] developed a method using a fully convolutional U-Net architecture, which involves registering input images to MNI152 space, cropping around the choroid plexus based on a probabilistic atlas, and using these cropped images to train a 3D U-Net with 41 overlapping patches per participant. The outputs are then reassembled in MNI space and transformed back to the native imaging space. Yazdan-Panah et al. [48] proposed a method using two cascaded 3D U-Nets. The first U-Net processes down-sampled whole images to identify probable choroid plexus locations. Patches are then randomly selected around high-probability voxels, which the second U-Net uses to segment the choroid plexus. The segmented patches are merged using a Hann windowing technique to create the final segmentation mask, ensuring smooth integration and normalization of the output.

In conclusion, understanding the anatomy and function of the choroid plexus is fundamental to advancing knowledge of the brain's ventricular system and its role in health and disease. Accurate MRI segmentation of the choroid plexus is a critical step in diagnosing and researching various neurological conditions. Continued research and technological advancements in this field are essential for improving clinical outcomes and expanding our understanding of the central nervous system.

1.5 Aim of the thesis

Given the growing interest in diffusion models within the scientific community and their expanding use in medical research, this study aims to develop a method utilizing Denoising Diffusion Probabilistic Models for the segmentation of the choroid plexus in brain MRI images. Inspired by [13], who successfully applied diffusion models to segment brain tumors with superior performance compared to current state-of-the-art methods, this study seeks to replicate and adapt their approach specifically for choroid plexus segmentation.

In pursuit of further improving the model's performance, various pre-training and self-supervised learning techniques have been considered and evaluated. Therefore, the objective of this work is to develop a methodology that leverages recent technological advancements in deep learning to assist the medical field in the study and analysis of the pathophysiology of the choroid plexus, a delicate and critical brain structure.

Chapter 2

State of the art

2.1 Diffusion Models

2.1.1 Denoising Diffusion Probabilistic Models

Denoising Diffusion Probabilistic Models represent a powerful class of generative models that leverage the concept of diffusion processes to generate high-quality samples from complex data distribution. These models have gained significant attention for their ability to produce realistic data by iteratively denoising randomly sampled noise. This section describes the detailed functioning of DDPMs, including the forward and reverse processes, training procedure, and inference.

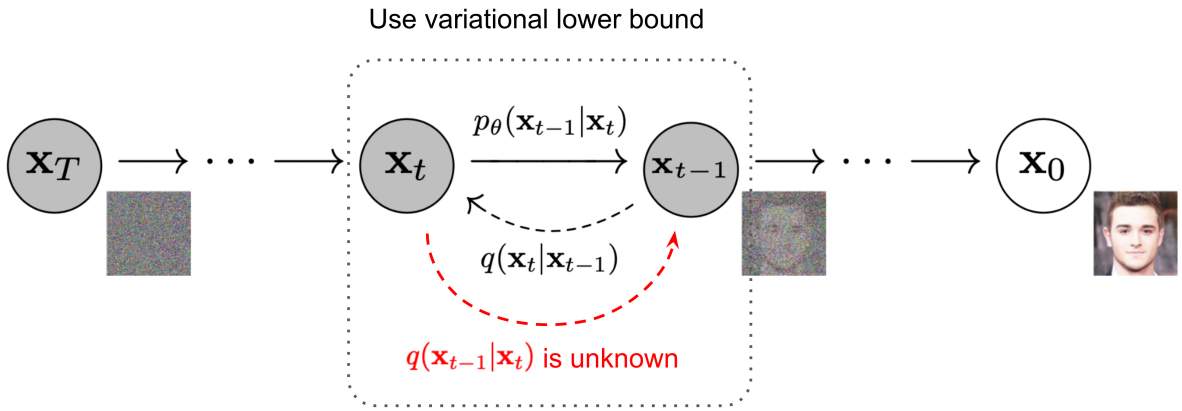

Fig. 2.1. The diagram represents the forward and reverse diffusion process in DDPMs.(What are Diffusion Models?)

DDPMs are latent variable models where x_1,..., x_T are latent of the same dimensionality as the data $x_0 \sim q(x_0)$. In fact, the forward diffusion process in Denoising Diffusion Probabilistic Mod-

els is a Markov chain that gradually adds Gaussian noise to the data over a series of time steps [50]. Therefore, given an input data point x_0 (e.g., an image), the forward process produces a sequence of noisy samples x_1, x_2..., x_T where T is the total number of time steps. This process can be formulated as:

$$q(x_t|x_{t-1}) \ = \ \boldsymbol{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t \mathbf{I})$$

where β_t is a variance schedule that controls the amount of noise added at each time step t. The entire forward process can be expressed as:

$$q(x_{1:T}|x_0) \ = \ \prod_{t=1}^{T} q(x_t|x_{t-1})$$

A notable property of the forward process is that it admits sampling x_t at any arbitrary time step t in closed form [17]: using the notation $\alpha_t \ = \ 1 - \beta_t$ and $\overline{\alpha}_t \ = \ \prod_{s=1}^{t} \alpha_s$, we have

$$q(x_t \mid x_0) \ = \ N(x_t; \sqrt{\overline{\alpha}_t}x_0, (1-\overline{\alpha}_t)\mathbf{I})$$

The reverse diffusion process aims to denoise the noisy sample x_T back to the original data point x_0. This process is also a Markov chain but in the reverse direction:

$$p(x_{t-1}|x_t) \ = \ \boldsymbol{N}(x_{t-1}; \mu_\theta(x_t,t), \Sigma_\theta(x_t,t))$$

where $\mu_\theta(x_t,t)$ and $\Sigma_\theta(x_t,t)$ are the mean and the variance predicted by a neural network parametrized by θ. Therefore, the goal during training is to learn the parameters θ that minimize the discrepancy between the true posterior $q(x_{t-1}|x_t)$ and the model posterior $p_\theta(x_{t-1}|x_t)$. Training a DDPM involves optimizing the parameter of the neural network to approximate the reverse process. This is done by minimizing the variational bound on the negative log-likelihood of the data. The loss function can be derived as:

$$L \ = \ \mathbb{E}_q \left[\sum_{t=1}^{T} \left(D_{KL} \left(q(x_{t-1}|x_t, x_0) \parallel p_\theta(x_{t-1}|x_t) \right) \right) + \text{const} \right]$$

Here, D_{KL} denotes the Kullback-Leibler divergence between the true posterior $q(x_{t-1}|x_t, x_0)$ and the model posterior $p_\theta(x_{t-1}|x_t)$. This function measures how much one probability distribution $p_\theta(x_{t-1}|x_t)$ diverges from a second, reference probability distribution $q(x_{t-1}|x_t, x_0)$ and can be viewed as the amount of information lost when the former is used to approximate the latter. Thus, this loss encourages the model to learn the reverse diffusion process effectively [51].
The success of Denoising Diffusion Probabilistic Models hinges on their parametrization, which

directly affects their ability to accurately model the reverse diffusion process. This is possible since the mean and the variance of the real distribution can be written as follow:

$$\mu_t(x_t,x_0) \;=\; \frac{\sqrt{\overline{\alpha}_{t-1}}\beta_t}{1-\overline{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1-\overline{\alpha}_{t-1})}{1-\overline{\alpha}_t}x_t$$

and

$$\Sigma_t \;=\; \tilde{\beta}_t\mathbf{I} \quad \text{where} \quad \tilde{\beta}_t \;=\; \frac{1-\overline{\alpha}_{t-1}}{1-\overline{\alpha}_t}\beta_t$$

It is also possible to reparametrize x_t, allowing to define other possible strategies to train the diffusion model:

$$x_t(x_0,\epsilon) \;=\; \sqrt{\overline{\alpha}_t}x_0 + \sqrt{1-\overline{\alpha}_t}\epsilon \quad \text{for} \quad \epsilon \sim N(0,\mathbf{I})$$

Thanks to above mentioned reparametrisation and some straightforward calculations is possible to define different training strategies for the DDPMs, each with its own advanges and trade-offs:

- Mean and Variance Parametrisation:
  As stated before, in the original formulation of DDPMs, the neural network is used to predict both the mean $\mu_\theta(x_t,t)$ and the variance $\Sigma_\theta(x_t,t)$ of the reverse diffusion step. This parametrisation ensures that the model captures the full conditional distribution at each time step:

  $$p(x_{t-1} \mid x_t) \;=\; N(x_{t-1};\mu_\theta(x_t,t),\Sigma_\theta(x_t,t))$$

  The variance is often fixed or parametrised [8] in a way to ensure stability during training, for example, by using a schedule similar to the forward process variance schedule β_t. This means imposing $\Sigma_\theta(x_t,t) \;=\; \beta_t$. Once the predicted mean and variance are available it is possible to sample $x_{t-1} \sim p_\theta(x_{t-1} \mid x_t)$

- Noise Prediction Paramtrisation:
  A more popular approach, particularly for its simplicity and effectiveness, is the noise prediction parametrisation [17]. In this approach, the neural network predicts the added noise ϵ at each time step rather than directly predicting the mean and variance. The model learns to minimize the difference between the true noise added during the forward process and the noise predicted by the neural network:

  $$L_{simple} \;=\; \mathbb{E}_{t,x_0,\epsilon}[\|\epsilon - \epsilon_\theta(x_t,t)\|^2]$$

  where ϵ is the noise added to the data point x_0 at time step t, and ϵ_θ is the noise predicted by the neural network. This parametrisation is beneficial because it directly targets the denoising task, simplifying the learning objective and often leading to more stable training. Once the predicted noise at the time step t is available, is possible to compute x_{t-1}

as follow:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(x_t,t)\right) + \beta_t z \quad \text{where} \quad z \sim N(0,\mathbf{I})$$

- Direct prediction of x_0:
  Another approach involves predicting the original data x_0 from the noisy sample x_t [17]. This can be advantageous because it aligns the model's objective closely with the ultimate goal of reconstructing the original data. The model is trained to minimize the reconstruction error:

  $$L_{x_0} = \mathbb{E}_{t,x_0,\epsilon}[\|x_0 - x_\theta(x_t,t)\|^2]$$

  where x_0 is the original samples and $x_\theta(x_t,t)$ is its parametrisation provided by the model. Although this approach can be intuitive, it may not always capture the underlying noise distribution as effectively as noise prediction parametrisation.

- Score-Based Parametrisation:
  Inspired by score-based generative models, this approach involves parametrising the score function [52], which is the gradient of the log probability density with respect to the data. The neural network predict the score of the data distribution, which is then used to iteratively refine the noisy samples:

  $$L_{score} = \mathbb{E}_{t,x_0,\epsilon}[\|s_\theta(x_t,t) - \nabla_{x_t}\log(q(x_t \mid x_0))\|^2]$$

  where $s_\theta(x_t,t)$ is the parametrised score function provided by the model and $\nabla_{x_t}\log(q(x_t \mid x_0))$ is the real score function. Score-based parametrisation can provide a more principled approach to modeling the underlying data distribution, but it may require more complex training procedures. Langevin dynamics can produce samples from a probability density $p(x)$ using only the score function $\nabla_x \log(p(x))$. Given a fixed step size $\gamma > 0$, and an initial value $\tilde{x}_0$, the Langevin method recursively computes the following:

  $$\tilde{x}_t = \tilde{x}_{t-1} + \frac{\gamma}{2}\nabla_x \log(p(\tilde{x}_{t-1})) + \sqrt{\gamma} z_t \quad \text{for} \quad z_t \sim N(0,\mathbf{I})$$

  Therefore, by approximating the score function with $s_\theta(x_t,t)$ and inverting the equation, it is possible to implement the reverse diffusion process.

As mentioned before, each parametrisation techniques offers different trade-offs in terms of model complexity, training stability and performance:

- Mean and Variance Parametrisation provides a comprehensive approach but can be more challenging to train due to the need to accurately model both mean and variance.

- Noise Prediction Parametrisation is simpler and more stable, make it the most popular choise for many practical applications.
- Direct prediction of x_0 aligns the training objective with the end goal but may struggle with capturing the noise distribution accurately.
- Score-Based Parametrisation offers a theoretically grounded method but can be more complex to implement and train effectively.

In practice, the choice of parametrisation depends on the specific requirements of the application, the nature of the data, and the computational resources available.

Fig. 2.2. Examples of sampling from a DDPM with intermediate results. [17]

During inference, new data samples are generated by sampling from a Gaussian distribution and iteratively applying the learned reverse diffusion process in order to remove the noise and recover the image structures. This iterative denoising continues until the final sample x_0 is obtained, which should resemble one extracted from the training data distribution.
For the implementation of DDPMs there are three important aspect to define: the neural network architecture, the variance schedule and the training hyperparameters.
The neural network used in DDPMs typically employs a U-Net architecture, known for its effectiveness in capturing multi-scale features [42]. This is achieved thanks to the characteristic residual connections, that allow to keep track of the features extracted by the encoder and pass them to corresponding decoder levels. The input to the network includes the noisy samples x_t and the time step t, encoded using positional embeddings, necessitating modifications to the U-Net to incorporate layers capable of extracting them [53].
The variance schedule β_t in DDPMs dictates the rate at which noise is added during the forward diffusion process. Commonly used schedules include linear increasing with t or cosine annealing schedules that ensure a smooth transition from low to high noise levels across the diffusion steps t. The forward process variances can be learned by reparametrisation or held constant as hyperparameter [17]. The choice of scheduler impacts the model's ability to generate realistic samples and is crucial for stable training and accurate inference.

Setting training hyperparameters is essential, including the learning rate, optimizer, and batch size. The learning rate is a crucial hyperparameter in neural networks that controls how quickly the model updates its weights during training. A rate that is too high can lead to divergence, while one that is too low may slow down convergence. The optimizer is the algorithm that adjusts the model's weights to minimize the loss function; common choices include Stochastic Gradient Descent (SGD) and Adam, each with unique features designed to enhance learning efficiency and stability. Finally, the batch size determines the number of samples processed through the network in each iteration; larger batch sizes can take advantage of hardware parallelization but require more memory, whereas smaller batch sizes allow for more frequent updates, potentially affecting the training process's speed and stability.

The learning rate is crucial for DDPMs due to their iterative denoising process and must be chosen carefully to ensure stable training dynamics and avoid issues such as divergence or slow convergence. Typically, learning rates for diffusion models are selected through empirical testing and validation on training data, starting with a conservative initial value and adjust it based on observed training dynamics and performance metrics. The choice of the optimizer also plays a crucial role for the same reason mentioned before. Adam optimizer is commonly used for its ability to adaptively adjust learning rates for each parameter based on the gradients' magnitudes, thereby accelerating convergence and mitigating issues related to vanishing or exploding gradients. In addition to these, when working with DDPMs, is fundamental to set the number of diffusion steps T and the variance scheduler parameters β_t. These hyperparameters are typically tuned using techniques such as grid search or Bayesian optimization to maximize model performance and stability during training.

A key advantages of DDPMs lies in their ability to incorporate additional information or context c into the generative process through conditioning. During training and inference, DDPMs can conditionally generate samples x_T^{cond} that combine the noisy sample x_T with c:

$$x_T^{cond} = [x_T, c]$$

where x_T is the output of the forward diffusion process and c represents the conditioning data. In biomedical image sythesis, the conditioning data c can include various forms of information such as:

- Anatomical priors providing spatial constraints or structural information [13].
- Contextual information including subject demographics, clinical metadata, or environmental factors relevant to the data generation context [52].

The effectiveness of conditioning in DDPMs relies on the neural network architecture used to process the conditional inputs c. Commonly, architectures such as U-Net or convolutional neural

networks (CNNs) are employed to encode and fuse x_T and c, yielding parameters $\mu_\theta(x_T^{cond})$ and $\Sigma_\theta(x_T^{cond})$ of the distribution from which the denoised sample x_0 is estimated during the reverse diffusion process.
Implementing conditional DDPMs requires careful consideration of several practical factors:

- Relevance of Conditioning Data: Selecting conditioning data c that is informative and relevant to the generative task is crucial for enhancing sample quality and relevance.
- Computational Efficiency: Balancing the computational overhead of integrating conditioning data with the overall efficiency of model training and inference.
- Generalization Across Conditions: Ensuring that conditional DDPMs generalize well across different conditions or contexts for robust performance in diverse applications.

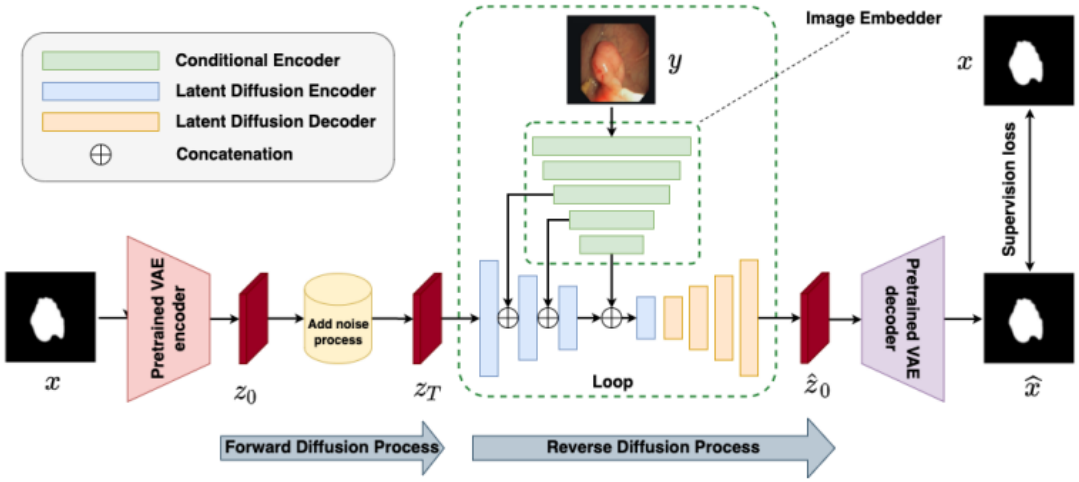

Fig. 2.3. Schematic representation of LSegDiff, an example of LDM that uses a conditional encoder with convolutional layers to extract features from the context before combining them with x_t. [7]

There are several effective techniques employed to integrate conditioning in DDPMs, thereby improving their performance across various applications. One straightforward approach involves concatenating [12], [13], [15], [54] the conditioning information c with the noisy sample x_T obtained from the forward diffusion process. This creates a joint input vector $x_T^{cond} = [x_T, c]$, which is subsequently processed by the neural network architecture of the DDPM. This method is widely used due to its simplicity and direct integration of c into the model's input space.

To handle diverse types of conditioning information c, multi-modal fusion networks are employed [55]. These specialized architectures facilitate the integration of inputs from

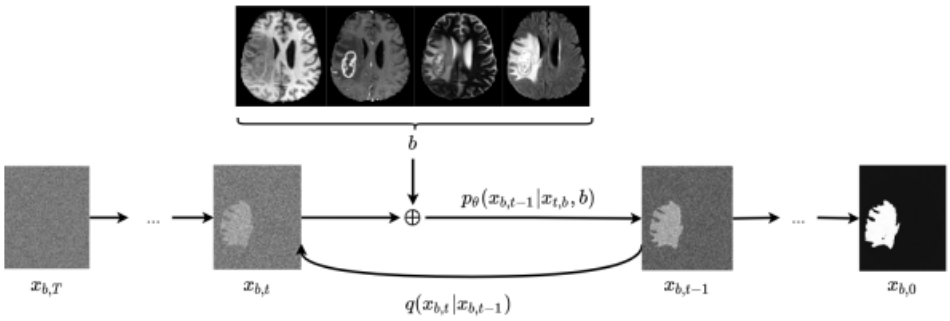

Fig. 2.4. Schematic representation of a DDPM for segmentation that uses concatenation to incorporate an anatomical prior [13].

different modalities (e.g., images, text, numerical data), allowing the DDPM to learn complex interactions between heterogeneous sources of information. By leveraging complementary data types, multi-modal fusion enhances the richness and accuracy of generated samples.

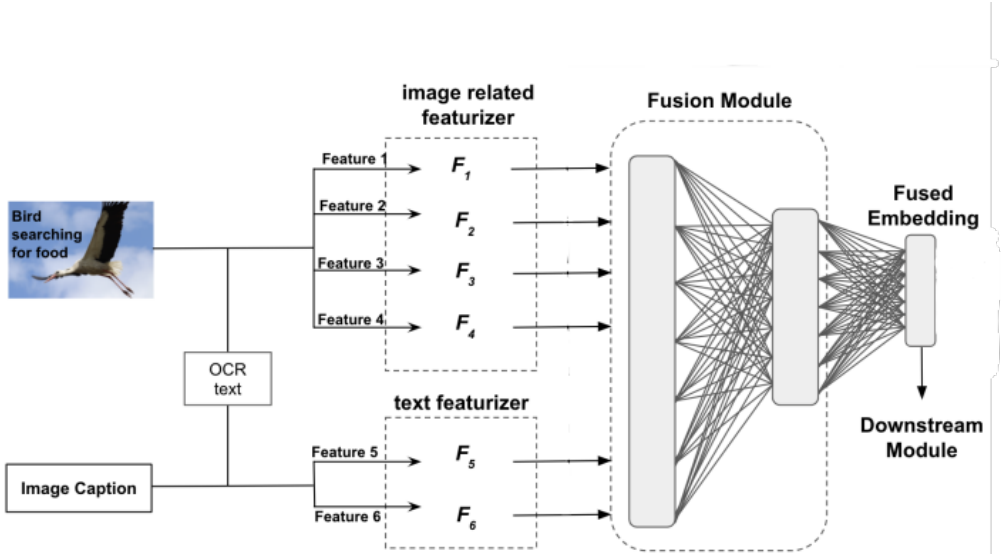

Fig. 2.5. Schematic representation of multi-modal fusion using deep neural networks. Image related feature and text feature are combined together in order to generate a multi-modal embedding used as context by the diffusion model. [56]

Another possibility involves the usage of the attention mechanisms [53]. These provide a mechanism for selectively focusing on relevant parts of the conditioning data c during the generation process. By assigning weights to different components or features of c, these mechanisms enable the DDPM to dynamically adjust its generation strategy based on the importance of various context elements. This approach improves model flexibility and adaptability to diverse types and scales of conditioning data.

The conditional properties of DDPMs empower these models to leverage additional infor-

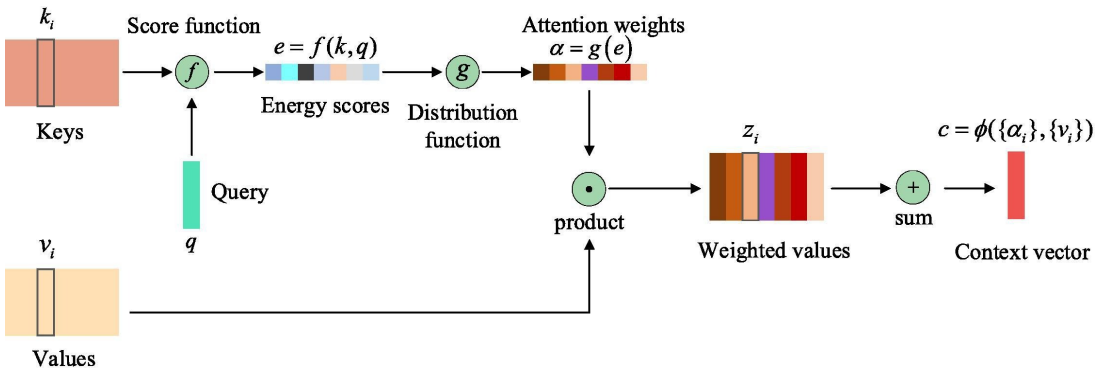

Fig. 2.6. Schematic representation of attention mechanisms with query, keys and values. [57]

mation effectively, enhancing their capability to generate realistic and contextually relevant samples across diverse applications. By integrating conditioning mechanisms, such as anatomical priors in medical imaging or contextual cues in natural language processing, DDPMs contribute to advancing generative modeling capabilities and addressing real-world data challenges.

The aim of this study is to employ Denoising Diffusion Probabilistic Models for the segmentation of the choroid plexus in MRI scans. The model is trained to synthesize segmentation maps starting from Gaussian random noise. In this context, it is crucial to use the conditional processes of DDPMs to introduce anatomical priors during the sampling phase. The goal is to generate a segmentation map specific to a given MRI image, rather than a random segmentation map. By conditioning on the MRI scan, the model can extract anatomical information that serves as constraints during the sampling phase, ensuring the resulting segmentation mask is tailored to the specific image.

2.1.2 Denoising Diffusion Implicit Models

Denoising Diffusion Implicit Models are an advancement of Denoising Diffusion Probabilistic Models, designed to overcome the high computational cost associated with the latter. One of the main disadvantages of DDPMs is the lengthy sampling procedure. Given T as the number of time steps used during the model's training, the reverse process in DDPMs, which is still a Markov chain, requires iterating T times, making the sampling process computationally intensive and time-consuming.

As discussed in the previous section, the general idea of diffusion models is to generate a series of noisy images x_0, x_1,..., x_T from an input image x by adding small amounts of noise over many time steps T. Then, a U-Net-like network is trained to predict x_{t-1} from x_t, for any time

step $t \in 1,...,T$.
In DDPMs, the noise prediction can be parametrized and rewritten using:

$$\sigma_t = \sqrt{\frac{1-\overline{\alpha}_{t-1}}{1-\overline{\alpha}_t}}\sqrt{\frac{1-\overline{\alpha}_t}{\overline{\alpha}_{t-1}}}$$

This results in the update rule:

$$x_{t-1} = \sqrt{\overline{\alpha}_{t-1}}\left(\frac{x_t - \sqrt{1-\overline{\alpha}_t}\epsilon_\theta(x_t,t)}{\sqrt{\overline{\alpha}_t}}\right) + \sqrt{1-\overline{\alpha}_{t-1}-\sigma_t^2}\epsilon_\theta(x_t,t) + \sigma_t z$$

where $z \sim N(0,\mathbf{I})$ represents a stochastic element presents in each sampling steps. In DDIMs however, σ_t is set to 0, resulting in an implict, non-Markovian structure that allows for deterministic generation and improved sample quality [16]. This update rule allows for a more direct and efficient denoising process, significantly reducing the number of steps required for high-quality sample generation.

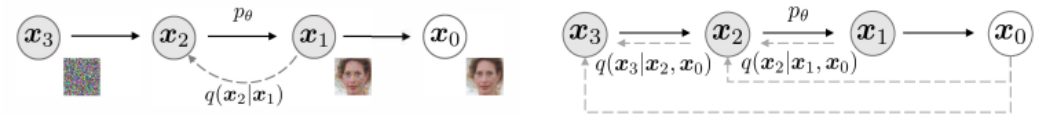

Fig. 2.7. Schematic representations of Denoising Diffusion Probabilistic Models (left) and Denoising Diffusion Implicit Models. [19]

A fascinating aspect of DDPMs is that the objective functions used can learn not only a generative process for the Markovian inference process but also generative processes for various non-Markovian processes parameterized by σ_t. Different choices of σ_t result in different generative processes, all while using the same model, thus eliminating the need for re-training. This means we can use pre-trained DDPMs and set $\sigma_t = 0$ to implement the deterministic sampling procedure.
One of the key advantages of DDIMs is their ability to perform accelerated sampling [19]. Traditional DDPMs require a large number of diffusion steps to generate high-quality samples, which can be computationally intensive. DDIMs, however, can achieve comparable or even superior sample quality with significantly fewer steps. This accelerated sampling process leverages the deterministic nature of the implicit sampling method. By carefully choosing the noise schedule, which influences the α_t values, and the number of diffusion steps, DDIMs can maintain the quality of the generated samples while reducing the computational load. This is particularly important for applications that require real-time or near-real-time performance.

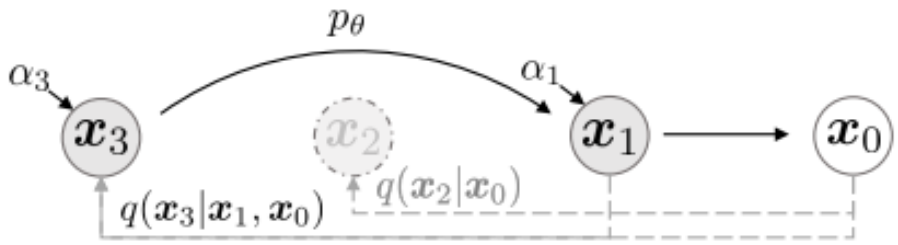

Fig. 2.8. Graphical representation of the accelerated sampling with DDIMs. [19]

The flexibility of the implicit model allows to skip certain steps in the diffusion process without sacrificing sample fidelity. This is achieved by adjusting the update rule to account for larger time intervals between steps, effectively reducing the number of steps needed:

$$x_{t-k} = \sqrt{\overline{\alpha}_{t-k}} \left(\frac{x_t - \sqrt{1-\overline{\alpha}_t}\epsilon_\theta(x_t,t)}{\sqrt{\overline{\alpha}_t}} \right) + \sqrt{1-\overline{\alpha}_{t-k}}\epsilon_\theta(x_t,t)$$

where k represents the step size. Increasing k allows the model can generate samples more quickly while still following the underlying diffusion process.
Moreover, we can rewrite the DDIMs generative process in order to highlight its similarity with the Euler method [19] to solve an ordinary differential equation (ODE):

$$\frac{x_{t-\Delta t}}{\sqrt{\overline{\alpha}_{t-\Delta t}}} = \frac{x_t}{\sqrt{\overline{\alpha}_t}} + \left(\sqrt{\frac{1-\overline{\alpha}_{t-\Delta t}}{\overline{\alpha}_{t-\Delta t}}} - \sqrt{\frac{1-\overline{\alpha}_t}{\overline{\alpha}_t}} \right) \epsilon_\theta(x_t,t)$$

To derive the corresponding ODE, we can reparameterize $(\frac{1-\overline{\alpha}}{\overline{\alpha}})$ with σ and $(x/\sqrt{\overline{\alpha}})$ with $\overline{x}$. In the continuous case, σ and x are function of t, where σ is a non negative, increasing function with $\sigma(0) = 0$. Thus, the ODE is given by:

$$d\overline{x}(t) = \epsilon_\theta \left(\frac{\overline{x}(t)}{\sqrt{\sigma^2+1}}, t \right) d\sigma(t)$$

where the initial condition is $x(T) \sim N(0, \sigma(T))$ for a very large $\sigma(T)$. This suggests that with enough discretization steps, it is possible to reverse the generation process, encoding x_0 to x_T and simulating the reverse of the ODE [16]. Therefore, x_t can be encoded into x_{t+1} as follows:

$$x_{t+1} = x_t + \sqrt{\overline{\alpha}_{t+1}} \left[\left(\sqrt{\frac{1}{\overline{\alpha}_t}} - \sqrt{\frac{1}{\overline{\alpha}_{t+1}}} \right) x_t + \left(\sqrt{\frac{1}{\overline{\alpha}_{t+1}} - 1} - \sqrt{\frac{1}{\overline{\alpha}_t} - 1} \right) \right] \epsilon_\theta(x_t,t)$$

This suggests that unlike DDPMs, DDIMs can be used to obtain encodings of the observations, which might be useful for other downstream application requiring latent representations of the data.
Denoising Diffusion Implicit Models provide a powerful framework for generative modeling, offering a deterministic and efficient approach to sample generation. By introducing an implicit, non-Markovian trajectory, DDIMs achieve high-quality sample generation with fewer steps, making them particularly suitable for applications like medical anomaly detection. The accelerated sampling process further enhances their practicality, allowing for real-time or near-real-time performance.

2.1.3 Patch-Based techniques

Diffusion models have emerged as a powerful class of generative models capable of producing high-quality images by learning to reverse a noise diffusion process. While effective, training these models on high-resolution images or three-dimensional (3D) volumes poses significant computational challenges. These models are computationally expensive and data-intensive, with training costs increasing exponentially with higher resolution and data diversity. To address these challenges, the patch-based training method has been developed, offering a scalable approach to handle large volumes of data efficiently.
The primary motivation for adopting patch-based methods in training diffusion models stems from the need to manage computational resources efficiently. High-resolution images require extensive memory and processing power, making direct training on entire images impractical. The patch-based approach mitigates these issues by focusing on smaller, more manageable image segments.
Patch-based techniques involve dividing each image into smaller patches, typically of a fixed size, such as 64x64 or 128x128 pixels. Each patch is treated as an individual training example. Once extracted, these patches undergo the standard diffusion process, where the model learns to denoise each patch. This approach significantly reduces memory usage and speeds up the training process.
To maintain the contextual integrity of the generated images, patches are often extracted with overlapping regions. This overlap ensures that the model captures the relationship between adjecent patches, leading to more coherent and seamless image generation.
During inference, the trained model generates patches which are then combined to form the complete image. Techniques like blending and smoothing are applied to handle overlapping regions and potential boundary artifacts, ensuring the final image is free from noticeable seams.
An example of patch-based method applied to diffusion model is Patch-DM [58]. Patch-DM trains on patches and uses feature collage to systematically combine partial features of neigh-

boring patches. This method addresses the high computational costs associated with generating high-resolution images, as it is resolution-agnostic.

In Patch-DM, training images are split into patches denoted as $x_0^{(i,j)}$, where (i,j) is the row and column number of the patch. Instead of generating the full image x_0, the model generates $x_0^{(i,j)}$ patches, which are then concatenated to form the complete image. A basic approach involves feeding the noised image patch $x_t^{(i,j)}$ to the diffusion model and outputting the corresponding noise $\epsilon_t^{(i,j)}$. However, this method can lead to severe boundary artifacts since patches do not interact with each other.

To mitigate this, patches can be shifted during each time step. At different time steps, the model processes either the original split patch $x_t^{(i,j)}$ or the shifted split patch $x_t^{'(i,j)}$, a technique called "patch collage". While this reduces boundary artifacts, they can still occur.

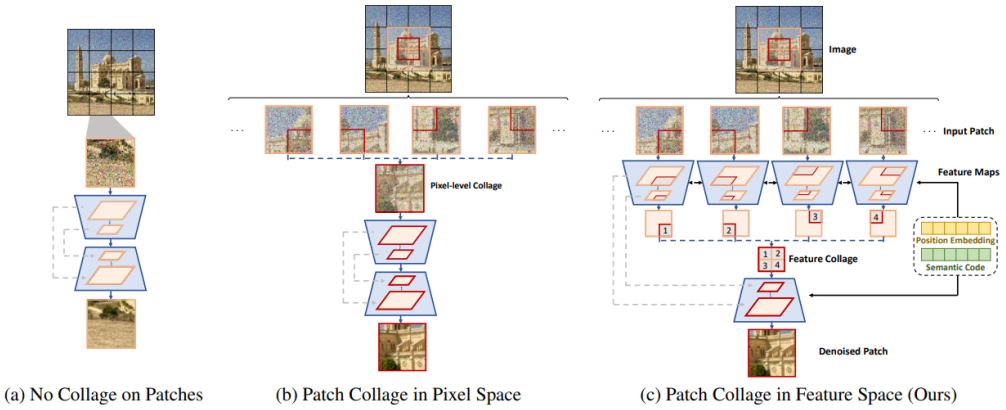

Fig. 2.9. Patch Generation for Image Synthesis. **(a)** Basic patch-wise synthesis by splitting images, causing severe border artifacts. **(b)** Reduces artifacts using shifted windows and patch collage in pixel space. **(c)** Patch-DM patch collage method in feature space, avoiding border artifacts and generating high-quality images. [58]

To further improve, a feature collage mechanism is implemented. Instead of performing "patch collage" in the pixel space, it is performed in the feature space. This allows the patches to be more aware of adjacent features, preventing border artifacts. More formally, the internal feature maps generated by the U-Net encoder are split and merged to generate shift patches, which are then processed by the network's decoder to predict the noise of the shifted patches. Position and semantic embeddings are added to generate more semantically consistent images.

During inference, at each time step t, image x_t is decomposed into patches and fed into the encoder. Before a feature map goes through the decoder, a split and collage operation is applied. The decoder then outputs the predicted noise of the shifted patch, leading to the denoised patches

and finally the denoised image x_{t-1}.

To avoid reconstructing images from generated patches, methods like Patch Diffusion [21] train the diffusion model on patches while sampling the entire image. In Patch Diffusion, by generalizing to an infinite number of noise scales, $T \rightarrow \infty$, the forward diffusion process could be characterized by a stochastic differential equation (SDE) and converted to an ordinary differential equation. The closed form of the reverse SDE is given by:

$$dx = [f(x,t) - g^2(t)\nabla_x \log(p_{\sigma_t}(x)]dt + g(t)dw$$

where $f(\cdot,t)$ is a vector-valued function called the drift coefficient, $g(t)$ is a real-valued function called the diffusion coefficient, dt represents a negative infinitesimal time step, w denotes a standard Brownian motion, and dw can be viewed as infinitesimal white noise. The corresponding ODE of the reverse SDE is name probability flow ODE, given by:

$$dx = [f(x,t) - 0.5g^2(t)\nabla_x \log(p_{\sigma_t}(x)]dt$$

Solving the reverse SDE or the ODE requires knowing the score function $\nabla_x \log(p_{\sigma_t}(x))$, which is generally intractable. Instead, a neural network parameterized score function $s_\theta(x,\sigma_t)$ is learned.

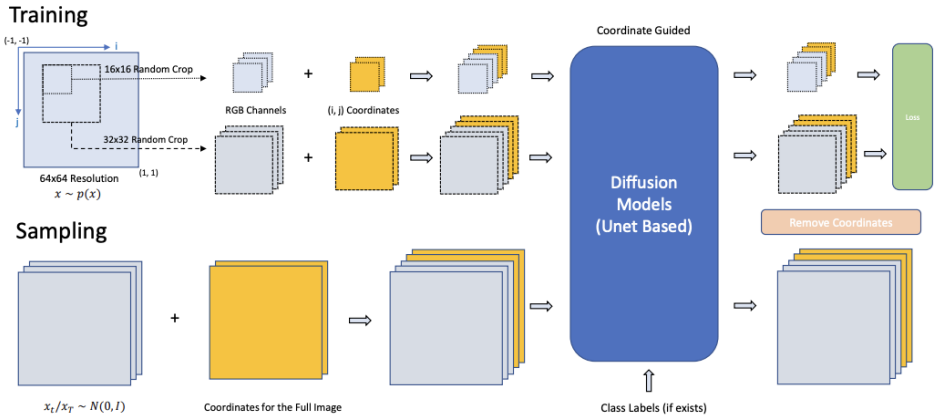

Fig. 2.10. Training and sampling with Patch Diffusion. [21]

Patch Diffusion learns this score function on random-size patches. For each training example, small patches $x_{i,j,s}$ are randomly cropped, where (i,j) are the coordinate of the left-upper corner, and s denotes the patch size. Coordinates are then normalized to $[-1,1]$ with respect to the original image resolution, by setting the upper left corner of the image as $(-1,-1)$ while the bottom right corner as $(1,1)$. Denoising score matching on image patches is conducted with the

corresponding patch locations and sizes as the conditions. The conditioning is obtain through concatenation of the pixel coordinates as additional channels to the original image. When computing the loss, the reconstructed coordinated channels are ignored and only the loss on the image channel is considered. In this way, the conditional score function $s_\theta(x,\sigma_t,i,j,s)$ is defined on each local patch. This approach significantly boosts training speed but challenges the score function to capture global dependencies between local patches. In fact, the conditional score function $s_\theta(x,\sigma_t,i,j,s)$ has only seen local patches and may have not captured the global cross-region dependencies. To resolve this, strategies like random patch sizes and involving a small ratio of full-size images are used. Training patch sizes are sampled form a mixture of small and large patch sizes, with large patches seen as sequences of smaller patches. This helps the score function learns how to unite scores from small patches into a coherent score map. Full-size images are also included during training to ensure the reverse diffusion converges towards the original data distribution.

During sampling, the process can be done globally, without explicitly sampling separate local patches and merge them afterward. Firtsly, full image coordinates are computed and parameterized, and the additional coordinate channels are concatenated with the image sample for conditioning at each step. Finally, the reconstructed coordinate channels are ignored at each reverse iteration, considering only the denoised samples.

An example of application of patch-based methods to diffusion models for the processing of 3D data is given by PatchDDM [54], that reduces resource consumption for 3D diffusion models. As for Patch Diffusion, PatchDDM can be applied to the total volume during inference while the training is performed only on patches. This is extremely important since limitations related to the computational resources only allow the processing of small 3D volumes, which impedes the processing of high-resolution biomedical images.

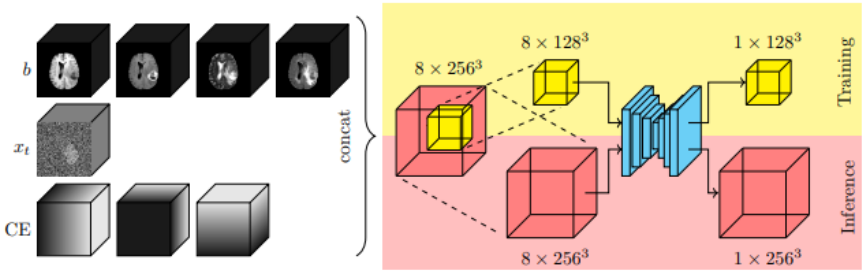

Fig. 2.11. Training and sampling with Patch-DDM. The diffusion model is optimized for memory efficiency and speed by training on coordinate-encoded patches. The input includes noised segmentation x_t, volumes b for segmentation as a condition, and coordinate encoding (CE) for the patches. [54]

The 2D-U-Net-based network architecture is adapted for the application on 3D data. In fact, for 3D data, the attention blocks in the network use disproportionally more memory, which made it infeasible to use on current hardware. For this reason, they are completely removed. The second fundamental change regards the skip connections. Instead of concatenation, averaging is used to reduce computational cost and avoid numerical issues like exploding gradients.
PatchDDM trains on randomly sampled patches, reducing computations and memory usage per iteration.The network is conditioned on patch positions by concatenating Cartesian coordinate grids to the input, where each coordinate is represented by one channel as a linear gradient ranging from -1 to 1 [59]. During inference, the entire volume is processed at once without having to sample patches and reassemble them. This means no boundary and stitching artifacts that can appear in traditional patch-based approaches.
PatchDDM has been successfully applied to segmentation tasks. In order to generate the segmentation of an input image, is necessary to condition the generation of the segmentation mask on that given image. The idea is to follow the method proposed in [13], where the input images are concatenated to every x_t as a condition. Thus, during training the original images are concatenated together with the coordinates embeddings, while during sampling only the image to be segmented is used to condition the process.
The patch-based methods for training diffusion models represent a practical and efficient solution for handling high-resolution image generation. By focusing on smaller patches, these approaches balance computational efficiency with the ability to produce high-quality images, making them a valuable technique in the field of generative modelling.

Chapter 3

Material and methods

3.1 Datasets

The data used for training the diffusion models originate from three different studies, for a total of 205 subjects [60]. For descriptive purposes, the datasets used will be referred to as dataset 1, dataset 2, and dataset 3.
Datasets 1 and 2 were provided by the Multiple Sclerosis Center of the University Hospital of Verona and were acquired prospectively between March 2019 and October 2021. Each participant in the acquisition process provided informed consent prior the starting of the studies, and all subsequent data acquisition and management procedures were conducted in accordance with the Declaration of Helsinki. Furthermore, the protocols were approved by the local Ethics Committee. A detailed description of the datasets is as follows:

- Dataset 1: This dataset comprises 67 subjects (21 males and 46 females) divided into 24 healthy controls (HC) and 43 patients with Relapsing-Remitting Multiple Sclerosis (RRMS). For the HC group, the mean age was 37.3 years, with a standard deviation of 9.5 years, while for the RRMS group, the mean age was 40.9 years with a standard deviation of 9.9 years. The hardware used for acquiring T1-w MRI images was a Philips Achieva TX with an 8-channel head coil (Software version R3.2.3.2). The parameters for the 3D T1-w MPRANGE sequence were as follows: resolution of 1x1x1 mm, SENSE acceleration factor of 2.5, echo time (TE) of 3.7 ms, repetition time (TR) of 8.4 ms, flip angle (FA) of 9°, and a total acquisition time of 4 minutes and 50 seconds.

- Dataset 2: This dataset consists exclusively of RRMS patients, totaling 61 subjects, divided into 13 males and 48 females. The mean age of the participants was 36.7 years, with a standard deviation of 10.1 years. The T1-w MRI images were acquired using a Philips Elition S scanner with a 36-channel head coil (Software version R5.7.2.1). The parame-

ters for the acquisition sequence, identical to the one used previously, were: resolution of 1x1x1 mm, compressed SENSE acceleration factor of 4, TE of 3.7 ms, TR of 8.4 ms, FA of 8°, and a total acquisition time of 3 minutes and 20 seconds.

Dataset 3 was acquired in the United Kingdom as part of the Biomarkers in Depression Study (BIODEP, NIMA consortium, https://www.neuroimmunology.org.uk/). All procedures followed during the study were approved by the local ethics committee in advance and were conducted in accordance with the Declaration of Helsinki. Before participating in the study, all subjects provided informed consent. The BIODEP study was also approved by the NRES Committee East of England Cambridge Central. Dataset 3 includes data from 77 subjects, 26 males and 51 females. Fifty-one of these subjects suffered from depression, with a mean age of 36.2 years and a standard deviation of 7.3 years. The remaining 26 subjects were healthy controls, with a mean age of 37.3 years and a standard deviation of 7.8 years. For each participant, structural MRI images were acquired simultaneously with PET images using a GE SIGNA PET/MR scanner. The parameters for the 3D T1-w Fast SPGR sequence were: resolution of 1x1x1 mm, TE of 2.99 ms, TR of 6.96 ms, FA of 12°, and a total acquisition time of 6 minutes.
In addition to the previously introduced datasets, this study also includes publicly available datasets employed for the pre-training procedures of the diffusion models. The used datasets are listed and described as follows:

- VOC2012: The PASCAL Visual Object Classes 2012 (VOC2012) dataset is a standard benchmark for object recognition and detection in natural images [61]. It contains 11,540 images with 27,450 annotated objects across 20 categories. Each image has an annotation file giving a bounding box and object class label for each object in one of the twenty classes present in the image. Additionally, a subset of 2,913 images is annotated with pixel-wise segmentation of each object by experts. Multiple objects from various classes may appear in a single image. This dataset was originally released as part of the PASCAL Visual Object Class Challenge in 2012 at http://host.robots.ox.ac.uk/pascal/VOC/voc2012/.

- BraTS2021: The Brain Tumor Segmentation 2021 (BraTS2021) dataset focuses on the segmentation of brain tumors from MRI scans and was released as part of the RSNA-ASNR-MICCAI BraTS 2021 challenge [62], [63]. It includes data from 2,000 patients with glioma, encompassing both high-grade (HGG) and low-grade (LGG) tumors, with a total of 8,000 multiparametric MRI scans. The MRI scans, provided in the Neuroimaging Informatics Technology Initiative (NIfTI) format (.nii.gz) for segmentation tasks, consist of T1-weighted (T1-w), post-contrast T1-weighted (T1Gd), T2-weighted (T2-w), and T2 Fluid Attenuated Inversion Recovery (FLAIR) volumes. These scans were acquired from multiple different institutions under standard clinical conditions, but with

different equipment and imaging protocols, resulting in a vastly heterogeneous image quality reflecting diverse clinical practices. Pre-processing steps include co-registration to the same anatomical template, interpolation to a common resolution of 1x1x1 mm and skull-stripping. Annotations for segmentation tasks were done manually by one to four raters, with approval from board-certified neuro-radiologists. These annotations include labels for the GD-enhancing tumor (ET), peritumoral edematous/invaded tissue (ED), and the necrotic tumor core (NCR). Additional infromation could be found at https://www.med.upenn.edu/cbica/brats2021/.

- Figshare Brain Tumor: The Figshare Brain Tumor dataset [64], [65] contains 3,064 T1-weighted contrast-enhanced images from 233 patients, including 708 meningiomas, 1,426 gliomas, and 930 pituitary tumors. The images have an in-plane resolution of 512×512 with a pixel size of 0.49×0.49 mm^2. The slice thickness is 6 mm with a 1 mm gap between slices. Tumor borders were manually delineated by three experienced radiologists. The intensity values at the 1st and 99th percentiles were computed and used to scale intensity values to the [0, 1] range using the min-max method for normalization. The data are organized in Matlab data format (.mat file). Each file stores a struct containing the following field for an image;
  - Label: 1 for meningioma, 2 for glioma and 3 for pituitary tumor.
  - PID: Patient ID.
  - Image: Image data,
  - TumorBorder: A vector storing the coordinates of dicrete points on tumor border.
  - TumorMask. A binary image with 1s indicating the tumor region.

  The data can be found at https://figshare.com/articles/dataset/brain_tumor_dataset/1512427.

- IXI: The IXI dataset consists of brain MRI scans from 600 healthy individuals, including both males and females, aged between 20 and 86 years. The data collections is part of the project ”IXI - Information eXtraction from Images” funded by EPSRC (Engineering and Physical Sciences Research Council) with grant reference GR/S21533/02. The MR image acquisition protocol for each subject includes: T1-w, T2-w, Proton Density weighted (PD-w), magnetic resonance angiography (MRA) and diffusion-weighted images (DWI). The scans were acquired using three different scanners:
  - Ge 1.5T MRI system from the Istittute of Psychiatry.
  - Philips 1.5T MRI system from Guy’s Hospital.

– Philips 3T MRI system from Hammersmith Hospital.

All the images are provided in NIfTI format. More information could be found at https://brain-development.org/ixi-dataset/.

- Medical Segmentation Decathlon - Brain Tumor: This dataset, part of the Medical Segmentation Decathlon challenge [66], focuses on the segmentation of brain tumors in MRI images. It comprises 750 4D volumes of multiparametric MRI data, with 484 volumes allocated for training and 266 for testing. These data are sourced from patients diagnosed with glioblastoma or low-grade glioma. The acquisition sequences used include: T1-w, T1Gd, T2-w, and FLAIR, acquired using different scanners. The spatial dimensions of the images are 240x340 pixels. This datasets focuses on the segmentation of three target region: edema, non-enhanced tumor and enhanced tumor. Segmentations were manually annotated by experts and validated to ensure the accuracy of the ground truth data. All images were transposed (without resampling) to the most approximate right-anterior-superior coordinate frame, and converted to the NIFTI radiological standard. More infromation about the dataset can be found at http://medicaldecathlon.com/

By integrating these comprehensive and diverse datasets, the diffusion models can be pre-trained on a wide range of imaging conditions and scenarios, enhancing their robustness and performance across various applications.

3.2 Preprocessing

The preprocessing steps applied to the datasets varied depending on the source. Images from Datasets 1 and 2 were used as-is without undergoing any preprocessing steps prior to their inclusion in this study. In contrast, images from Dataset 3 were subjected to a rigid-body transformation aligning them to the MNI Talairach ICBM 152 2006c template, facilitated by MIAKAT v4.2.6 software available at http://www.miakat.org/MIAKAT2/index.htm.
The ground truth (GT) manual segmentation for each subject was conducted in the native space of the T1-weighted images. Specifically, segmentation of the choroid plexus (ChP) was confined to the two lateral ventricles due to challenges in reliably distinguishing ChP in the third and fourth ventricles using conventional 3T MRI imaging [67]. For Datasets 1 and 2, ChP segmentation was performed through consensus by a junior and senior neuroradiologist using ITK-snap. Similarly, Dataset 3 underwent ChP segmentation using Analyze software v.12, available at https://analyzedirect.com. The GT data served as the basis for training the diffusion models and as a reference for evaluating their performance during validation and

testing phases.

3.3 Implementation

3.3.1 Basic Training Procedure

This part of the study implements a traditional diffusion model training approach specifically tailored for segmenting brain MRI images, as proposed by [13]. The Python environment utilized key libraries such as MONAI [68], TorchIO [69], and StyleAugmentation [70], along with other standard libraries.

Firstly, Dataset 1 was loaded and split into training, validation and test sets with a 80/10/10 split, resulting in 846 training slices and 105 validation slices. Using MONAI, TorchIO, and StyleAugmentation, a series of transformations were defined to prepare both datasets for the training loop.

For the training set, transformations included restructuring tensors containing images and segmentations to ensure compatibility with MONAI's diffusion model implementations. Adjustments were made to standardize the data to the RAS orientation, and images and segmentations were interpolated to a resolution of 1x1x1 mm (bilinear interpolation for the images and nearest neighbour for the segmentation). A central volume of 128x128x32 voxels was cropped to focus on regions likely to contain the choroid plexus, thereby minimizing empty segmentation masks. Image intensities were then scaled between 0 and 1, with the original intensities ranging from 0 to 99.5. Data augmentation techniques, including random affine transformations, elastic deformations, and various types of noise addition (bias field, blur, spike, motion artifact) with randomized parameters, were applied to simulate common MRI imaging artifacts and enhance the model's robustness. Then, each axial slice of the cropped volume was extracted to obtain two-dimensional training images. Finally, StyleAugmentation was used to alter image styles while preserving semantic content. All the data augmentation techniques mentioned above were applied randomly with a probability of 0.2, except for the random spike noise, which was applied with a probability of 0.1.

The transformations applied for data augmentation are detailed as follows:

- RandAffine: This technique applies a random affine transformation to the image. The transformation parameters are sampled randomly as they are not specified in advance.
- RandElasticDeformation: This method introduces a nonlinear deformation based on control points, with maximum deformations of 5, 5, and 0 along the respective axes. The smoothness of the deformation is regulated by setting the number of control points to 5.

- RandomBiasField: This technique simulates low-frequency artifacts due to inhomogeneities in the scanner's magnetic field. The strength of the artifact is controlled by a coefficient, which was set to 0.5.
- RandomBlur: This method applies a Gaussian blur with a filter of random size to the image. The standard deviations along the axes are set to their default values.
- RandomSpike: This technique introduces random spike artifacts typical of MRI scans. Default parameters are used for this augmentation.
- RandomMotion: This method simulates random motion artifacts in MRI images. The simulated movements are set to 5, and nearest-neighbor interpolation is used.
- StyleAugmentation: This technique alters the image style through randomization of a style transfer network's actions. The style embedding produced by the network can be linearly interpolated with the image's embedding, controlled by a parameter α. A higher α value results in a more significant change in style. The default α value of 0.5 was employed in this work.

For the validation set, similar transformations were applied, excluding data augmentation techniques. With transformations defined, both datasets were processed to create data loaders for subsequent model training and validation, with batch sizes set to 128.
The diffusion model was defined by establishing a deep neural network for noise prediction, a scheduler, and an inferer. The network was based on a modified 2D U-Net architecture, incorporating self-attention mechanisms in the bottleneck and additional layers in the initial levels to obtain temporal embeddings of the timesteps. The model took two inputs—the image and the segmentation map—and outputted the predicted noise added to the segmentation map.
The scheduler defines the noising and denoising processes. During initialization, the number of timesteps used was set to 1000, following recommendations from multiple diffusion model studies. Once the scheduler was defined, it initialized the model's inferer for sampling by reversing the forward diffusion process used to add the noise. Before training, the optimizer was set to Adam with a learning rate of 2.5×10^{-5}, as recommended by [13], and Mean Squared Error (MSE) loss was selected to minimize the error between predicted and actual noise added to images.
Training spanned over 6000 epochs, where each iteration involved random timestep values for noise addition via the scheduler. Anatomical priors were introduced by concatenating images with noisy segmentations, serving as model inputs alongside corresponding timesteps. The model then predicted the noise added to the segmentation mask. Loss calculation, backpropagation with Adam optimizer, and model parameter updates followed.

Algorithm 1 Training Procedure

Require: $[\mathbf{b}, \mathbf{x_{b,0}}]$, the training dataset containing images and corresponding segmentation masks

Require: $f_\theta(\cdot)$, the 2D U-Net-based model parameterized by θ

for $epoch \leftarrow 1$ to 6000 **do**
  for $b, x_{b,0} \leftarrow [\mathbf{b}, \mathbf{x_{b,0}}]$ **do**
    sample $t \sim \mathcal{U}(0{,}1000)$
    sample $\epsilon \sim \mathcal{N}(0, I)$
    $x_{b,t} \leftarrow \sqrt{\overline{\alpha}_t} x_{b,0} + \sqrt{1 - \overline{\alpha}_t}\epsilon$
    $X_t \leftarrow b \oplus x_{b,t}$
    $\epsilon_\theta(X_t{,}t) \leftarrow f_\theta(X_t, t)$
    $L \leftarrow \mathbb{E}_{t,x_{b,0},\epsilon}[\|\epsilon - \epsilon_\theta(X_t, t)\|^2]$
    Update model parameters θ using L and Adam
  end for
end for

Every 50 epochs, validation was performed. The model was switched to validation mode, and the same process described for the training phase was applied to the validation data until the calculation of the loss function. After a burn-in period of 10 validations, early stopping was checked to prevent overfitting. Two methods were considered: comparing the current validation loss with that obtained 10 validation cycles earlier, or using a moving average window of 5 samples with an overlap of 3 samples and comparing the mean validation loss in the two windows. In both cases, if for 10 consecutive cycles the current validation loss or the current window's average loss exceeded the validation loss obtained 10 validation cycles earlier or the previous window's average loss, early stopping was triggered. These methods focused on the underlying trend to avoid issues from the oscillatory nature of validation loss, which could trigger early stopping even without an increasing trend. Correct implementation of early stopping during validation is crucial for diffusion models, as they are prone to overfitting, causing a lack of diversity during the sampling process.

Once model training was completed, its performance was evaluated on unseen images from the test set, which consisted exclusively of data from Dataset 1, totaling 105 samples.

Post-training, the same transformations used on the validation set were applied to the test data to ensure congruence with the images used for model training. Since the sampling process in Denoising Diffusion Probabilistic Models has a random component, applying the model multiple times (n times) to a given image results in n different segmentation maps. As demonstrated in [13], these n sampled maps can be used to generate an average segmentation map, improving segmentation quality, and a variability map, highlighting regions where the model exhibits greater uncertainty. Therefore, by using a value of n equal to 5, five segmentation maps were

generated for each image by applying the model. Starting from two-dimensional pure Gaussian noise, denoising was performed over 1000 steps, as initially set in the scheduler definition. For each timestep t from 1000 to 0, the concatenation of the noisy sample and one slice of the image was given as input to the model for the prediction of the noise added at timestep t. Then, through the inferer, the predicted noise was removed to obtain a slightly denoised version of the noisy sample. By proceeding this way for each timestep t, a segmentation map was generated by the model. Finally, after generating segmentation masks for each slice of the volume, these individual masks were stacked together to reconstruct the entire segmentation volume.
Once the ensemble of segmentations for each image was obtained, the average map and the uncertainty map were calculated. The Dice Score was used to evaluate the model. This score was calculated for each segmentation volume obtained, as well as for the average volume, to assess whether the ensemble improves the segmentation of the choroid plexus. The Dice Score for the average can also be evaluated by varying the value of n to find the optimal value that maximizes segmentation quality. Finally, the uncertainty map can be plotted alongside the average map to visualize where the model has the most difficulty in segmenting the images.

Algorithm 2 Sampling Procedure

Require: b, the original brain MRI
Ensure: $x_{b,0}$, the predicted segmentation mask
1: sample $x_{b,T} \sim \mathcal{N}(0, I)$
2: **for** $t \leftarrow 1000$ to 1 **do**
3: $\quad X_t \leftarrow b \oplus x_{b,t}$
4: $\quad x_{b,t-1} \leftarrow \frac{1}{\sqrt{\alpha_t}} \left(x_{b,t} - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(X_t, t) \right) + \beta_t z$, with $z \sim \mathcal{N}(0, I)$
5: **end for**

3.3.2 Pre-training VOC

This part of the study investigates the potential of pre-training on non-biomedical images to enhance the performance of the diffusion model. Pre-training tasks utilized the public VOC2012 dataset for segmentation, retaining the architecture, scheduler, inferer, and loss function from prior implementation with notable adjustments. The first difference lies in the learning rate for the Adam optimizer during pre-training, set to an higher value of 5×10^{-4}.
The transformations applied to the VOC2012 dataset differ from those applied to the brain MRI dataset. The dataset was divided into training and validation sets with a 90/10 split, resulting in 2622 training samples and 291 validation samples. Transformations adapted dataset structures to align with MONAI's diffusion model requirements. RGB images were converted

to grayscale, resized uniformly to 128x128 pixels, and intensity from 0 to 99.5 were rescaled to [0,1]. Only the images underwent affine transformations with random parameters for data augmentation. Since the segmentation masks can contain multiple classes and values other than 0 and 1, all classes were unified into a single one.
Validation set transformations mirrored those of the training set, omitting affine transformations. Data loaders were configured with a batch size of 128 after transformations were applied to both datasets.
Pre-training spanned 2000 epochs with validation and early stopping checks every 50 epochs, following a similar methodology as previously described, with the only difference being the use of natural images.
After completing the pre-training process, fine-tuning was performed to specialize the model in the task of choroid plexus segmentation from brain MRI images. This phase was conducted using data from Dataset 1. The fine-tuning process was executed in the same manner as the training process described in the previous section, with the number of epochs reduced to 2450 due to the prior pre-training phase where the model learned generic information from natural images. Additionally, the learning rate was reverted to 2.5×10^{-5}.
Upon completion of the fine-tuning process, the model's performance was evaluated in terms of Dice Score. As in the previous case, the average map was also assessed to determine if it improved segmentation performance, together with the uncertainty map to highlight the regions where the model showed greater difficulty in segmenting the choroid plexus.

3.3.3 Pre-training encoder

The objective of this study segment is to assess the efficacy of utilizing self-supervised learning methods with unlabeled biomedical data for pre-training the encoder of the network employed in the diffusion model, aiming to enhance performance in the subsequent choroid plexus segmentation task. The images used for pre-training come from the publicly available medical datasets listed before, excluding the IXI dataset, along with the combined training sections from datasets 1 and 2.
First, all data were loaded without considering the segmentation maps, simulating a condition where ample unlabeled data is available alongside a small fraction of labeled data. To exploit all the unlabeled data, self-supervised learning algorithms were implemented to allow the model to learn latent representations of the data, which can later be leveraged for discriminative tasks. The total number of data points is 14,635. The data was then split into training and validation sets, with percentages of 90% and 10% respectively, resulting in a total of 13,172 training data and 1,463 validation data. Subsequently, transformations were defined for both datasets.

For the training set, transformations mirrored those used during the fine-tuning phase described earlier, up to the addition of different types of noise. At this point, two copies of each image were generated. Since the pretext task will be a reconstruction task, it was necessary to apply random holes and shuffle image regions with random parameters, different for each copy of the image, so that they become unrecognizable by a simple pixel comparison. This process allows the network to learn to reconstruct the original image and generates suitable representations in the latent space that can be used later for the segmentation task. Similar transformations were applied to the validation set.
The transformations applied to generate the model's input are detailed as follows:

- RandCoarseDropoutd 1: This transformation randomly generates eight holes in the image, with each hole having a spatial dimension sampled uniformly from a range of 5 to 32 pixels. These holes are filled with null (zero) values.
- RandCoarseDropoutd 2: This variant generates eight holes with spatial dimensions uniformly sampled from a range of 20 to 64 pixels. Unlike the first variant, these holes are filled with random values.
- RandCoarseShuffled: This method randomly selects ten regions in the image, each with a dimension of 8 pixels. Within each selected region, the pixels are shuffled with a probability of 0.8.

For each copy of the original image, one of the first two transformations is applied. The selection between these two transformations is made randomly, with each having an equal probability of 0.5.
The network used for pre-training has a different architecture compared to the previous sections. It is still based on the U-Net architecture but without the initial layer for time embeddings, as it is not necessary. This resulted in a more generic network, applicable to various tasks, including reconstruction, but which still shares a significant portion of weights with the original network, allowing them to be pre-trained. In addition, the network output included reconstructed images from modified inputs and their respective latent representations. During pre-training, only the encoder part of the network was retained for subsequent fine-tuning. As in previous cases, the Adam optimizer was chosen with a learning rate of 1×10^{-4} while the batch size was reduced to 32 for computational reason.

The pre-training process was carried out over 2000 epochs, with validation and early stopping checks every 50 epochs. For each epoch, the two modified versions of each image were given as input to the model, generating both the latent representations of the two images and the reconstructed images. To achieve the training goal, a hybrid loss was defined [71], consisting

Algorithm 3 Pre-training Procedure

Require: $[\mathbf{b}, \mathbf{b_1}, \mathbf{b_2}]$, the training dataset containing original images and their modified versions
Require: $g_\Theta(\cdot)$, the pre-training model parameterzied by Θ

for $epoch \leftarrow 1$ to 2000 **do**
  for $b, b_1, b_2 \leftarrow [\mathbf{b}, \mathbf{b_1}, \mathbf{b_2}]$ **do**
    $x_1, z_1 \leftarrow g_\Theta(b_1)$
    $x_2, z_2 \leftarrow g_\Theta(b_2)$
    $L_{recon} \leftarrow \mathbb{E}_b[\|b - x_1\|]$
    $L_{CL} \leftarrow L_{SimCLR}(z_1, z_2)$
    $L \leftarrow L_{recon} + L_{recon} \cdot L_{CL}$
    Update model parameters Θ using L and Adam
  end for
end for

of Mean Absolute Error (L1 loss) for the reconstruction task and SimCLR contrastive loss, with a temperature value of 0.05, to increase the similarity of the latent representations generated by similar images. In particular, the L1 loss was calculated using the reconstructed output from one of the modified version and the original image, while the constrastive loss was calculated using the two latent representations of both the modified version of each image. The hybrid loss was then calculated, and the model's parameters were updated by backpropagation with the Adam optimizer.

Once the pre-training process was completed, the fine-tuning phase was carried out, using Dataset 1. The model was redefined in the same way as in the previous sections, but with the weights of the encoder initialized based on the pre-trained network's weights. In order to do so, was necessary to check which weights are in common between the encoders of the two networks. The training loop followed the same steps described in previous sections. At the end of the fine-tuning phase, the model's performance was evaluated in terms of Dice Score. Additionally, the average map and uncertainty map were assessed to evaluate the quality of the final segmentation.

3.3.4 Diffusion Self-supervised learning

Similarly to preceding sections, this part of the study explores the potential of self-supervised learning with unlabeled biomedical data to bolster the model's performance in choroid plexus segmentation. To this end, the same datasets used in the implementation described in the previous subsection will be employed.

Although self-supervised learning techniques were again used during the pre-training phase, there are significant differences compared to the previous approach. First and foremost, the

pretext task will be a denoising task. This is to assess the feasibility of using diffusion process for the pre-training of non-diffusion models. Therefore, the network used during pre-training remains a two-dimensional U-Net architecture adapted to the context of diffusion processes. However, in this case, a single input to the network was chosen—the image itself—since the data available for self-supervised learning ware assumed to be unlabeled. The scheduler, inferer, and dataset transformations mirrored those in the preceding sections, with training and validation sets identical in size and makeup. The optimizer used is Adam with a learning rate of 1×10^{-4}.

Another key departure from previous approaches involves the loss function used during diffusion-based training. In the previous implementations, an MSE loss was used, applied to the predicted error and ground truth, without any weighting of its components based on the timesteps. However, in this case, the goal is to learn effective visual representations of the images using diffusion in a self-supervised learning process. This can be achieved by focusing more on the content and disregarding insignificant or trivial details of the image. This is not guaranteed if an uniform weighting is used for all timesteps. Therefore, a perception-prioritized (P2) weighting was chosen [72], which provides a good inductive bias for learning rich visual representations by increasing the weighting in the coarse and content phases and suppressing the weights in the initial timesteps, i.e., when the signal-to-noise ratio (SNR) is high. The formulation of the P2 loss is as follows:

$$L_t \;=\; \mathbb{E}_{x_0,\epsilon}[\frac{1}{(k+SNR(t))^{\gamma}}\|\epsilon-\epsilon_\theta(x_t,t)\|^2],$$

$$L_{P2} \;=\; \sum_{t\geq 1} L_t$$

where γ is an hyperparameter that controls the strength of the weight reduction on learning imperceptible details, while k is an hyperparameter that prevents the weights from exploding for extremely low SNR values and determines the sharpness of the weighting scheme. Both were empirically setted to 1. The signal-to-noise ratio $SNR(t)$ of the noisy sample x_t was obtained by taking the ratio of the squares of the coefficients of x_0 and ϵ, corresponding to the signal and noise variances, respectively:

$$SNR(t) \;=\; \frac{\overline{\alpha_t}}{1-\overline{\alpha_t}}$$

Following these preparations, the pre-training process mirrored previous implementations, incorporating the same setup for training the diffusion model. The only difference is the need to extract the $\overline{\alpha_t}$ values from the scheduler necessary to calculate the weights for the P2 loss.

Fine-tuning, conducted solely on Dataset 1, maintained consistency in data transformations and

Algorithm 4 Self Supervised Pre-Training Procedure

Require: $\mathbf{b_0}$, the training dataset containing images
Require: $f_\theta(\cdot)$, the 2D U-Net-based model parameterized by θ
for $epoch \leftarrow 1$ to 2000 **do**
  for $b_0 \leftarrow \mathbf{b_0}$ **do**
    sample $t \sim \mathcal{U}(0{,}1000)$
    sample $\epsilon \sim \mathcal{N}(0, I)$
    $b_t \leftarrow \sqrt{\overline{\alpha}_t} b_0 + \sqrt{1 - \overline{\alpha}_t}\epsilon$
    $\epsilon_\theta(b_t{,}t) \leftarrow f_\theta(b_t{,}t)$
    $\overline{\alpha}_t \leftarrow \prod_{s=1}^{t} \alpha_s$
    $SNR(t) \leftarrow \frac{\overline{\alpha}_t}{1-\overline{\alpha}_t}$
    $L_{P2} \leftarrow \mathbb{E}_{b_0,\epsilon}[\frac{1}{(k+SNR(t))^\gamma}\|\epsilon - \epsilon_\theta(b_t{,}t)\|^2]$
    Update model parameters θ using L_{P2} and Adam
  end for
end for

optimizer choice but employed each axial slice instead of randomly extracting one per data and utilized a lower learning rate of 2.5×10^{-5}. As in the previous implementations, the utilized loss function was the mean square error loss and the fine-tuning loop was carried out over 2600 epochs.

Algorithm 5 Fine-Tuning Procedure

Require: $[\mathbf{b}{,}\mathbf{x_b}]$, the training dataset containing images and corresponding segmentation masks
Require: $f_\theta(\cdot)$, the 2D U-Net-based pre-trained model parameterized by θ
for $epoch \leftarrow 1$ to 2600 **do**
  for $b, x_b \leftarrow [\mathbf{b}, \mathbf{x_b}]$ **do**
    set $t \leftarrow 0$
    $\hat{x_b} \leftarrow f_\theta(b{,}t)$
    $L \leftarrow \mathbb{E}_{t,x_b}[\|x_b - \hat{x_b}\|^2]$
    Update model parameters θ using L and Adam
  end for
end for

A substantial difference directly concerns the procedure followed in the training loop. Here, the task used for fine-tuning is not a denoising task, as the goal is to develop a model that does not use diffusion processes for segmentation but only during pre-training. Instead of defining a new network that shares weights with the pre-trained one and only loading those, it was decided to use the same architecture with the pre-trained weights. To this end, the timestep value provided as input with the images to be segmented must be setted to zero, as no noise was added. The prediction was used together with the ground truth masks for the calculation of the aforementioned loss. Through backpropagation and Adam, the loss was used to update the

parameters. In this way, at the end of the training loop, a model is obtained that takes as input one slice of a brain MRI image and outputs the segmentation of the choroid plexus without implementing any diffusion, but making good use of the visual representations learned during pre-training via the diffusion process.
Since the random component in generating segmentation maps is lost, the model was evaluated solely in terms of Dice Score calculated on the single segmentations obtainable for each image. It is necessary to remember that when applying the model, it is not sufficient to provide only the image to be segmented as input, but a null timestep value is also required.

3.3.5 Patch-Based Diffusion

The objective of this part of the study is to verify the feasibility of a diffusion model for the 3D segmentation of the choroid plexus. To achieve this, patch-based approaches were implemented along with other techniques to enhance the performance in segmenting 3D images.
Firstly, Dataset 1 was loaded and split into training, validation, and test sets with an 80/10/10 split, resulting in 846 training samples and 105 validation samples. Using MONAI, TorchIO, and StyleAugmentation, a series of transformations were defined to prepare both datasets for the training loop.
For the training set, the same transformations used in the basic training procedure described earlier were applied, up to the addition of noise. A notable difference involves the central volume cropping: instead of extracting a central volume of 128x128x32 voxels, a central volume of 128x128x128 voxels was used. This is because the entire 3D volume was employed for training. The entire volume underwent a modified version of StyleAugmentation that operates on 3D data, altering the image style of each slice within the same volume consistently. Specifically, this version of StyleAugmentation defines the new style using one slice and then applies it to all slices of the volume, ensuring consistency for the 3D training loop.
Following the process suggested by [54], a custom transformation was applied to extract the normalized voxel coordinates into three tensors, one per dimension, and concatenate these with the images. Additionally, a custom random patch extractor was defined. Similar to PatchDDM, this transformation extracts random patches from the images by selecting the coordinates of the center from a probability distribution given by $p = \mathcal{U}[-1/3,1/3] + \mathcal{U}[-2/3,2/3]$, centered in the central voxel. Moreover, following [21], the patch size was randomly sampled to allow different sizes, which are beneficial for learning the relationships between patches. Specifically, the sizes could be 32x32x32, 64x64x64, or 128x128x128. Allowing for a small fraction of full-resolution images is fundamental to help the model learn how to combine different patches, ensuring the reverse diffusion converges toward the original data distribution.

The random patch extractor sampled five patches from each image to obtain multiple views of the 3D volume and reduce the loss of information.
For the validation set, the same transformations were applied, excluding data augmentation techniques. With transformations defined, both datasets were processed to create data loaders for subsequent model training and validation, with batch sizes set to 4. Each batch was defined by patches of the same size to avoid issues during training.
The scheduler, inferer, and optimizers were defined as in the earlier section, while the model underwent some modifications. Following [54], the U-Net architecture used previously was modified by removing the attention mechanisms at each level and substituting the concatenation in the residual skips with an averaging operation. Both changes were made for computational reasons, allowing for a deeper structure compared to the one used with two-dimensional data. The network received as input three-dimensional data with five channels, given by the image, the corresponding label, and the three tensors containing the Cartesian coordinates of the voxels.
The training loop was executed as in the earlier section for 2650 epochs, using the MSE loss to update the model parameters. Every 50 epochs, validation was performed, and after a burn-in period of 10 validations, early stopping was checked to prevent overfitting. Once the model training was completed, its performance was evaluated on unseen images from the test set.
Post-training, the same transformations used on the validation set were applied to the test data, except for the random patch extraction. As proposed in both [54] and [21], the aim is to train the model on patches to make the process faster and then apply the model to the entire 3D volume to obtain the complete segmentation, avoiding the need to reconstruct images from patches. The model's performance was evaluated in terms of Dice Score. Additionally, the model was applied multiple times to the same 3D image, allowing for the definition of the mean map and the uncertainty map. These were assessed to evaluate the quality of the final segmentation.

3.3.6 Three-dimensional Diffusion Self-supervised learning

Building on the methodology used in the two-dimensional case, this section aims to determine whether pre-training the model using self-supervised learning with unlabeled biomedical images can enhance its performance. For this purpose, the same datasets from the two-dimensional analysis are utilized, except the Figshare dataset, which consists only of two-dimensional images, is replaced by the IXI dataset. This substitution, preferred over the solely removal, ensures a comparable volume of data to that used previously.
Initially, all data were loaded without segmentation maps, simulating a scenario with abundant unlabeled data and a smaller fraction of labeled data. The total number of data points is 15,045,

split into training and validation sets with a 90% to 10% ratio, resulting in 13,441 training data points and 1,604 validation data points. Transformations were then defined for both datasets.
The same transformations described in the previous section were applied to both training and validation data. In fact, the pretext task involved denoising, aiming to use the diffusion process for pre-training a non-diffusion model. Therefore, the model used for pre-training was the same modified three-dimensional U-Net-like network from the previous section, with the only difference being the absence of labels and the focus on image denoising. The network expected four input channels: the image and three tensors containing the Cartesian coordinates of the voxels. Finally, the scheduler and inferer mirrored those in the preceding section, while the Adam optimizer was used with a learning rate of 1×10^{-4}.
As in the two-dimensional case, uniform weighting of the MSE loss was replaced by perception-prioritized weighting. The resulting P2 loss enhances learning of rich visual representations by emphasizing coarse and content phases and de-emphasizing initial timesteps. The hyperparameters were consistent with those used in the two-dimensional scenario.
The pre-training loop was executed for 2000 epochs. Every 50 epochs, validation was performed, and after 10 validations, early stopping was checked to prevent overfitting. During each epoch, images and corresponding Cartesian coordinates were loaded and separated into different tensors. Random noise was added to the images for a random number of timesteps uniformly distributed between 0 and 1000. The Cartesian coordinates were then concatenated with the images into a single tensor and fed into the model. The noise prediction and ground truth noise were used to compute the P2 weighted loss, which was used in backpropagation with the Adam optimizer to update the parameters.
Upon completing pre-training, images from the training section of Dataset 1 were loaded and split into training, validation, and test sets for fine-tuning. The fine-tuning phase maintained consistency in data transformations and optimizer choice, but labels were loaded, and the learning rate was set to 2.5×10^{-5}. A MSE loss function was used and the fine-tuning loop was carried over 100 epochs.
During the fine-tuning training loop, the task was not denoising, as the goal was to develop a model that uses the diffusion process only during pre-training, not for segmentation. Instead of extracting the pre-trained weights and updating those of a new network, the entire pre-trained network was used. To this end, the timesteps value provided as input with the images to be segmented must be set to zero, as no noise was added. Then, the prediction, along with the ground truth segmentations, was used to calculate the aforementioned loss, which was finally used to update the parameters through backpropagation with Adam.
At the end of the training loop, the resulting model could take a brain MRI volume as input and output the corresponding segmentation of the choroid plexus without implementing any

diffusion, effectively leveraging the visual representations learned during pre-training.
Since the random component during sampling is eliminated, the model was evaluated solely based on the Dice Score calculated from the single segmentation obtainable per image. It is important to remember that when applying the model, both the image to be segmented and a null timestep value must be provided as input. The evaluation images were taken from the initially defined test set and underwent the same transformations as the validation set, except for random patch extraction. The training process was accelerated by training on patches, but then the model was applied to the entire 3D volume to obtain the complete segmentation, thus avoiding errors that could arise from reconstructing the image from patches.

Chapter 4

Results

4.1 Two-Dimensional Segmentation

During the evaluation phase, all images form the test set were processed using the trained models according to the Algorithm 2. For the diffusion SSL model, a single segmentation mask was produced per image. In contrast, for other models, n segmentation maps were generated for each image to create an ensemble. These masks were then thresholded at 0.5 to yield binary segmentations. The performance of the models were evaluated in terms of Dice Score, which was calculated between the predicted maps and the corresponding ground truth annotations for each image.

Table 4.1 reports the average Dice scores obtained from the first sample generated for each image. For the diffusion SSL model, the reported value is the average Dice score from the single mask produced per image. These results were compared with those reported by Wolleb et al. [13]. For this comparison, the model proposed by Wolleb et al., available at https://github.com/JuliaWolleb/Diffusion-based-Segmentation, was adapted for the specific task of choroid plexus segmentation and trained using Dataset 1. The same procedure outlined in Algorithm 2 was employed to generate the ensemble for each image, and the average Dice score for the first sample was calculated.

For models generating an ensemble of masks, the mean segmentation map was calculated by averaging, pixel-wise, the n segmentation maps for each image. These mean maps were then thresholded at 0.5 to produce binary segmentations, and the corresponding Dice scores were calculated and reported in Table 4.1. This approach aimed to determine whether the ensemble of segmentations could enhance performance.

In addition to the average Dice scores calculated and reported in Table 4.1, variance values were also computed to provide insight into the distribution of Dice scores around the mean. To facilitate a more direct visualization of this distribution, histograms and boxplots of the

performance metric values were also plotted. These visualizations not only allow for a clearer view of the distribution of scores but also help identify potential outliers, offering a deeper understanding of the mean and variance values obtained across different models. The boxplots for each model applied to the three different testing set are shown in Fig. 4.1, while the variance values are reported in Table 4.1 alongside the average values, enclosed in square brackets.

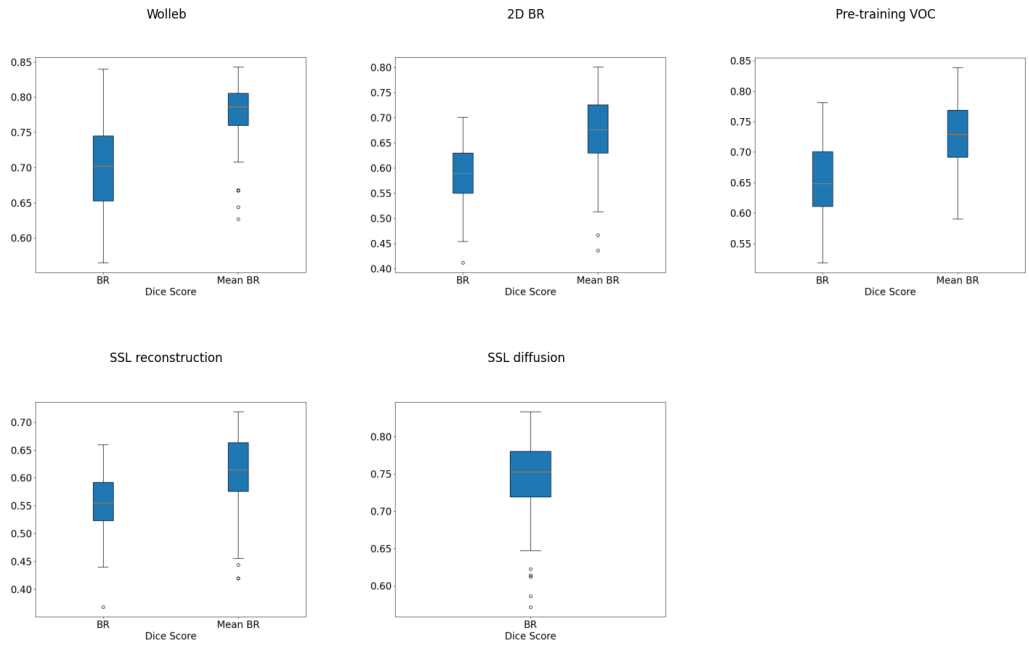

Fig. 4.1. Boxplots of mean Dice score and ensemble Dice score for the two-dimensional models.

To determine the optimal number of samples for the ensembles, we plotted the Dice Score of the mean map as a function of the ensemble size n for the baseline model. For each image, the mean map was evaluated across different ensemble sizes, and the corresponding Dice Score was calculated. The goal was to identify the ensemble size that, on average, provides the best performance improvement across all images in the test set. To achieve this, the Dice Score curves obtained for each image were averaged element-wise. The optimal ensemble size was then chosen by identifying the point where this averaged curve maximizes the performance metric. In making this choice, also the trade-off between performance improvements and temporal resources was considered. The curve calculated for the baseline model is reported in Fig. 4.2.

For models generating an ensemble of masks, an uncertainty map was calculated from twenty samples. This process involved using the available segmentation maps for each image to compute the pixel-wise variance. The pixel-wise variance serves as a measure of uncertainty,

Method	Mean Dice score	Ensemble Dice score
Baseline	0.700 [0.004]	0.780 [0.002]
2D BR	0.587 [0.003]	0.668 [0.006]
Pre-training VOC	0.656 [0.003]	0.727 [0.003]
SSL reconstruction	0.556 [0.003]	0.612 [0.005]
SSL Diffusion	0.745 [0.002]	-

Table 4.1: Segmentation scores of my 2D methods and baseline in terms of Dice score.

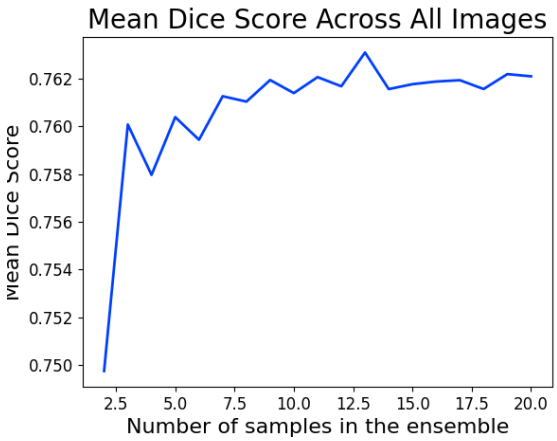

Fig. 4.2. Ensemble performance with respect to the number of samples.

highlighting areas where the model's predictions were inconsistent. Higher variance indicates regions where the model was less certain about the segmentation boundaries, while lower variance suggests greater confidence.

By analyzing these uncertainty maps alongside the mean segmentation map, we can gain deeper insights into the reliability of the model's predictions. The mean map, representing the average segmentation across the ensemble, provides a central tendency of the model's predictions. When combined with the uncertainty map, it becomes possible to identify regions of the image where the segmentation is most trustworthy and where caution is warranted.

Finally, the total sampling time for generating the ensemble of segmentations was calculated for all models, except for the diffusion SSL, where the sampling time for a single segmentation map was recorded. For models generating an ensemble, the average total sampling time is reported in Table 4.2, alongside the sampling time for diffusion SSL.

To explore the potential for reducing sampling time, the DDIM sampler was also tested. The corresponding total sampling times are included in Table 4.2. Additionally, the mean Dice scores are reported to assess whether the reduction in sampling time maintains high performance.

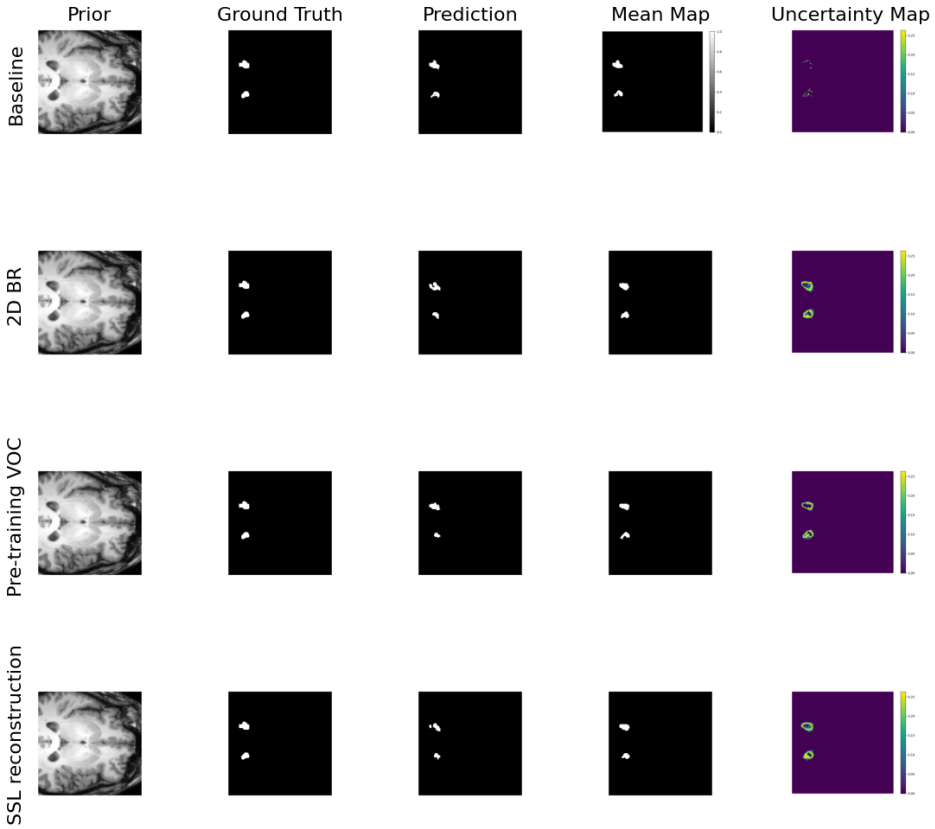

Fig. 4.3. Anatomical prior, ground truth segmentation, first prediction, mean map and uncertainty map for the diffusion-based models.

Method	DDPM		DDIM	
	Mean Sampling Time	Mean Dice Score	Mean Sampling Time	Mean Dice Score
Baseline	15.098	0.700	12.514	0.777
2D BR	15.097	0.475	14.067	0.432
Pre-training VOC	15.909	0.656	13.998	0.547
SSL reconstruction	16.302	0.556	14.968	0.462
SSL diffusion	0.046	0.745	-	-

Table 4.2: Mean sampling times and mean dice scores for both scheduler

To evaluate the generalization capability of the trained models, especially given that data augmentation techniques were introduced in the proposed methodologies compared to the baseline, we applied the different models, including the baseline, to the testing portion of Dataset 2 and Dataset 3. These datasets are drawn from a different population than the one

Method	Dataset 2		Dataset 3	
	Mean Dice Score	Ensemble Dice Score	Mean Dice Score	Ensemble Dice Score
Baseline	0.533 [0.026]	0.655 [0.026]	0.392 [0.034]	0.459 [0.044]
2D BR	0.404 [0.015]	0.474 [0.022]	0.324 [0.004]	0.362 [0.006]
Pre-training VOC	0.503 [0.013]	0.590 [0.026]	0.378 [0.003]	0.433 [0.009]
SSL reconstruction	0.419 [0.012]	0.499 [0.014]	0.205 [0.004]	0.245 [0.005]
SSL Diffusion	0.667 [0.028]	-	0.556 [0.015]	-

Table 4.3: Dice scores of the 2D methods and the baseline over the testing fraction of Dataset 2 and Dataset 3.

used for training, making it ideal for assessing the models' ability to generalize. The same metrics used for evaluating the test fraction of Dataset 1 were employed here, and the results are presented in Table 4.3. By analyzing the Dice Score values, we can evaluate the accuracy of the models in segmenting the choroid plexus in brain MRI images from different populations, and consequently, their effectiveness in generalizing to new, unseen populations.

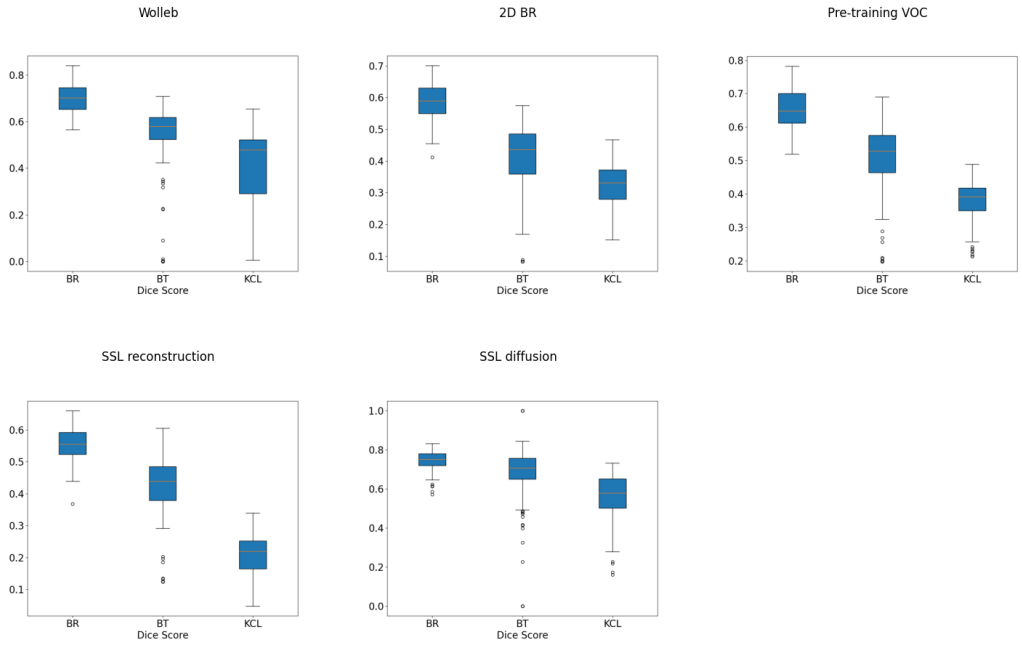

Fig. 4.4. Boxplots of mean Dice score for the two-dimensional models over Dataset 1, Dataset 2 and Dataset 3.

4.2 Three-Dimensional Segmentation

During the evaluation phase, the test set images, preserved in their original 3D format, were processed using the trained three-dimensional models following the procedure outlined in Algorithm 2. For each image, n segmentation volumes were generated using all models to create an ensemble, except for the diffusion SSL model, where only a single segmentation mask was produced. These volumes were subsequently thresholded at 0.5 to produce binary segmentations. The performance of the models was assessed using the Dice Score, calculated by comparing the predicted segmentations with the corresponding ground truth annotations for each image.

Table 4.4 presents the average Dice scores obtained from the first sample generated for each image. For diffusion SSL model, the reported value is the average Dice score from the single mask produced per image. These results were compared with those reported by Bieder et al. [54]. For this comparison, the model proposed by Bieder et al, available at https://github.com/florentinbieder/PatchDDM-3D, was replicated within the same workflow context used for the two-dimensional models. The model was adapted for the specific task of choroid plexus segmentation and trained using Dataset 1. The same procedure described in Algorithm 2 was employed to generate the ensemble for each image, and the average Dice score for the first sample was calculated.

Additionally, for the models generating an ensemble of masks, the mean segmentation volume was determined by averaging the n segmentation masks obtained for each image on a voxel-wise basis. These mean volumes were then thresholded at 0.5 to produce binary segmentations, and the corresponding Dice scores were calculated and reported in Table 4.4. This approach was intended to evaluate whether the ensemble of segmentations could improve performance. As in the two-dimensional case, variance values were computed and visualized using boxplots 4.5 to better understand the score distribution and identify potential outliers.

Method	Mean Dice score	Ensemble Dice score
Baseline	0.520 [0.012]	0.528 [0.013]
3D BR	0.456 [0.009]	0.457 [0.009]
SSL diffusion	0.413 [0.010]	-

Table 4.4: Segmentation scores of my 3D methods and baseline in terms of Dice score.

For models generating an ensemble of masks, an uncertainty map was derived from twenty

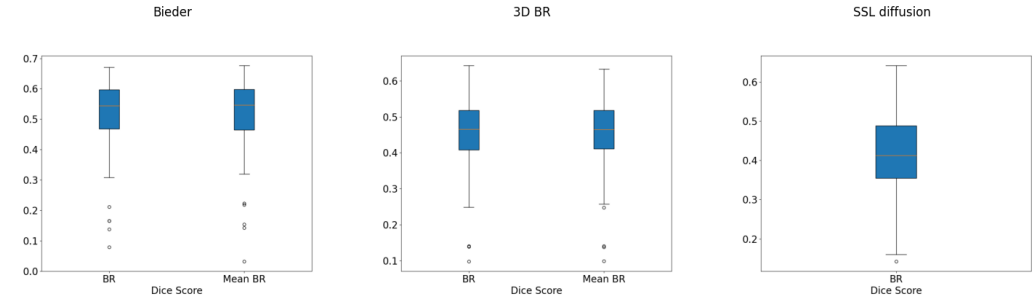

Fig. 4.5. Boxplots of mean Dice score and ensemble Dice score for the three-dimensional models.

available samples by computing the voxel-wise variance across the segmentation maps for each image. This variance serves as a measure of uncertainty, highlighting areas where the model's predictions showed inconsistency. By analyzing these uncertainty maps alongside the mean segmentation volume, deeper insights into the reliability of the model's predictions can be gained.

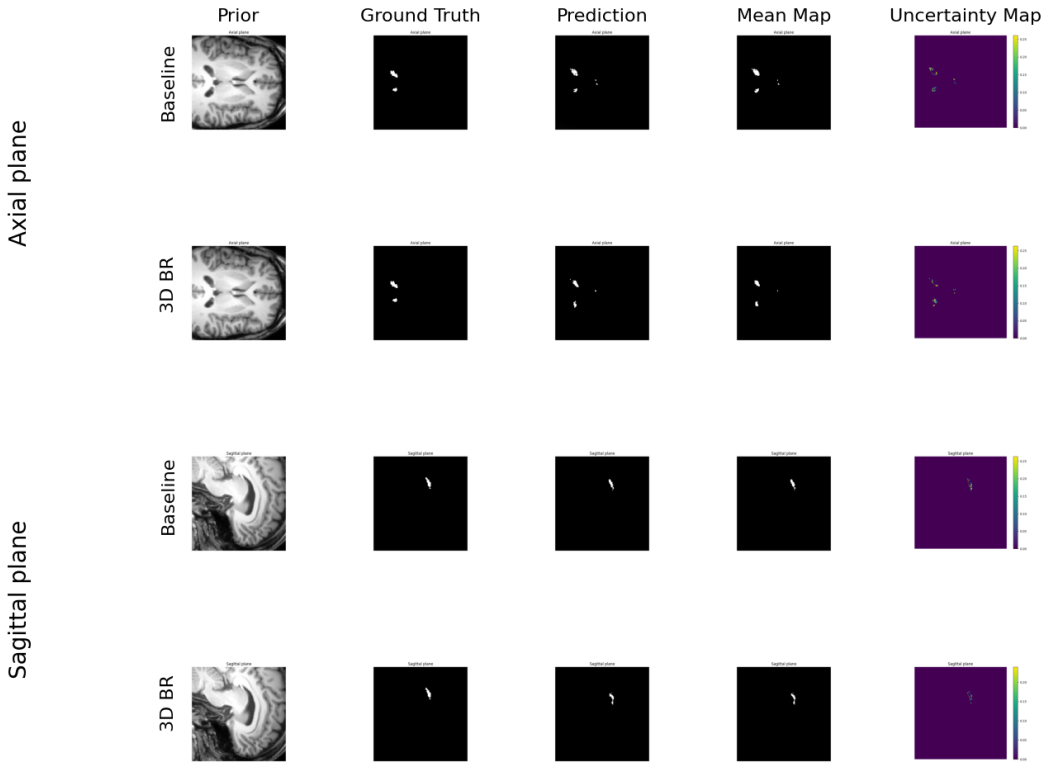

Fig. 4.6. Anatomical prior, ground truth segmentation, first prediction, mean map and uncertainty map for the diffusion-based models. Only the axial and saggital plane are reported since is difficult to appreciate the choroid plexus in the coronal plane.

In addition, the total sampling time required to generate the ensemble of segmentations was calculated for all models, and the average total sampling time is reported in Table 4.5. For the diffusion SSL model, only the sampling time for a single segmentation map was recorded. Also in this case, the DDIM sampler was tested and the corresponding total sampling times are included in Table 4.5. Mean Dice scores are also reported to assess whether the reduction in sampling time maintains high performance.

Method	DDPM		DDIM	
	Mean Sampling Time	Mean Dice Score	Mean Sampling Time	Mean Dice Score
Baseline	185.929	0.520	133.490	0.228
3D BR	184.900	0.456	184.036	0.338
SSL diffusion	5.518	0.413	-	-

Table 4.5: Mean sampling times and mean dice scores for both scheduler

Finally, similar to the two-dimensional case, the generalization capability of the proposed models was evaluated by applying them to the testing fractions of Dataset 2 and Dataset 3. The same metrics used for Dataset 1's test subset were calculated and are presented in Table 4.6. This analysis allows us to assess the accuracy of the models in segmenting the choroid plexus in volumes derived from different populations than the one used during training, thereby evaluating the models' ability to generalize to new, unseen populations.

Method	Dataset 2		Dataset 3	
	Mean Dice Score	Ensemble Dice Score	Mean Dice Score	Ensemble Dice Score
Baseline	0.469 [0.013]	0.472 [0.013]	0.077 [0.013]	0.077 [0.013]
3D BR	0.355 [0.009]	0.355 [0.009]	0.266 [0.012]	0.266 [0.013]
SSL Diffusion	0.277 [0.010]	-	0.228 [0.005]	-

Table 4.6: Dice scores of the 3D methods and the baseline over the testing fraction of Dataset 2 and Dataset 3.

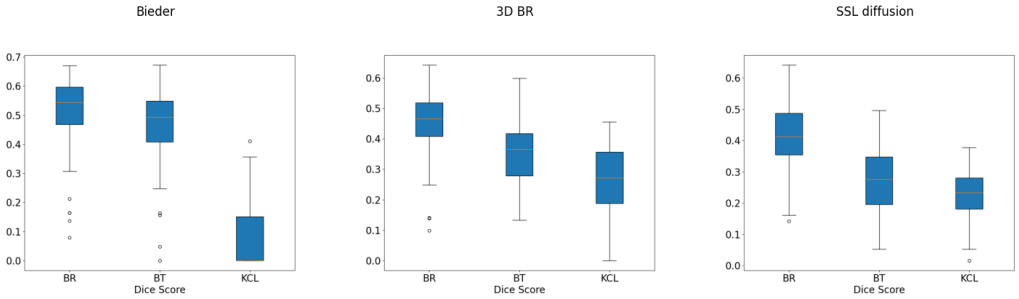

Fig. 4.7. Boxplots of mean Dice score for the three-dimensional models over Dataset 1, Dataset 2 and Dataset 3.

Chapter 5

Discusssion

5.1 Two-Dimensional Segmentation

5.1.1 Dice Scores

The comparison of average Dice Scores in Table 4.1 indicates that the model developed by Wolleb et al. [13] achieves excellent performance in segmenting the choroid plexus. Originally designed for brain tumor segmentation from MRI, this model also effectively segments the choroid plexus, despite its significantly smaller size compared to the original target. Wolleb et al.'s model served as a baseline for evaluating other methodologies that build upon its foundational idea, incorporating additional deep learning techniques. Contrary to expectations, only one of the proposed models show improvement in the performance. The following sections will analyze the reasons for the observed results expressed in terms of Dice Score values.

Firstly, unlike the baseline model, other models used data augmentation for part of the training data, including affine transformations, elastic deformations, and the application of various MRI artifacts. This approach aims to realistically emulate the quality and conditions of data available in clinical practice. Furthermore, to emphasize this aspect, data duplication was avoided for the transformed data, better simulating real-world conditions where data is often corrupted by artifacts and noise, and clean versions are unavailable. This led to a lower representation of artifact-free data in the training and validation sets compared to those used for training Wolleb et al.'s model, thus justifying the inferior performance of the proposed models when tested on data coming from the artifact-free population, as shown in Table 4.1.

Another possible factor contributing to the decline observed in the proposed models is the premature interruption of the training loops. Due to time constraints, the training for all models was halted before reaching completion or triggering early stopping, which likely prevented the models from fully learning from the data and achieving optimal performance. Consequently,

future work should involve resuming the training from the last checkpoint to determine if extended training can enhance model performance.

The performance of the models were further analyzed through the distribution of Dice Scores, as shown in the boxplots 4.1. As previously discussed, these plots illustrate the decline in performance for three of the proposed models compared to the baseline. Specifically, a decrease in the median can be observed, although it tends to increase across the various methods presented due to the incorporation of pre-training and self-supervised learning. The boxplots also offer insights into the variability of the results. Both the boxplots and the reported variance values indicate variability comparable to that of the baseline, resulting in similar deviations around the mean. The observed values suggest that all models exhibit stability around the median, with relatively small deviations from the central tendency. However, the presence of some outliers suggests significant deviations in model performance in certain instances. Since these outliers represent only a small fraction of the test set, they do not substantially impact the overall distribution, allowing us to conclude that the models demonstrate general stability on the test set.

Despite the performance drop observed in most of the proposed models, data augmentation is generally expected to enhance the robustness and generalization capability of these models. By training with augmented data, the models are anticipated to demonstrate greater resilience to noise and artifacts present in input images, effectively segmenting the region of interest even when the images are corrupted. This reduces the need for extensive pre-processing, thereby decreasing the overall time required to generate segmentation maps from image acquisition. Additionally, models trained with data augmentation are expected to generalize better, yielding more accurate segmentations on data from populations different from the training set. To verify this generalization capability, Dataset 2 and Dataset 3, containing data not used for training and from a different population than Dataset 1, were used. Contrary to the expectation, the results reported in Table 4.3 indicate a decline in the performance of the proposed models on Dataset 2 and Dataset 3. The reported values are not only lower than those obtained with the baseline model, which did not use data augmentation, but the performance drop is comparable to or even more pronounced in the proposed models than in the baseline. However, examining the distribution of Dice Scores reveals an important insight. Both the boxplots 4.4 and the calculated variance indicate a decrease in variability, particularly in Dataset 3, which includes data from a population most distinct from the training data. This increased stability in the proposed models on unseen populations is attributable to the data augmentation applied during training. A notable exception is the model pre-trained with a diffusive pre-text task. In this case, the model performed better on Dataset 2 and Dataset 3 than the baseline, highlighting its superior generalization ability. The results suggest that traditional data augmentation techniques

are effective for non-diffusive models, such as those based on the U-Net architecture, but that more specific augmentation strategies may be required for diffusive models to enhance their generalization ability. Nevertheless, the performance on unseen data underscores a critical point: while data augmentation can improve generalization, the limited amount of training data available does not allow the models to achieve performance levels comparable to those on Dataset 1. Therefore, a possible future direction could be to retrain the models using larger datasets to achieve good performance across different data populations and explore different data augmentation modalities, specifically tailored for diffusive models. For example, it could be beneficial to implement offline data augmentation by coregistering the training data with each other. This approach would increase the volume of data and enhance variability, leading to more robust model training and improving the generalization ability.

A potential explanation for the limited effectiveness of data augmentation methods on diffusion-based models could be related to the intrinsic workings of these models. As previously mentioned, diffusion models are trained to remove Gaussian noise added to images, enabling them to generate segmentation masks starting from pure noise. Some of the data augmentation techniques proposed in this work involve adding simulated noise that mimics the types typically observed in MRI images. During training, the model might identify this simulated noise as an artifact specifically introduced for training purposes. Consequently, the model could learn to predict and remove this added noise rather than generalize from it, thereby diminishing the effectiveness of the data augmentation and reducing its potential to enhance model generalization. This phenomenon could explain the observed reduction in the effectiveness of data augmentation for improving the performance of diffusion-based models. However, this explanation remains a hypothesis based on the nature of noise addition techniques and the theoretical foundations of diffusion models. To validate this hypothesis, future research should attempt to replicate the study while excluding these data augmentation techniques. This would help determine whether there is a significant difference in results when applied to Datasets 2 and 3, thus providing more insight into the impact of noise addition on model performance.

The results presented in Table 4.3 allow for another significant observation: the decline in performance is more pronounced on Dataset 3 compared to Dataset 2. This outcome can be explained by considering the nature of the data contained within the two datasets. The data in Dataset 2 were obtained from subjects diagnosed with Relapsing-Remitting Multiple Sclerosis. Since the majority of subjects in Dataset 1 also suffered from the same condition, the models were able to learn how to accurately segment images with features characteristic of RRMS during the training phase. This, in turn, led to a smaller drop in performance when applied to the testing portion of Dataset 2. However, as discussed in the previous chapters, while both

datasets were acquired using scanners from the same manufacturer, they involved different models, software versions, and some variations in acquisition sequence parameters. These factors contribute to a shift in the statistical distribution of the data, resulting in reduced model performance on previously unseen data.

The situation with Dataset 3 is even more challenging. The data in this dataset were acquired in London, as opposed to Verona, where the previous datasets were collected. Furthermore, Dataset 3 includes a small fraction of healthy subjects and a majority of patients with depression. The absence of data with features typical of depressed subjects in the training images is likely a key reason for the observed drop in performance. Additionally, the use of a scanner from a different manufacturer introduces a phenomenon known as manufacturer shift [73]. This is recognized as a major cause of changes in statistical distributions, as different MRI scanners produce images with distinct characteristics related to the specific MRI physics of the manufacturer. For these two reasons—the lack of relevant features in the training data and the manufacturer shift—when the models are applied to Dataset 3, the performance degradation is more significant compared to their application on Dataset 2. The nature of the data in the former leads to a more substantial shift in the statistical distribution of the images relative to those used for training.

As previously mentioned, the conclusions regarding the generalization capabilities of the proposed models are further supported by the analysis of Dice Score distributions. Specifically, the boxplots reveal a bigger or comparable decline in the median value for the proposed methodologies when applied to data from Dataset 2 and Dataset 3. Additionally, the interquartile range and whiskers indicate a lower variance in Dice Scores compared to those observed in the baseline model, reflecting reduced variability due to the use of data augmentation, especially over Dataset 3. However, the variability remains higher than that recorded for Dataset 1, underscoring the need for larger training sets that include data from diverse populations and scenarios to achieve performance comparable to that on the test fraction of Dataset 1. The measured variances of the Dice Score distributions further reinforce this observation. The proposed methodologies demonstrate smaller variance values, indicating greater stability compared to the baseline when applied to unseen data from populations different from those represented in the training data.

Another important observation is the improvement in model performance with pre-training on non-biomedical images, increasing the average Dice Score and reaching the second-best performance among all proposed models. This result is significant given the difficulty of obtaining large quantities of biomedical images, crucial for achieving excellent performance without transfer learning. The results show that it is possible to compensate for the lack of brain MRI images by using natural images and pre-training and fine-tuning practices. In this specific

case, VOC2012 was used due to the limited computational resources available. However, using larger publicly available datasets of non-biomedical images, such as Microsoft COCO [74], could lead to further performance improvements, especially in scenarios with even more limited training data.

The benefit of using natural images comes at the cost of implementing a pre-training phase with a vast amount of data, followed by a fine-tuning phase. This can increase the training time due to the larger data volume handled during pre-training. Nevertheless, the number of epochs in the fine-tuning phase can be reduced, partially compensating for the required training time. The goal of pre-training is to approach the global minimum of the loss function, facilitating its attainment during fine-tuning with fewer data. Therefore, a higher learning rate can be used in the initial phase, allowing for faster convergence and reducing the required time. It is noteworthy that among all the proposed methodologies, pre-training with natural images required the least training time, thanks in part to the validation loss check that triggered early stopping during the pre-train phase, preventing overfitting and saving time.

The results presented in Table 4.1 indicate that the model pre-trained with a contrastive generative pre-text task performed the worst. Unexpectedly, it not only underperformed compared to the model pre-trained on non-biomedical images but also lagged behind the model that did not utilize any pre-training. This poor performance can be attributed to two potential factors, aside from the premature interruption of the training loop. First, the choice of loss function during pre-training may have contributed to the suboptimal results. The pre-training utilized a hybrid loss function, combining a reconstruction loss with a contrastive loss applied to the latent representation at the network bottleneck. However, this specific hybrid function, along with the chosen temperature hyperparameter, may not have been well-suited for the model architecture. Future work should explore different configurations to identify the optimal loss function and hyperparameter settings for this specific model. Second, only the encoder of the network was pre-trained, and the pre-trained weights were transferred to the encoder of the final network for fine-tuning. This transfer of weights can introduce errors, especially if there are differences between the architectures of the two networks. To mitigate this, it would be more appropriate to pre-train the encoder of the same network that will be used for fine-tuning. For example, setting the timesteps to zero during pre-training could prevent the diffusion process and make the pre-training more relevant to the final task. Additionally, pre-training the entire network, rather than just the encoder, could potentially lead to greater improvements in model performance. Therefore, future work should explore these alternative strategies to determine if they lead to better outcomes. Implementing these solutions could provide insights into whether such adjustments would improve the model's performance.

Among the proposed methodologies, the one employing diffusion in a self-supervised learning

context for pre-training a non-diffusive segmentation model achieves the best results. This underscores the importance of self-supervised learning methods in research. Beyond the challenge of obtaining large volumes of medical images, acquiring corresponding segmentations is also difficult, as discussed in previous chapters. Therefore, the ability to leverage unlabeled data is crucial as it improves model performance. Diffusion models are particularly useful as they learn significant latent representations of images during training, which are beneficial for subsequent segmentation. Additionally, using the same U-Net architecture for noise prediction during pre-training and fine-tuning avoids errors in weight transfer between networks and inconsistencies between the network structure and the implemented pretext task. Nevertheless, it is worth noting that this model is based on the direct application of a U-Net network, which is recognized as a state-of-the-art architecture for medical segmentation tasks. This could explain the superior performance of this model.

Another point of reflection concerns the choice of loss function. To maintain consistency with Wolleb et al.'s work, Mean Square Error was used during model fine-tuning. However, given the nature of the choroid plexus, a different loss function might improve performance. Comparing the baseline results with those of Wolleb et al.'s original work shows a significant reduction in Dice Score, likely due to the size difference between the target regions. Using a custom-built loss function, including for example a Focal Loss term, could enhance performance by addressing the high background pixel proportion relative to the unitary pixels in the segmentations. Additionally, the weighted MSE used in pre-training with diffusion models could be a useful alternative for all models, focusing more on the image content and ignoring insignificant details by increasing the weighting in the coarse and content phases and reducing the weights in the initial timesteps when the SNR is high. Nonetheless, these suggestions are hypotheses and have not been tested; therefore, it is not guaranteed that the suggested modifications will yield the discussed results.

A final consideration similar to the previous one regards the choice of the optimizer. For all the different methodologies, Adam was chosen based on Wolleb et al.'s original work. However, even if it is a solid and reasonable choice supported by multiple works in the literature, the models could benefit from different choices, such as AdamW. Another important aspect is the choice of the learning rate. In this work, few experiments were conducted with this hyperparameter and finally, supported by the evidence, the selected value was the same utilized for the baseline model. Nevertheless, a more accurate grid search could be done to evaluate if there is a better value to be chosen.

In conclusion, despite seemingly unsatisfactory results, the methodologies proposed highlight several positive aspects. They increase robustness to MRI artifacts and enhance model generalization, even if on a reduced scale than the expected. Furthermore, they demonstrate the

effectiveness of transfer learning, both with labeled natural images and unlabeled biomedical images, emphasizing its importance in common practice. Finally, they suggest potential future modifications to improve model performance based on the region of interest's nature and the implemented diffusion process.

5.1.2 Ensemble

As mentioned in the previous chapters, one of the advantages of diffusion models is the ability to generate an ensemble of segmentation maps for each image. Due to the probabilistic nature of the sampling process, repeating it for a given image generates a different segmentation map each time. Therefore, with models using a diffusion process, this phenomenon was exploited to generate an ensemble of segmentations, resulting in an average map and an uncertainty map. The following section discusses the obtained results.

First, it is crucial to address the choice of ensemble dimensionality. As discussed earlier, we evaluated the Dice Score of the average map by varying the number of samples per image to determine the dimensionality that yields the most significant improvement in performance. For the baseline model, we generated a Dice Score curve for each image in the test set and then averaged these curves. By identifying the number of samples that maximized the performance metric, we determined the optimal ensemble size for each model.

As shown in Figure 4.2, the highest average Dice Score for the baseline model is achieved with thirteen samples. However, this number was not selected as the ensemble size. The difference in Dice Score between using 2 and 13 samples is 0.013, while the difference between 2 and 5 samples is 0.010. In other words, the increase in Dice Score from using 13 samples instead of 5 is, on average, 0.003. However, this marginal gain comes with a significant increase in the time required for sampling the entire ensemble. Specifically, obtaining 13 samples takes more than two and a half times the time required to sample 5. Given the need to balance performance improvement with sampling time, an ensemble size of 5 was chosen. This decision also aligns with the original work of Wolleb et al., allowing for a more direct comparison of the results. For this reason, the same value was chosen for the proposed models, a decision supported by the results shown in Table 4.1, which indicate an improvement in performance when comparing the single map to the average map. However, it is likely that different models may have distinct optimal ensemble sizes. Therefore, future work should involve replicating the test conducted on the baseline for each model, in order to determine whether specific values for each model could further enhance performance.

The average Dice Scores for the ensembles, presented in Table 4.1, indicate that all diffusion-based models benefit from using an average map. This suggests that the pixel-level averaging

of the segmentations can lead to better delineation of the choroid plexus. The averaging process attenuates segmentation errors while preserving correctly classified pixels. Consequently, thresholding the average maps helps assign erroneously classified pixels to the background, thereby improving segmentation accuracy. Despite requiring quintuple the sampling time, this approach is advantageous since each segmentation sampling takes approximately 15 seconds, making the temporal compromise worthwhile for performance gains. The improvement in performance gained by considering the mean maps is further confirmed looking at the boxplots. By comparing the distribution of the Dice Scores for the single sample and the mean map, an increase in the median value is observable across all the different methodology.
Another benefit of generating an ensemble is the ability to calculate an uncertainty map (Figure 4.3) as the pixel-level variance of the different segmentation masks. When paired with the average map, the uncertainty map reveals where the model is most uncertain in segmenting the region of interest, providing valuable insights for clinicians. Our results show that all proposed methodologies exhibit greater uncertainty along the choroid plexus borders. This can be attributed to several factors:

- Boundary Ambiguity: In medical images, object boundaries often present gradual transitions rather than sharp edges, making it challenging for the model to precisely delineate where the object ends and the background begins. This is particularly evident in the case of the choroid plexus in T1-weighted MRI images, where it is challenging to delineate the region of interest within the ventricles.

- Inter-annotator Variability: Training segmentations were annotated by different specialists, introducing a certain degree of inter-user variability in the defined boundaries. This inter-user variability contributes to the model's uncertainty.

Additionally, uncertainty maps highlight a limitation of diffusion models in segmenting very small structures like the choroid plexus. In some slices, the plexus may be represented by a single pixel or a small cluster of pixels. In such cases, the model shows high variance in segmenting the region of interest in these areas. This is closely related to the nature of diffusion models. During training, Gaussian noise is progressively added to the data, which the model learns to remove. This allows generating a segmentation map from a pure Gaussian noise image during sampling. Such regions consisting of single pixels or small isolated pixel clusters may be interpreted by the model as random occurrences of Gaussian noise, leading to their removal from the final segmentation maps, in particular if there are uncertainties in the ground truth segmentations. As mentioned before, a possible future direction to avoid this problem is the usage of tailored loss functions during the training phase that allow to focus more on the region of interest and its peculiar features.

In any case, it is important to acknowledge that uncertainty maps obtained with ensembles of few samples provide only a rough estimate of variance. For more precise maps, increasing the ensemble size is necessary. However, as indicated in Wolleb et al.'s study, even with 100 samples per ensemble, the highest variance is typically observed around the tumor perimeter, corroborating our previous discussions. To illustrate this, we selected three images and calculated the variance among the twenty sampled masks, which are presented as examples of uncertainty maps.

In conclusion, generating multiple segmentation maps for each image enables the calculation of average maps that enhance segmentation performance and uncertainty maps that identify where the model struggles the most in segmenting the region of interest. Despite the increased sampling time, these methods offer significant advantages in improving the accuracy and reliability of medical image segmentation.

5.1.3 Schedulers

As previously mentioned, the remarkable success of diffusion models has led to the development of several variants aimed at enhancing performance. One such variant is the Denoising Diffusion Implicit Model, which replaces the stochastic sampling process with a deterministic one. This approach allows skipping steps in the reverse diffusion process, thereby reducing the sampling time while minimizing the impact on model performance. In this work, we tested the use of both types of schedulers, leveraging the fact that the model only needs to be trained once with either method. The following section discusses the results obtained from this comparison. As shown in Table 4.2, in almost all cases, the choice of the DDIM scheduler reduces the sampling times but at the cost of decreased performance, measured in terms of the average Dice Score for the first sample. Although the time reduction is on the order of a few seconds, this modest gain is insufficient to justify the observed decline in performance. The primary objective in medical image segmentation is to achieve the highest possible accuracy, as even small improvements can be critical in clinical settings. Therefore, the slight reduction in sampling time does not outweigh the negative impact on segmentation quality. This is especially true given that the classic scheduler already operates within a reasonable timeframe and maintains higher performance levels. An exception to this trend is observed in the baseline model, where the application of the DDIM scheduler results in both a performance increase and a reduction in sampling time by approximately two and a half seconds. This outcome is unexpected and contradicts the established theory and results reported in other studies involving DDIMs. Therefore, further investigation is warranted to understand the underlying reasons for this phenomenon. If it is determined that this result is not due to an evaluation error, the

baseline model using the DDIM scheduler would represent the most effective model proposed in this study.

In conclusion, while DDIM schedulers can offer a reduction in sampling time, the marginal gains do not compensate for the reduction in segmentation performance. Therefore, the classic scheduler remains the preferred choice for achieving higher performance in a reasonable timeframe.

5.2 Three-Dimensional Segmentation

5.2.1 Dice Scores

The analysis of the three-dimensional models followed a similar approach to that of the two-dimensional models, revealing comparable trends in the outcomes. Specifically, three models were evaluated: the baseline model, the model with data augmentation, and the model pre-trained using a diffusive self-supervised learning approach. The key observations and findings are discussed below.

The baseline model achieved the highest performance among the models tested. However, its results were not comparable to those reported in the original work by Bieder et al. [54]. This discrepancy may be attributed to the structural differences between the regions of interest, as the model was originally developed for brain tumor segmentation from MRI images. Despite this, the application of data augmentation techniques, as discussed for the two-dimensional models, is expected to improve the model's robustness to typical distortions found in real MRI images. However, this came at the cost of a relevant reduction in overall performance, as measured by the Dice Score and highlighted by the median values in the boxplots, on artifact-free data. This performance decrease is likely due to the introduction of more realistic training conditions, better simulating the challenges encountered in clinical practice. While there was a reduction in performance, the model's ability to generalize to different data populations outside the training set is expected to improve.

Interestingly, the results from Dataset 2 and Dataset 3 reveal a similar trend from the two-dimensional case. The performance of the three-dimensional models with data augmentation on data from unseen populations experienced a more significant decline compared to the baseline model. This trend is clearly illustrated in Table 4.6 and the accompanying boxplots 4.7, which show a more pronounced performance drop for the augmented models on Dataset 2. However, a notable deviation occurs with Dataset 3: contrary to the two-dimensional scenario, the models incorporating data augmentation actually performed better. Despite this improvement, the results obtained with the baseline model on Dataset 3 are unexpectedly low, which is

likely due to an error during the evaluation phase, especially since this trend is not observed on Dataset 2, which is more similar to the training population. An alternative explanation could be that the data augmentation had a positive effect only on the most diverse population, thereby compromising performance on both Dataset 1 and Dataset 2. However, this alternative explanation is less likely, as it would suggest a highly selective and unusual effect of data augmentation that contradicts the principle at the base of this methodology. Additionally, the variance for the proposed models is comparable or even high than that of the baseline model, indicating less stable results across different data populations. The boxplots further confirm this instability, showing a wider interquartile range for the proposed models compared to the baseline, which suggests that 50% of the recorded results are more dispersed around the median value.

One potential reason for this difference from the two-dimensional trend could be the premature interruption of the training process. Due to time constraints, the training loops for both three-dimensional models were halted before completion, potentially preventing the models from learning adequately to perform well on unseen populations. Notably, the three-dimensional models were trained for significantly fewer epochs compared to their two-dimensional counterparts. To fully assess their potential, future work should involve completing the training to determine whether performance improves. If the performance does not improve, it may indicate that the implemented data augmentation is less effective in enhancing model generalizability in three-dimensional scenarios compared to two-dimensional ones, necessitating the evaluation of alternative techniques.

Another noteworthy observation concerns the model pre-trained with a diffusive pretext task. In the two-dimensional case, this model was the best performer, but in the three-dimensional scenario, it was the worst-performing one. This discrepancy is largely attributed to the fact that, due to time constraints, the pre-training was halted after 300 epochs instead of the intended 2000 before proceeding to fine-tuning, determining a substantial difference with its two-dimensional counterpart. Given that this model, which utilizes a straightforward U-Net architecture without diffusive processes, is expected to perform well, future work should ensure that the pre-training is fully completed before beginning fine-tuning to achieve the anticipated results.

Another critical factor contributing to the performance drop is the amount of data available. In the two-dimensional case, 32 slices per image were used as training examples, resulting in a larger training set. In contrast, in the three-dimensional case, only 5 patches were extracted per volume, making the training set approximately six times smaller. This substantial reduction in training data likely contributed to the lower performance observed, as the model was trained on a much more restricted dataset. As mentioned in the context of the two-dimensional case, future work should prioritize retraining the models with larger and more diverse datasets. One

potential approach is to implement offline data augmentation through the coregistration of training data, which could enhance the models' performance and improve their generalization capabilities.

The difference in performance decline between Dataset 2 and Dataset 3 in the three-dimensional models mirrors the pattern observed in the two-dimensional models. As previously mentioned, Dataset 1 and Dataset 2 are more similar, whereas Dataset 1 and Dataset 3 differ more significantly. This underscores the importance of different acquisition protocols and the need to maximize model generalization through data augmentation techniques. Among the various factors contributing to performance decline, the most significant is the manufacturer shift. Different MRI scanners produce images with distinct characteristics due to variations in MRI physics, leading to changes in statistical distributions. Despite the benefits of data augmentation in enhancing generalization, the small amount of available training data limited the models' ability to achieve performance levels comparable to those on Dataset 1.

A significant finding from the analysis of the three-dimensional models is the clear need for further research and experimentation to validate the effectiveness of transfer learning and data augmentation techniques in enhancing model generalization, especially when training data is limited. However, it is also clear that the limited amount of training data impacted the models' ability to perform well across all datasets. This finding reiterates the need to further explore the choice of loss functions and optimizers, as mentioned earlier for the two-dimensional models, to better optimize the performance of three-dimensional models, considering the specificities of the target regions to be segmented.

Finally, it is worth discussing the patch-based methods employed and the structural modifications made to the U-Net architecture. Regarding the former, the proposed models incorporate three key aspects: the number of patches extracted per image, the sampling of the patch centers, and the sampling of patch sizes. In this specific case, five patches per image were extracted to remain consistent with the approach taken by Bieder et al. in their work, which was also replicated in the baseline model. However, it is certainly possible to conduct additional tests to fine-tune this hyperparameter for optimal model performance. It should be noted, though, that increasing the number of patches also increases the volume of data used during training, which in turn requires more time and computational resources. Additionally, the centers of these patches were sampled from a distribution similar to that implemented by Bieder et al. This choice was made not only to ensure consistency with the baseline for fair performance comparison but also because the proposed distribution assigns a higher probability to selecting central regions of the volume as patch centers. Given the location and anatomical structure of the choroid plexus, this approach frequently extracts patches containing the region of interest, enabling the model to learn relevant features for subsequent image segmentation. Nevertheless,

slight adjustments to the distribution, tailored to the specific location and characteristics of the choroid plexus, could potentially enhance performance. Thus, exploring these modifications represents a possible future direction for this work.

The final aspect mentioned above pertains to the sampling of patch sizes. To ensure faster training and reduce computational demands, patches with dimensions of 32x32x32 were predominantly used. To avoid neglecting relationships between different regions within the overall volume, a small proportion of patches with dimensions of 64x64x64 and a very small proportion of full-resolution images (128x128x128) were also included. This was achieved by randomly sampling patch sizes, assigning a 75% probability to 32, 15% to 64, and 10% to 128. These values were primarily chosen based on the available computational resources. However, the selected values are quite reasonable given the goal of including larger patches and full-resolution images. That said, a potential future development of this work could involve fine-tuning these percentages to further optimize model performance.

As for the architectural modifications made to the U-Net, the decision to use additive skip connections allows for a significant increase in the network's depth while still maintaining rapid training. In fact, compared to the two-dimensional models, the three-dimensional models required less training time. Therefore, it may be interesting to test additive skip connections in two-dimensional models in the future to see if they also reduce training time without significantly affecting performance. This could be an important step in using diffusion models in the medical context, where it is essential for the model to be ready for use in a short time.

In conclusion, the results obtained with the three-dimensional models confirm the observations made for the two-dimensional models while offering new insights for improving segmentation techniques in three-dimensional contexts. These findings underscore the need to continue exploring pre-training strategies, data augmentation, and methodological choices to address the challenges posed by the different characteristics of the datasets used. Furthermore, the patch-based methods require further investigation to improve performance and enhance the time and computational efficiency of diffusion models when dealing with three-dimensional volumes. Therefore, although this is just an initial attempt to adapt diffusion models and patch-based techniques for the three-dimensional segmentation of the choroid plexus, the results underscore valuable insights that can guide future research aimed at enhancing performance and generalization ability.

5.2.2 Ensemble

In the context of three-dimensional models, the approach taken for the ensemble analysis and performance evaluation mirrors that described for the two-dimensional models, with some

differences due to the volumetric nature of the data. Here, too, the goal was to determine the optimal ensemble size by assessing the impact of the number of samples on the average Dice Score of the resulting segmentation maps and analyse the mean and uncertainty maps.
Regarding the choice of ensemble dimensionality, the decision was made to retain the value identified in the two-dimensional baseline case. This coincides with the value used in the original work by Bieder et al., thereby enabling a direct comparison of the results. However, it is important to note that, based on the findings, the selected value is unlikely to represent the optimal dimensionality for the two three-dimensional models based on diffusion processes. It is reasonable to expect that these two models have different optimal ensemble sizes that could improve performance, especially given the transition from two-dimensional to three-dimensional mask generation. Therefore, as previously suggested, future work should focus on testing the models to identify their optimal dimensionalities and determine whether these allow for a favorable balance between performance improvement with the average map and the time required for ensemble acquisition.
When comparing the average Dice Scores for individual samples and the mean maps, only marginal improvements are evident. Given the time required to sample an entire volume using diffusion-based models, the minimal performance gain does not justify the significant increase in sampling time. Specifically, with an ensemble size of five, the Dice Score improvement is less than 0.01 for both the baseline model and the model with data augmentation. This small improvement does not warrant a fivefold increase in sampling time, indicating that evaluating a single sample may be the more efficient option. This is further confirmed looking at the median values and the variability of the Dice Scores distribution in the boxplots. This is likely due to the premature interruption of the training loops, which prevents observing a significant performance improvement, as well as the chosen ensemble dimensionality, which has not been specifically optimized for three-dimensional models. Nonetheless, generating an ensemble remains valuable for other purposes, such as the evaluation of uncertainty maps.
Regarding the calculation of uncertainty maps, the three-dimensional models also exhibit greater variability near the choroid plexus boundaries. This phenomenon can be attributed to factors already discussed in the previous section, such as boundary ambiguity and inter-annotator variability. However, the problem is further exacerbated by the three-dimensional nature of the images, where the volumetric representation can make it even more challenging for the model to accurately delineate the boundaries. In particular, it was noted that very small structures, similar to what was observed in the two-dimensional models, tend to be treated as Gaussian noise by the model. This results in high variance in the uncertainty maps, suggesting that more focused training techniques, such as the use of tailored loss functions, may be necessary for more precise segmentation of such structures.

In summary, applying ensembles in three-dimensional models does not lead to significant Dice Score improvements when evaluating the average map, particularly when considering the substantial increase in sampling time. However, the trend observed in two-dimensional models regarding variance is confirmed in the three-dimensional uncertainty maps, offering valuable insights into the areas most challenging to segment accurately.

5.2.3 Scheduler

In line with the previous experiments using two-dimensional models, the application of a different scheduler was also tested with the three-dimensional models employing diffusion mechanisms. Specifically, the Denoising Diffusion Implicit Models scheduler was explored to determine if it could reduce sampling time without significantly compromising the previously achieved performance. This section discusses the outcomes of this analysis.

The results, summarized in Table 4.5 show that while using the DDIM scheduler reduces the sampling time for the proposed diffusion-based model, similar to the 2D models, the time savings are minimal—on the order of a second. This small gain becomes particularly relevant when considering performance differences between schedulers. Specifically, the reduction in sampling time with the DDIM scheduler is negligible compared to the classical scheduler, while segmentation accuracy, measured by the mean Dice Score, decreases significantly, by more than 0.1. For the baseline model, the reduction in sampling time is more pronounced, approximately 50 seconds. However, the performance drop is even more substantial than with the other model, making the time savings less justifiable given the trade-off in accuracy. In the medical context, where both quick results and high accuracy are critical, this trade-off is crucial when selecting a scheduler. Consequently, the classical scheduler is the preferred choice, offering a better balance between performance and sampling time compared to the DDIM scheduler.

Although comparing the sampling times of the entire segmented volume between 2D and 3D models was not the primary focus of this section, it is worth addressing briefly. For the 2D models, the times reported in Table 2 pertain to the sampling of a single slice, not the entire volume, which consists of 32 slices. In contrast, the 3D models directly sample a volume comprising 128 axial slices. To make a fair comparison, the sampling times of the 2D models should be multiplied by 32 and then compared with those of the 3D models, considering that the latter involves four times as many slices.

Focusing on the DDPM scheduler, the time required to sample 32 slices is two and a half times longer than that needed by the 3D model to sample the entire volume. The disparity increases with data-augmented models, where the time required to sample the 32 slices exceeds

three times that of the 3D model. Conversely, when considering the DDIM scheduler, the difference is more contained—about two times longer for the 2D model compared to the 3D model—and unlike before, it improves for the data-augmented model compared to the baseline. This improvement is due to the fact that, for the 2D models, the DDIM scheduler offers a better compromise between reduced sampling time and performance, compared to the 3D models.
Overall, it is evident that sampling the entire segmented volume is significantly more time-efficient than sampling individual slices. In the former case, the diffusion process only needs to be performed once per volume, whereas in the latter, it must be repeated for each slice. This represents one of the main advantages of developing 3D models directly, which, thanks to the advancement of patch-based techniques, are becoming temporally and computationally efficient even during the training phase, thereby addressing their primary limitation.
In summary, the three-dimensional models demonstrate that, although offering potential time savings, the adoption of a DDIM scheduler does not justify the loss of segmentation accuracy, making the classic scheduler the preferred choice. Furthermore, the comparison between the sampling times of two-dimensional and three-dimensional models shown the advantage of developing models directly capable of segmenting the entire volume, allowing for a significant reduction in the sampling times.

Chapter 6

Conclusion

In this preliminary work, several methodologies based on Denoising Diffusion Probabilistic Models were proposed and compared to already published and available models taken as reference baseline. Both two-dimensional and three-dimensional segmentation of the choroid plexus in brain MRI images was tested. The main objective was to explore techniques commonly used in deep learning to improve performance compared to the baselines. One of the major challenges in the field of artificial intelligence applied to the medical domain is the limited availability of training data and the frequent lack of corresponding labels. This presents a significant obstacle to the development of effective models for automatic segmentation.

Our results reveal that techniques such as data augmentation, transfer learning, and self-supervised learning hold considerable promise for enhancing deep learning models in medical segmentation tasks. Data augmentation has demonstrated its potential to help models generalize better to diverse populations, even with limited training data. Transfer learning and self-supervised learning allow the incorporation of external datasets, including non-biomedical or unlabeled data, to further boost model performance. However, our findings also suggest that more research is needed to optimize these techniques for diffusion-based models, especially regarding data augmentation. Tailored approaches for data augmentation and innovative transfer learning strategies should be explored to fully leverage these models' capabilities.

Another crucial aspect emerging from this work is the choice of DDPMs. Although these models were not originally designed for discriminative tasks like medical segmentation, they offer a notable advantage: the ability to generate ensembles of different segmentation masks for each image. This allowed us to demonstrate that using the average map can, in some cases, improve the accuracy of segmentations in terms of Dice Score. Furthermore, the ability to generate multiple segmentation masks for each image enables the creation of uncertainty maps, an aspect of great value in the medical context. These maps help to identify areas where the model is more uncertain in its predictions and those where it provides more reliable results,

offering clinicians a valuable tool to improve the analysis of the obtained segmentations and minimize the risk of erroneous clinical decisions, or to provide insight on the intrinsic difficultness of the automatic segmentation task, potentially provide a spatial varying index of reliability.

Finally, our exploration into three-dimensional segmentation yielded less favorable results but offered valuable insights. The initial outcomes underscore the need for further refinement of patch-based techniques tailored to the anatomical specifics of the choroid plexus. Additionally, as observed in the two-dimensional scenario, expanding the training dataset, and developing data augmentation methods specifically designed for diffusion-based models are crucial for improving performance and generalization.

In conclusion, while the results are promising, there is still a need for ongoing research to enhance the effectiveness of these models. Future work should focus on developing specialized approaches that address the unique complexities of the choroid plexus and other similar anatomical structures.

Bibliography

[1] A. Razavi, A. van den Oord, and O. Vinyals, "Generating diverse high-fidelity images with vq-vae-2," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/5f8e2fa1718d1bbcadf1cd9c7a54fb8c-Paper.pdf.

[2] Z. Kong, W. Ping, J. Huang, K. Zhao, and B. Catanzaro, "Diffwave: A versatile diffusion model for audio synthesis," Sep. 2020. doi: https://doi.org/10.48550/arXiv.2009.09761. [Online]. Available: https://www.researchgate.net/publication/344335006_DiffWave_A_Versatile_Diffusion_Model_for_Audio_Synthesis.

[3] X. L. Li, J. Thickstun, I. Gulrajani, P. Liang, and T. B. Hashimoto, "Diffusion-lm improves controllable text generation," in *Proceedings of the 36th International Conference on Neural Information Processing Systems*, ser. NIPS '22, New Orleans, LA, USA: Curran Associates Inc., 2024, isbn: 9781713871088. doi: https://doi.org/10.48550/arXiv.2205.14217. [Online]. Available: https://dl.acm.org/doi/10.5555/3600270.3600583.

[4] A. Kazerouni, E. K. Aghdam, M. Heidari, *et al.*, "Diffusion models in medical imaging: A comprehensive survey," vol. 88, 2023, p. 102 846. doi: https://doi.org/10.1016/j.media.2023.102846. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1361841523001068.

[5] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, "Generative adversarial nets," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'14, Montreal, Canada: MIT Press, 2014, pp. 2672–2680. doi: https://doi.org/10.48550/arXiv.1406.2661. [Online]. Available: https://dl.acm.org/doi/10.5555/2969033.2969125.

[6] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *Proceedings of the 31st International Con-*

ference on International Conference on Machine Learning - Volume 32, ser. ICML'14, Beijing, China: JMLR.org, 2014, pp. II–1278–II–1286. doi: https://doi.org/10.48550/arXiv.1401.4082. [Online]. Available: https://dl.acm.org/doi/abs/10.5555/3044805.3045035.

[7] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real nvp," 2017. doi: https://doi.org/10.48550/arXiv.1605.08803. [Online]. Available: https://arxiv.org/abs/1605.08803.

[8] P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," vol. abs/2105.05233, 2021. arXiv: 2105.05233. [Online]. Available: https://arxiv.org/abs/2105.05233.

[9] F. Khader, G. Mueller-Franzes, S. T. Arasteh, *et al.*, "Medical diffusion: Denoising diffusion probabilistic models for 3d medical image generation," 2023. arXiv: 2211.03364. [Online]. Available: https://www.nature.com/articles/s41598-023-34341-2.

[10] H. Li, Y. Yang, M. Chang, *et al.*, "Srdiff: Single image super-resolution with diffusion probabilistic models," vol. 479, 2022, pp. 47–59. doi: https://doi.org/10.1016/j.neucom.2022.01.029. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231222000522.

[11] A. Lugmayr, M. Danelljan, A. Romero, F. Yu, R. Timofte, and L. V. Gool, "Repaint: Inpainting using denoising diffusion probabilistic models," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11 451–11 461, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:246240274.

[12] A. Durrer, J. Wolleb, F. Bieder, *et al.*, "Denoising diffusion models for 3d healthy brain tissue inpainting," *ArXiv*, vol. abs/2403.14499, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:268554050.

[13] J. Wolleb, R. Sandkühler, F. Bieder, P. Valmaggia, and P. C. Cattin, "Diffusion models for implicit image segmentation ensembles," in *International Conference on Medical Imaging with Deep Learning*, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:244908570.

[14] R. S. Zimmermann, L. Schott, Y. Song, B. A. Dunn, and D. A. Klindt, "Score-based generative classifiers," *ArXiv*, vol. abs/2110.00473, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:238253287.

[15] A. Durrer, J. Wolleb, F. Bieder, *et al.*, "Diffusion models for contrast harmonization of magnetic resonance images," in *International Conference on Medical Imaging with Deep Learning*, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:257532623.

[16] J. Wolleb, F. Bieder, R. Sandkühler, and P. C. Cattin, "Diffusion models for medical anomaly detection," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2022: 25th International Conference, Singapore, September 18–22, 2022, Proceedings, Part VIII*, Singapore, Singapore: Springer-Verlag, 2022, pp. 35–45, isbn: 978-3-031-16451-4. doi: 10.1007/978-3-031-16452-1_4. [Online]. Available: https://doi.org/10.1007/978-3-031-16452-1_4.

[17] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33, Curran Associates, Inc., 2020, pp. 6840–6851. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

[18] D. Baranchuk, I. Rubachev, A. Voynov, V. Khrulkov, and A. Babenko, "Label-efficient semantic segmentation with diffusion models," vol. abs/2112.03126, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:244908617.

[19] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," vol. abs/2010.02502, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:222140788.

[20] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," 2021, pp. 10 674–10 685. [Online]. Available: https://api.semanticscholar.org/CorpusID:245335280.

[21] Z. Wang, Y. Jiang, H. Zheng, *et al.*, "Patch diffusion: Faster and more data-efficient training of diffusion models," vol. abs/2304.12526, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:258309253.

[22] X. Han, Z. Zhang, N. Ding, *et al.*, "Pre-trained models: Past, present and future," vol. abs/2106.07139, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:235421816.

[23] A. Hosna, E. Merry, J. Gyalmo, Z. Alom, Z. Aung, and M. A. Azim, "Transfer learning: A friendly introduction," *Journal of Big Data*, vol. 9, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:253048129.

[24] F. Zhuang, Z. Qi, K. Duan, *et al.*, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, pp. 43–76, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:207847753.

[25] K. R. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big Data*, vol. 3, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:256407097.

[26] A. Argyriou, T. Evgeniou, and M. Pontil, "Multi-task feature learning.," Jan. 2006, pp. 41–48.

[27] J. Gao, W. Fan, J. Jiang, and J. Han, "Knowledge transfer via multiple model local structure mapping," in *Knowledge Discovery and Data Mining*, 2008. [Online]. Available: https://api.semanticscholar.org/CorpusID:17429744.

[28] H. Guan and M. Liu, "Domain adaptation for medical image analysis: A survey," *IEEE Transactions on Biomedical Engineering*, vol. 69, pp. 1173–1185, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:231951465.

[29] T. Mehmood, A. E. Gerevini, A. Lavelli, and I. Serina, "Combining multi-task learning with transfer learning for biomedical named entity recognition," in *International Conference on Knowledge-Based Intelligent Information & Engineering Systems*, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:226257074.

[30] J. Gui, T. Chen, J. Zhang, *et al.*, "A survey on self-supervised learning: Algorithms, applications, and future trends.," *IEEE transactions on pattern analysis and machine intelligence*, vol. PP, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:261046875.

[31] V. R. de Sa, "Learning classification with unlabeled data," in *Neural Information Processing Systems*, 1993. [Online]. Available: https://api.semanticscholar.org/CorpusID:9890353.

[32] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *European Conference on Computer Vision*, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:50698.

[33] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A survey on contrastive self-supervised learning," *CoRR*, vol. abs/2011.00362, 2020. arXiv: 2011.00362. [Online]. Available: https://arxiv.org/abs/2011.00362.

[34] K. He, H. Fan, Y. Wu, S. Xie, and R. B. Girshick, "Momentum contrast for unsupervised visual representation learning," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9726–9735, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:207930212.

[35] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *ArXiv*, vol. abs/1807.03748, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:49670925.

[36] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," *ArXiv*, vol. abs/2002.05709, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:211096730.

[37] X. Liu, F. Zhang, Z. Hou, *et al.*, "Self-supervised learning: Generative or contrastive," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, pp. 857–876, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:219687051.

[38] L. Wu, H. Lin, Z. Gao, C. Tan, and S. Z. Li, "Self-supervised on graphs: Contrastive, generative, or predictive," *CoRR*, vol. abs/2105.07342, 2021. arXiv: 2105.07342. [Online]. Available: https://arxiv.org/abs/2105.07342.

[39] V. Purma, S. Srinath, S. Srirangarajan, A. Kakkar, and P. A.P., "Genselfdiff-his: Generative self-supervision using diffusion for histopathological image segmentation," *ArXiv*, vol. abs/2309.01487, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:261530530.

[40] I. R. I. Haque and J. J. Neubert, "Deep learning approaches to biomedical image segmentation," *Informatics in Medicine Unlocked*, vol. 18, p. 100 297, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:213875226.

[41] F. Isensee, P. F. Jaeger, S. A. A. Kohl, J. Petersen, and K. H. Maier-Hein, "Nnu-net: A self-configuring method for deep learning-based biomedical image segmentation," *Nature Methods*, vol. 18, no. 2, pp. 203–211, 2021, issn: 1548-7105. doi: 10.1038/s41592-020-01008-z. [Online]. Available: https://doi.org/10.1038/s41592-020-01008-z.

[42] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *ArXiv*, vol. abs/1505.04597, 2015. [Online]. Available: https://api.semanticscholar.org/CorpusID:3719281.

[43] J. Huo, O. Xi, S. Ourselin, and R. Sparks, "Generative medical segmentation," *ArXiv*, vol. abs/2403.18198, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:268723596.

[44] C. E. Johanson, J. A. I. Duncan, P. M. Klinge, T. W. Brinker, E. G. Stopa, and G. D. Silverberg, "Multiplicity of cerebrospinal fluid functions: New challenges in health and disease," *Cerebrospinal Fluid Research*, vol. 5, pp. 10 –10, 2008. [Online]. Available: https://api.semanticscholar.org/CorpusID:216101759.

[45] F. Marques, J. Sousa, G. Coppola, *et al.*, "Kinetic profile of the transcriptome changes induced in the choroid plexus by peripheral inflammation," *Journal of cerebral blood flow and metabolism : official journal of the International Society of Cerebral Blood Flow and Metabolism*, vol. 29, pp. 921–32, Mar. 2009. doi: 10.1038/jcbfm.2009.15.

[46] M. Lovelace, B. Varney, G. Sundaram, *et al.*, "Current evidence for a role of the kynurenine pathway of tryptophan metabolism in multiple sclerosis," *Frontiers in Immunology*, vol. 7, Aug. 2016. doi: 10.3389/fimmu.2016.00246.

[47] M. Madhukar, A. K. Choudhary, D. K. B. Boal, M. S. Dias, and M. R. Iantosca, "Choroid plexus: Normal size criteria on neuroimaging," *Surgical and Radiologic Anatomy*, vol. 34, pp. 887–895, 2012. [Online]. Available: https://api.semanticscholar.org/CorpusID:21601963.

[48] A. Yazdan-Panah, M. Schmidt-Mengin, V. A. G. Ricigliano, T. Soulier, B. Stankoff, and O. Colliot, "Automatic segmentation of the choroid plexuses: Method and validation in controls and patients with multiple sclerosis," *NeuroImage : Clinical*, vol. 38, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:257355093.

[49] J. J. Eisma *et al.*, "Deep learning segmentation of the choroid plexus from structural magnetic resonance imaging (mri): Validation and normative ranges across the adult lifespan," *Fluids and Barriers of the CNS*, vol. 21, no. 1, p. 21, 2024. doi: 10.1186/s12987-024-00525-9.

[50] J. N. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," *ArXiv*, vol. abs/1503.03585, 2015. [Online]. Available: https://api.semanticscholar.org/CorpusID:14888175.

[51] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *CoRR*, vol. abs/1312.6114, 2013. [Online]. Available: https://api.semanticscholar.org/CorpusID:216078090.

[52] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," in *Neural Information Processing Systems*, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:196470871.

[53] A. Vaswani, N. M. Shazeer, N. Parmar, *et al.*, "Attention is all you need," in *Neural Information Processing Systems*, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:13756489.

[54] F. Bieder, J. Wolleb, A. Durrer, R. Sandkühler, and P. C. Cattin, "Diffusion models for memory-efficient processing of 3d medical images," *ArXiv*, vol. abs/2303.15288, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:257767010.

[55] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Ng, "Multimodal deep learning," in *International Conference on Machine Learning*, 2011. [Online]. Available: https://api.semanticscholar.org/CorpusID:352650.

[56] S. Sankaran, D. Yang, and S. Lim, "Multimodal fusion refiner networks," *CoRR*, vol. abs/2104.03435, 2021. arXiv: 2104.03435. [Online]. Available: https://arxiv.org/abs/2104.03435.

[57] Z. Niu, G. Zhong, and H. Yu, "A review on the attention mechanism of deep learning," *Neurocomputing*, vol. 452, pp. 48–62, 2021, issn: 0925-2312. doi: https://doi.org/10.1016/j.neucom.2021.03.091. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S092523122100477X.

[58] Z. Ding, M. Zhang, J. Wu, and Z. Tu, "Patched denoising diffusion models for high-resolution image synthesis," *ArXiv*, vol. abs/2308.01316, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:260378901.

[59] R. Liu, J. Lehman, P. Molino, *et al.*, "An intriguing failing of convolutional neural networks and the coordconv solution," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, ser. NIPS'18, Montréal, Canada: Curran Associates Inc., 2018, pp. 9628–9639.

[60] V. Visani, M. Veronese, F. B. Pizzini, *et al.*, "Aschoplex: A generalizable approach for the automatic segmentation of choroid plexus." [Online]. Available: https://api.semanticscholar.org/CorpusID:269138531.

[61] M. Everingham, L. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, pp. 303–338, 2010. [Online]. Available: https://api.semanticscholar.org/CorpusID:4246903.

[62] U. Baid, S. Ghodasara, M. Bilello, *et al.*, "The rsna-asnr-miccai brats 2021 benchmark on brain tumor segmentation and radiogenomic classification," *ArXiv*, vol. abs/2107.02314, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:235742974.

[63] S. Bakas, H. Akbari, A. Sotiras, *et al.*, "Advancing the cancer genome atlas glioma mri collections with expert segmentation labels and radiomic features," *Scientific Data*, vol. 4, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:3697707.

[64] J. Cheng, W. Huang, S. Cao, *et al.*, "Enhanced performance of brain tumor classification via tumor region augmentation and partition," *PLoS ONE*, vol. 10, 2015. [Online]. Available: https://api.semanticscholar.org/CorpusID:2159979.

[65] J. Cheng, W. Yang, M. Huang, *et al.*, "Retrieval of brain tumors by adaptive spatial pooling and fisher vector representation," *PLoS ONE*, vol. 11, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:13961011.

[66] M. Antonelli, A. Reinke, S. Bakas, *et al.*, "The medical segmentation decathlon," *Nature Communications*, vol. 13, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:235390655.

[67] O. Senay, M. Seethaler, N. Makris, *et al.*, "A preliminary choroid plexus volumetric study in individuals with psychosis," *Human Brain Mapping*, vol. 44, pp. 2465 –2478, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:256613621.

[68] M. J. Cardoso, W. Li, R. Brown, *et al.*, "Monai: An open-source framework for deep learning in healthcare," *arXiv preprint arXiv:2211.02701*, Nov. 2022. doi: 10.48550/arXiv.2211.02701.

[69] F. Pérez-García, R. Sparks, and S. Ourselin, "TorchIO: A Python library for efficient loading, preprocessing, augmentation and patch-based sampling of medical images in deep learning," *Computer Methods and Programs in Biomedicine*, p. 106 236, 2021, issn: 0169-2607. doi: https://doi.org/10.1016/j.cmpb.2021.106236. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0169260721003102.

[70] P. T. Jackson, A. Atapour-Abarghouei, S. Bonner, T. P. Breckon, and B. Obara, "Style augmentation: Data augmentation via style randomization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 83–92.

[71] Y. Tang, D. Yang, W. Li, *et al.*, "Self-supervised pre-training of swin transformers for 3d medical image analysis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 20 730–20 740.

[72] J. Choi, J. Lee, C. Shin, S. Kim, H. J. Kim, and S.-H. Yoon, "Perception prioritized training of diffusion models," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11 462–11 471, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:247922317.

[73] W. Yan, L. Huang, L. Xia, *et al.*, "Mri manufacturer shift and adaptation: Increasing the generalizability of deep learning segmentation for mr images acquired with different scanners," *Radiology: Artificial Intelligence*, vol. 2, e190195, Jul. 2020. doi: 10.1148/ryai.2020190195.

[74] T. Lin, M. Maire, S. J. Belongie, *et al.*, "Microsoft COCO: common objects in context," *CoRR*, vol. abs/1405.0312, 2014. arXiv: 1405.0312. [Online]. Available: http://arxiv.org/abs/1405.0312.