

Capstone Project Phase B

**Camouflaged Object Detection with Diffusion Model**

**24-2-R-5**

Students:

Hagar Tibi – 209063411 – [Hagar.Tibi@e.braude.ac.il](file:///C:\Users\user\AppData\Local\Microsoft\Windows\INetCache\IE\8OPL4U6S\Hagar.Tibi@e.braude.ac.il)

Roi Darom – 313264822 – [Roi.Darom@e.braude.ac.il](file:///C:\Users\user\AppData\Local\Microsoft\Windows\INetCache\IE\8OPL4U6S\Roi.Darom@e.braude.ac.il)

Supervisors: Dr. Renata Avros & Prof. Zeev Volkovich

**Table of Contents**

[**Abstract 3**](#_Toc187927482)

[**1. Introduction 3**](#_Toc187927483)

[**2. Related Work 4**](#_Toc187927484)

[**2.1 Camouflaged Object Detection Models 4**](#_Toc187927485)

[**2.1.1. Edge Semantic Information 4**](#_Toc187927486)

[**2.2.1. Multi-task Learning 5**](#_Toc187927487)

[**2.2.2. Transformer-based Methods 5**](#_Toc187927488)

[**2.3 Current Diffusion methods 6**](#_Toc187927489)

[**2.3.1. MedSegDiff 6**](#_Toc187927490)

[**2.3.2. ODISE 7**](#_Toc187927491)

[**3. Background 7**](#_Toc187927492)

[**3.1. Image representation 7**](#_Toc187927493)

[**3.2. Image segmentation 8**](#_Toc187927494)

[**3.3 Image restoration 8**](#_Toc187927495)

[**3.4. Diffusion Mathematical Background 9**](#_Toc187927496)

[**3.4.1. Markov Chain 9**](#_Toc187927497)

[**3.4.2. Noise 10**](#_Toc187927498)

[**3.4.3. Mask 10**](#_Toc187927499)

[**3.4.4. Forward Process Formulation 10**](#_Toc187927500)

[**3.4.5. Reverse Process Formulation 11**](#_Toc187927501)

[**3.5. Convolutional Neural Network 12**](#_Toc187927502)

[**3.6. U-net Architecture 13**](#_Toc187927503)

[**3.6.1. U-net block 15**](#_Toc187927504)

[**3.6.2. Up-sampling and Down-sampling 16**](#_Toc187927505)

[**3.7. Framework Components 16**](#_Toc187927506)

[**3.7.1. Vision Transformer (ViT) 16**](#_Toc187927507)

[**3.7.2. Feature Fusion (FF) 17**](#_Toc187927508)

[**3.7.3. Injection Attention Module (IAM) 18**](#_Toc187927509)

[**4. Product 20**](#_Toc187927510)

[**5. Dataset 21**](#_Toc187927511)

[**6. Research Process 22**](#_Toc187927512)

[**6.1. Process 22**](#_Toc187927513)

[**6.2 . Research 22**](#_Toc187927514)

[**7. Models 23**](#_Toc187927515)

[**8. Results 26**](#_Toc187927516)

[**9. Testing the model 28**](#_Toc187927517)

[**10. Conclusions 30**](#_Toc187927518)

[**11. User Documents 31**](#_Toc187927519)

[**11.1 Instruction 31**](#_Toc187927520)

[**11.2Diagrams 34**](#_Toc187927521)

[**12. References 34**](#_Toc187927522)

### **Abstract**

This paper explores the adaptation of the DiffCOD (Diffusion Camouflaged Object Detection) model for tumor detection in brain MRI images. The framework leverages diffusion models to identify tumor regions by utilizing their capability to reverse noise effects and extract meaningful features from noisy or degraded inputs. During training, the model learns to recognize and localize tumors by analyzing patterns of degradation and noise in MRI images, while the inference process refines noisy inputs into diagnostic-quality outputs highlighting tumor regions. The framework incorporates an Injection Attention Module (IAM) to integrate conditional semantic features that enhance tumor detection and a Feature Fusion (FF) module to combine multi-scale features, ensuring critical diagnostic details are preserved. This approach is particularly relevant for medical imaging, where accurate tumor detection is vital for effective diagnosis and treatment planning, and demonstrates the potential of diffusion-based frameworks in advancing healthcare technologies.

Refer to the project files and source code: [GitHub](https://github.com/HagarTibi/Image-Restoration-with-Diffusion-Model-24-2-R-5.git)

### **Introduction**

Camouflaged object detection (COD) identifies objects blending seamlessly with their surroundings, posing a challenge due to their high similarity with backgrounds. This task is vital in fields like agricultural pest detection, medical image segmentation, and industrial defect detection. Traditional COD methods, often inspired by human vision, utilize convolutional neural networks and auxiliary cues to improve accuracy. However, these methods struggle with the complex nature of camouflaged objects.  
Recently, diffusion models have excelled in tasks like image synthesis by learning the reverse diffusion process, though their use in COD is not well-explored. diffCOD approaches COD as a denoising diffusion process, adding Gaussian noise to ground-truth masks during training, which the model learns to reverse. During inference, it refines noisy masks into accurate segmentation through forward-and-reverse diffusion steps. diffCOD enhances denoising by incorporating encoded input image priors and integrating semantic features using a cross-attention-based Injection Attention Module (IAM).

In addition to camouflaged object detection, this project aims to modify the diffCOD model to restore and enhance image quality of damaged images. By applying the denoising diffusion process, the model can effectively reduce various types of noise and interferences, resulting in clearer and higher-quality images. This enhancement is crucial for applications where image quality is compromised due to aging, environmental factors, or other sources of degradation.

תמונה שמכילה טקסט, צילום מסך, יונק

התיאור נוצר באופן אוטומטי

[***Figure 1****. mainstream COD approach vs Diffusion based approach*](#link1)[*[1]*](#link1)

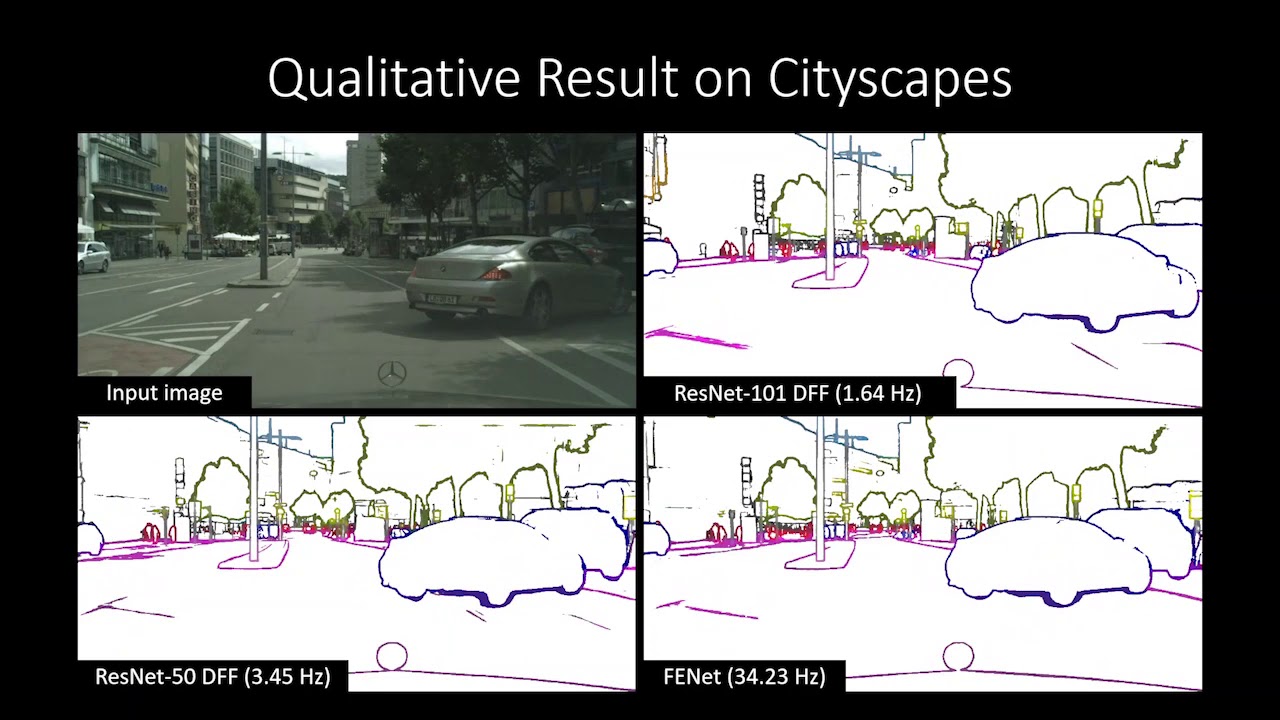
### **2. Related Work**

### **Camouflaged Object Detection Models**

Camouflaged object detection (COD) involves identifying objects that blend into their surroundings. Traditional COD methods, which are non-generative, segment objects directly from the background using various strategies. These strategies often focus on enhancing feature representation and improving segmentation accuracy through:

### **2.1.1. Edge Semantic Information**

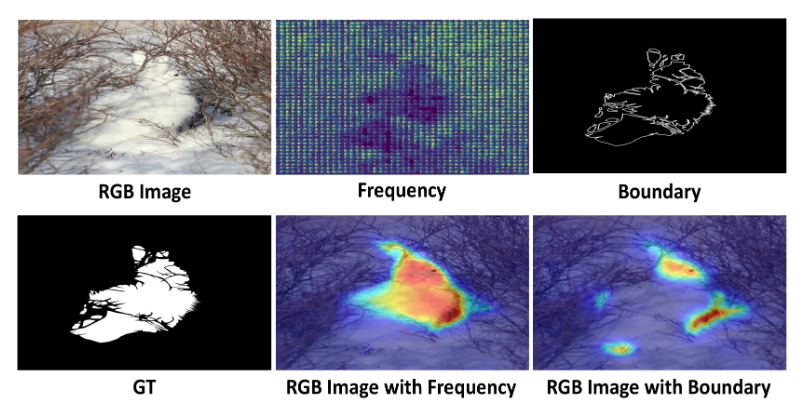
Methods such as BGNet for edge semantic information highlight object structures and boundaries by introducing additional cues. Edge semantic information involves identifying and emphasizing the edges or boundaries of objects within an image. BGNet (Boundary Guided Network) leverages this information to enhance object detection and segmentation by providing clearer delineation of objects. This is especially useful in complex scenes where precise edge detection is crucial for accurate object recognition.



***[Figure 2](#link2)****[. Semantic Edge Detection Network Example [2]](#link2)*

**2.2 Frequency Domain Features**

Methods that utilize frequency domain features enhance detection accuracy by focusing on specialized information, like diffCOD's use of encoded input image priors to improve mask refinement. Frequency domain features, as implemented in methods like FDCOD, enhance detection accuracy by analyzing the frequency components of an image. These methods transform image data into the frequency domain, allowing for the examination of periodic patterns and structures that may not be easily visible in the spatial domain. By leveraging these features, they can identify and emphasize important characteristics, leading to more accurate object detection.



[***Figure 3****. FDCOD* with *different additional cues. [3]*](#link3)

### **Multi-task Learning**

Methods that utilize multi-task learning enhance detection accuracy by performing multiple related tasks simultaneously, which can lead to improved performance on each individual task. By sharing information across tasks, these methods can learn more robust and generalized features. For example, SegMaR leverages multi-task learning to enhance object detection by combining segmentation and recognition tasks.

תמונה שמכילה טקסט, צילום מסך, תרשים, קו

Description automatically generated

[***Figure 4****. Uncertainty-based loss function weighting for multi-task learning . [4]*](#link4)

### **Transformer-based Methods**

Transformer is A type of deep learning model that has revolutionized the fields of natural language processing (NLP) and, more recently, computer vision. It is designed to handle sequential data and can capture long-range dependencies through a mechanism known as attention.

FSPNet, for example, employ transformers to process the input data in a way that accounts for both local and global information. This enables them to effectively identify and differentiate between objects, even in complex and cluttered environments.

Transformer-based methods are also highly effective for image restoration. By capturing long-range dependencies and relationships within the data, these models can maintain consistency and coherence across the entire image. This is crucial for tasks such as noise reduction and detail enhancement, ensuring that the restored images retain both local details and global context, resulting in higher visual fidelity and overall quality.

תמונה שמכילה טקסט, צילום מסך, גופן, תרשים

Description automatically generated

[***Figure 5****. transformer architeture simplified. [5]*](#link5)

### **2.3 Current Diffusion methods**

Existing COD methods often struggle with accurate segmentation in complex scenarios. To address these challenges, generative models, specifically denoising diffusion models, are introduced into the COD task. These models progressively refine object masks from noisy images, achieving exceptional performance, particularly for objects with fine textures. Recent advances in diffusion models have shown promising results in various segmentation tasks

### **MedSegDiff**

MedSegDiff is a diffusion-based method designed for medical image segmentation. By leveraging diffusion models, it improves segmentation accuracy by capturing complex patterns and structures within medical images, which are often challenging to delineate using traditional methods. This approach allows for more precise and detailed segmentations, enhancing the overall diagnostic capabilities in medical imaging.

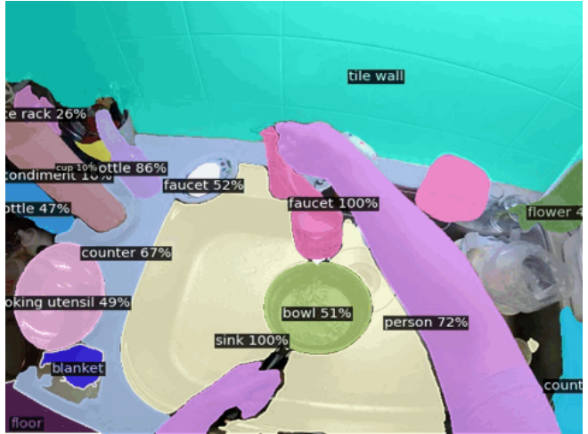
תמונה שמכילה טקסט, צילום מסך, תרשים, עיצוב

התיאור נוצר באופן אוטומטי

[***Figure 6****. Diffusion Based medical image segmentation. [6]*](#link6)

### **ODISE**

ODISE combines a trained text-image diffusion model with a discriminative model to achieve open-vocabulary panoptic segmentation, allowing for broader segmentation capabilities. This method enables the segmentation of a wide range of object categories by leveraging the strengths of both generative and discriminative approaches, thus enhancing the versatility and accuracy of the segmentation process in diverse scenarios.

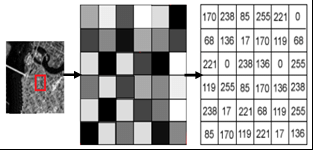


[***Figure 7****. visual result of  Open-vocabulary DIffusion-based panoptic Segmentation [7]*](#link7)

### **Background**

### **Image representation**

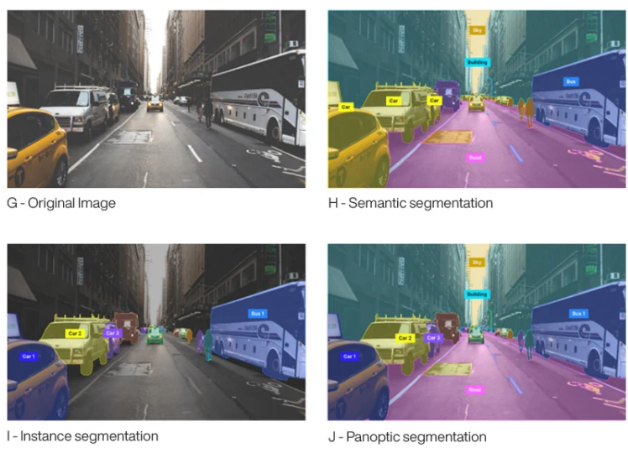
Image representation involves converting a picture into a format that computers can process and understand. Pictures consist of tiny dots called pixels, each with a specific color or brightness value. These pixels are organized in a grid format that defines the image's dimensions and color information, such as the RGB format for color images. In computational terms, an image can be represented as a two-dimensional array arranged in rows and columns, with x and y coordinates representing the pixel values, typically ranging from 0 (black) to 255 (white). The goal of image representation is to extract significant details from the picture, enabling the computer to perform tasks like object recognition, image classification, or segmentation.



[***Figure 8****. image representation by pixels [8]*](#link8)

### **Image segmentation**

Image segmentation is a crucial task in computer vision that involves partitioning an image into multiple segments or regions, each representing different objects or areas within the scene. This process transforms a pixel-level representation of an image into a higher-level abstraction, facilitating tasks such as object detection, recognition, and analysis. Techniques for image segmentation can be broadly categorized into classical methods and modern deep learning approaches. Classical methods include thresholding, edge detection, and region-based techniques, which rely on pixel intensity and local features. In contrast, deep learning-based methods, particularly convolutional neural networks (CNNs) and transformer models, leverage hierarchical feature extraction and end-to-end learning to achieve superior accuracy and robustness. Advanced techniques such as fully convolutional networks (FCNs), U-Net, Mask R-CNN, and semantic segmentation models incorporate multi-scale feature (multi-scale features capture information at various levels of detail within an image, from fine textures to broader structures, enabling more comprehensive analysis and understanding) learning, skip connections, and contextual information to effectively segment complex scenes with high precision.



[***Figure 9****. Differenet types of image segmentation. [9]*](#link9)

### **Image restoration**

Image restoration is a fundamental task in computer vision that involves reconstructing or recovering an image that has been degraded by noise, blur, or other distortions. The objective is to restore the image to its original or near-original state, thereby enhancing its quality and making it suitable for further analysis and processing. Image restoration techniques are crucial for various applications, including medical imaging, satellite imagery, historical document preservation, and general photography. Image restoration techniques aim to reverse image degradations using various algorithms and models, categorized into classical methods and modern deep learning approaches. Classical methods include filtering techniques like mean, median, and Wiener filtering to reduce noise and enhance quality; deconvolution to reverse blurring; and interpolation methods such as bilinear and bicubic to restore lower resolution images. Modern deep learning approaches encompass convolutional neural networks (CNNs) that learn complex patterns to remove noise and artifacts, autoencoders that compress and reconstruct images, and generative adversarial networks (GANs) that generate high-quality restored images by learning the distribution of clean images.

### **Diffusion Mathematical Background**

The diffusion probability model has reaped plenty of attention due to its simple training process and excellent performance. It is mainly divided into forward process and reverse process.

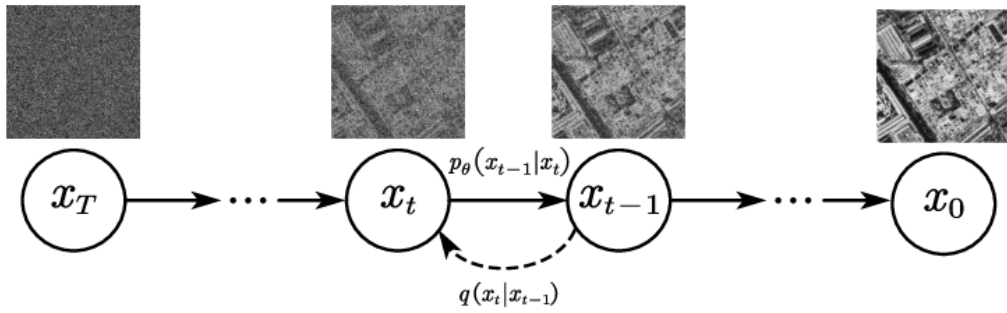
### **Markov Chain**

A Markov chain is a mathematical system that undergoes transitions from one state to another within a finite or countable number of possible states. It is characterized by the Markov property, which states that the future state of the process depends only on the present state and not on the sequence of events that preceded it. Formally, this can be expressed as:

where represents the state at step n.

A key feature of Markov chains is their use in both forward and reverse processes.

In the context of diffusion models, Markov chains are utilized to iteratively refine noisy images through a series of probabilistic transitions. The forward process involves adding Gaussian noise to the data, and the reverse process, modeled as a Markov chain, progressively removes this noise. This denoising process can be described by a sequence of states where is the original data, is the noise, and intermediate states represent the noisy versions of the data. The goal is to learn the transition probabilities that effectively reverse the noise addition, leading to the recovery of the original data from the noisy observations.



[***Figure 10.*** *Denoising diffusion model bases on Markov chain, where q and t represent adding and removing noise respectively. [10]*](#link10)

### **Noise**

Noise is essentially disturbances that obscure or interfere with the intended signal for certain data. Noise is created by environmental sources or electronic interference. In the field of image processing, noise manifests as unwanted pixel values, leading to loss of brightness and details in the image.

* Gaussian Noise

Gaussian (normal) distributed random noise is applied to the gray values of original images. Gaussian noise is commonly used in modelling scenarios due to its mathematical properties and prevalence in real-world noise sources. The mathematical properties of this distribution allow averaging over many pixels to help detect and cancel out the noise.

* Speckle Noise

Speckle noise (Multiplicative Noise) is commonly found in radar and medical imaging, particularly in ultrasound images. It is a granular noise that occurs due to the interference of multiple wave reflections. Unlike Gaussian noise, which is additive, speckle noise is multiplicative, meaning it affects the intensity of the pixels proportionally.

* Salt-and-Pepper Noise

This type of noise manifests as random occurrences of black and white pixels within an image, resembling the look of salt and pepper. It is caused by sudden and sharp disturbances in the image signal, such as faulty memory locations or malfunctioning pixels in camera sensors.

### **Mask**

A binary or multi-class image that indicates specific regions of interest within the original image. Each pixel in the mask corresponds to a pixel in the original image and holds a value that signifies whether that pixel belongs to the objects of interest (foreground) or the background.

### **Forward Process Formulation**

The diffusion model operates through two main processes: the forward process and the reverse process. In the forward process, Gaussian noise with variance ∈ (0,1) is gradually added to the original image transforming it step by step until it becomes a completely noisy image that resembles an isotropic Gaussian distribution (symmetric around its mean). This process is described by the equation:

breakdown of the parameters:

* : The noisy image at time step t.
* : The noisy image at the previous time step t−1.
* : the probability of moving to state given the previous state .
* N: Denotes a Gaussian distribution.
* : The mean of the Gaussian distribution, which is a scaled version of .
* : The covariance matrix of the Gaussian distribution, with controlling the amount of noise added at each step and being the identity matrix indicating isotropic (equal in all directions) noise.

The latent variable ​ can be directly obtained from the original image ​ using:

breakdown of the parameters:

* : The noisy image at time step t.
* : The original image.
* : The probability distribution of ​ given the original image ​.
* : The mean of the Gaussian distribution, which is a scaled version of the original image ​.
* : The covariance matrix of the Gaussian distribution, indicating the amount of noise added to the original image.

here and . that means can be seen as a noisy version of ​, where the noise level increases with time t.

This project transitions from camouflaged object detection (COD) to tumor detection in brain MRI images by adapting an existing diffusion model. The goal is to leverage the model's capacity to understand noise patterns and extract meaningful features to identify and delineate tumors in MRI scans. The forward process, originally designed to add Gaussian noise for image detecting camouflaged objects, is now utilized to simulate subtle variations in MRI images that can help the model learn the nuanced features associated with tumor regions. Parameters like and ​ play a crucial role in preserving diagnostically significant features while handling noise, ensuring that the model focuses on identifying abnormalities. By training on MRI datasets, the model learns to reverse the noise effects and accurately highlight tumor boundaries, repurposing the diffusion process for robust and precise tumor detection. This adaptation ensures the framework is capable of handling diverse MRI characteristics, aiding in reliable medical diagnostics.

### **Reverse Process Formulation**

The reverse process involves converting the noisy image back into the original image through a series of steps. This is formulated as:

Here is the probability of transitioning back to state from with and being learned parameters that adjust the mean and variance of the distribution.

breakdown of the parameters:

* : The noisy image at time step t.
* : The noisy image at the previous time step t−1.
* : The mean function parameterized by , which predicts the mean of the Gaussian distribution for the reverse step.
* : The covariance (variance) function parameterized by θ, which predicts the variance of the Gaussian distribution for the reverse step.

The combination of the forward process q and the reverse process p forms a variational auto-encoder. The objective is to minimize the variational lower bound (VLB), which measures the difference between the true data distribution and the model distribution. The VLB is defined as:

Each term in the VLB represents a different aspect of the learning process:

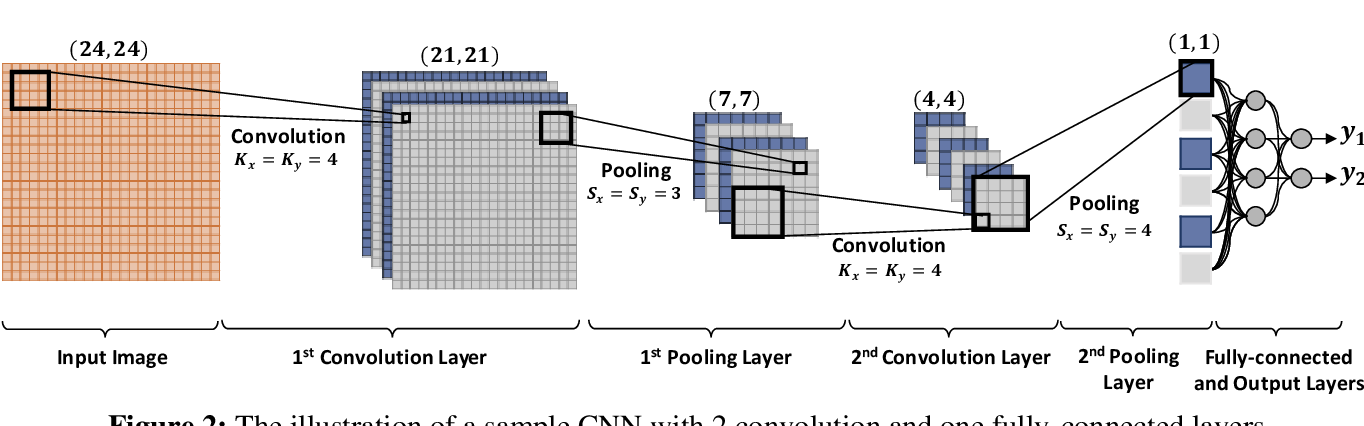
* measures the likelihood of the original image given the first step in the reverse process, ​. It ensures that the reverse process starts off correctly reconstructing ​ from the initial noisy image.

This is the KL divergence between the true posterior distribution and the learned reverse process distribution . It measures how well the reverse process matches the forward process at each step t.

* measures how well the final noisy image matches the assumed Gaussian distribution. It ensures that the final noisy state is consistent with the Gaussian noise model.

### **Convolutional Neural Network**

A Convolutional Neural Network (CNN) is a specialized type of artificial neural network designed primarily for analyzing visual data. CNNs are widely used in image and video recognition, as well as in other areas such as natural language processing. This network mimics the structure of neurons in the human brain and is highly effective for tasks like image segmentation and classification.



**[Figure 11.](#link11)** [Convolution Neural network [11]](#link11)

The architecture of a CNN comprises five key components:

Input Layer:

This layer represents the image based on its pixel values, typically divided into RGB (Red, Green, and Blue) channels.

Convolution Layer:

This primary layer extracts feature maps from the image. During each iteration, a filter of size H×W×D (height, width, depth) is initialized to represent the weights. The filter convolves over the image, computing dot products and connecting pixels into a single neuron. An activation function is then applied to each neuron to facilitate visual representation and user comprehension.

Hyperparameters used in the Convolution Layer:

**Stride:** Determines the step size between filter applications over the image, controlling the displacement of the filter across the image's length and width.

**Padding:** Adds extra pixels, usually zeros, at the image borders to preserve spatial dimensions and maintain edge features.

**Batch Size:** Specifies the number of samples passing through the network in each forward/backward iteration, affecting training stability and speed.

**Activation Function:** Introduces non-linearity to each neuron, enabling the network to capture and represent complex relationships among data pixels, thus enhancing feature maps.

Pooling Layer: This layer completes the feature extraction process by reducing the dimensions of the feature maps, making them more manageable.

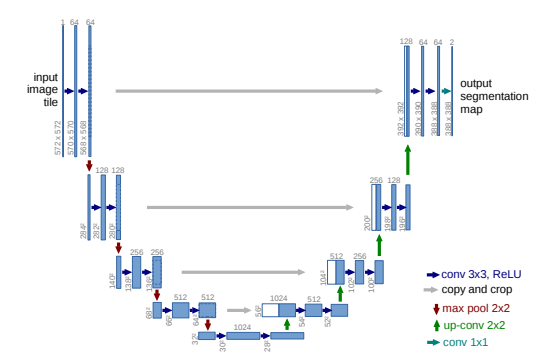
Fully Connected Layer: Combines and normalizes the features extracted from previous layers. It contains neurons that connect to the entire input volume.

Output Layer: Composed of neurons arranged in a grid, with the number of neurons determined by the task requirements. Each neuron represents the probability of the input pixel belonging to a specific category.

### **U-net Architecture**

U-Net is a convolutional neural network architecture specifically designed for biomedical image segmentation. The network is structured in a symmetric U-shape, consisting of a contracting path (encoder) and an expansive path (decoder).

* Encoder Path: The downsampling part of U-Net that reduces the image size while extracting features using convolutional and max pooling layers.
* Decoder Path: The upsampling part of U-Net that reconstructs the image to its original size, refining features with upsampling and convolutional layers.
* Bottleneck: The central part of U-Net where the feature maps are most compressed, serving as the transition between the encoder and decoder.
* Skip Connections: Links between corresponding layers in the encoder and decoder, allowing high-resolution features from the encoder to be combined with the upsampled features in the decoder, preserving spatial information.



**[Figure 12.](#link12)** [Architecture of U-net model [12]](#link12)

Now let us look at the flow of the U-net network:

* Encoder Path:

Structure: Composed of multiple convolutional layer blocks with max-pooling layers in between.

Function: Captures high-level features and reduces the spatial dimensions of the input image through down-sampling operations.

* Decoder Path:

Structure: Composed of multiple convolutional layer blocks with up-sampling layers in between.

Function: Increases the spatial dimensions of the feature maps, expanding them back to the original image size.

* Skip Connections:

Function: Connects corresponding layers in the encoder and decoder paths, allowing the network to recover spatial details lost during down-sampling by directly concatenating feature maps from the encoder to the decoder.

* Final Layer:

Structure: Typically, a 1x1 convolutional layer.

Function: Reduces the number of channels to produce a final segmentation map with the same dimensions as the input image. This is followed by an activation function for semantic segmentation tasks.

* Training:

Loss Function: The loss function (also known as a cost function or objective function) is a mathematical function that takes in the model's predictions and the actual values and outputs a single number representing the difference (or error). Lower values of the loss function indicate better model performance.

The Parameters that used to Improve the Loss:

**Model Architecture**: Choosing and fine-tuning the structure of the model (e.g., number of layers in a neural network, types of layers).

**Data Augmentation**: Enhancing the training dataset with transformations like rotations, flips, and scaling to improve generalization.

**Cross-Validation**: Using different subsets of data to validate the model performance and ensure it generalizes well to unseen data.

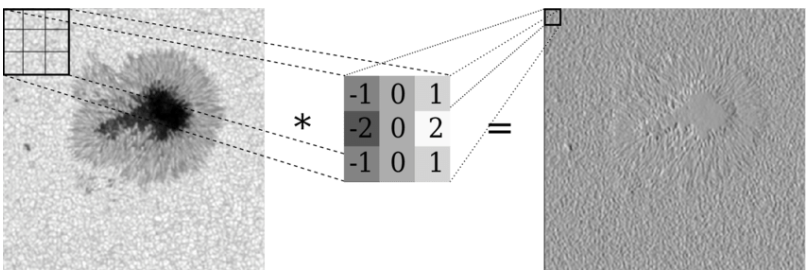
**Hyperparameter Tuning**: Systematically searching for the best hyperparameters (e.g., using grid search, random search, or Bayesian optimization).

### **U-net block**

The U-Net block utilizes convolutions like a standard CNN block. However, instead of applying a single filter, the U-Net block performs multiple convolutions (typically 2-3) followed by activation functions to learn more complex features. The result of these convolutions is a feature map that captures specific aspects of the image based on the applied filters.

Each U-Net block comprises two identical components:

1. Convolutional: A 3x3x filter is applied to the image. In the encoding layer, the number of feature maps increases, while in the decoding layer, they decrease. The filter weights are adjusted during model training.
2. Activation Function (ReLU): The Rectified Linear Unit (ReLU) activation function returns 0 for negative inputs and keeps positive inputs unchanged. This introduces non-linearity with minimal computational overhead.

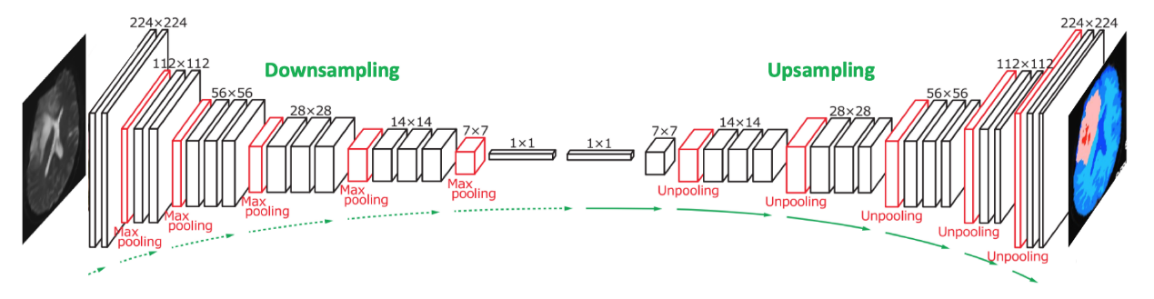


[**Figure 13**. convolution with a filter [13].](#link13)

### **Up-sampling and Down-sampling**

Up-sampling and Down-sampling are techniques used to alter the size of an image. Up-sampling, also known as pooling, reduces the image size by half, while down-sampling increases the image size.

1. Down-Sampling: Typically performed using max-pooling. This method involves applying a filter of a specific size to the image and selecting the maximum value within each filter region.
2. Up-Sampling: Several methods can be employed to increase the image size:
   1. Nearest Neighbors: The pixel values are duplicated to adjacent pixels based on the filter size.
   2. Bed of Nails: Pixels are placed at specific intervals determined by the filter size, consistently across the entire image.
   3. Max Up-Pooling: The maximum value chosen during down-sampling is placed at the same index during the up-sampling step.
   4. Pixel Shuffle: Pixels within each feature map are rearranged. Each set of
   5. 𝑟×𝑟 elements in the channel (where 𝑟 is the scaling factor) is reshaped into a single element in a new, enlarged height and width dimension.



**[Figure 14.](#link14)** [Up and Down sampling [14]](#link14)

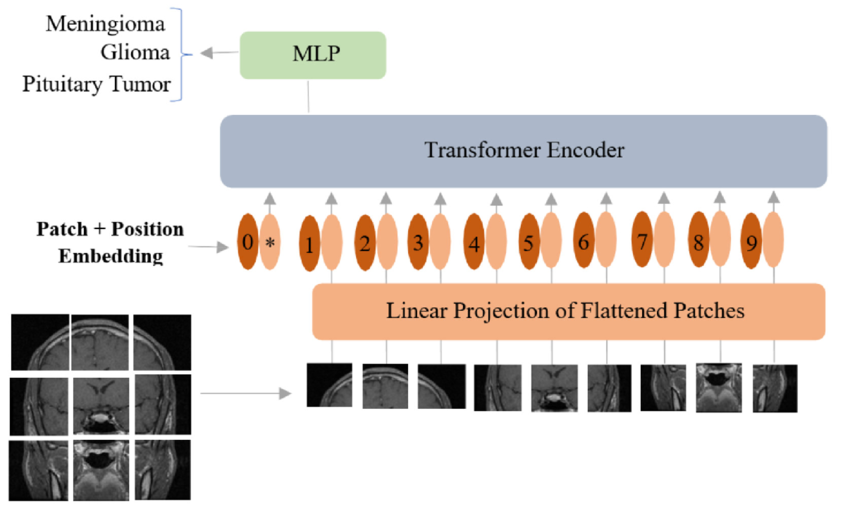
### **Framework Components**

The diffCOD framework utilizes a diffusion model to tackle the camouflaged object detection (COD) task. The core denoising network of diffCOD is built on a U-Net architecture. To obtain robust conditional semantic features, the model integrates multi-scale features extracted from a Vision Transformer (ViT) backbone, combined through a feature fusion (FF) process. This approach ensures that the resulting features are rich in multi-scale details. Furthermore, to enable the texture patterns and localization information in the conditional semantic features to effectively guide the denoising process, an injection attention module (IAM) based on cross-attention is introduced. This IAM reduces the discrepancy between diffusion features and image features, leveraging the strengths of both to enhance performance.

### **Vision Transformer (ViT)**

The Vision Transformer (ViT) is an architecture that adapts the transformer model, originally designed for natural language processing, to image processing tasks. It divides an image into a sequence of smaller patches. Each patch is flattened and embedded into a fixed-size vector, serving as an input token for the transformer model.

The ViT process begins with embedding these image patches and adding positional encodings to retain spatial information. These embeddings are then passed through multiple transformer layers, which include multi-head self-attention mechanisms and feed-forward neural networks. The self-attention mechanism allows the model to capture long-range dependencies and contextual relationships across different parts of the image, effectively modelling global information.



[**Figure 15.** Vision Transformer mode adopted for classification of brain tumors from MRI.  
MLP (multilayer perceptron): is the extra learnable patch embedding to be used by the final classification head. [15]](#link15)

The primary purpose of ViT is to leverage the power of transformers in capturing complex patterns and relationships across an entire image. This approach leads to improved performance in various vision tasks, such as image classification, segmentation, and object detection, by providing a global understanding of the image content.

### **Feature Fusion (FF)**

Feature Fusion (FF) is a technique used in machine learning and computer vision to combine features from multiple sources or scales to enhance the performance of a model. In image processing, features at different scales capture varying types of information. Low-level features capture edges and textures, while high-level features capture shapes and objects. By integrating these multi-scale features, feature fusion combines fine details with broader contextual information, enriching the overall representation. This integration is especially useful in tasks such as image restoration, segmentation, and object detection, where both local and global contexts are crucial.

Feature fusion employs techniques like concatenation, addition, weighted sum, and attention mechanisms to combine features. Concatenation joins features from different sources or scales along a specified dimension, while addition combines them elementwise, maintaining the same dimensionality. The weighted sum approach involves learned weights during training, allowing dynamic adjustment of each feature set's importance. Attention mechanisms weigh features based on relevance, focusing the model on the most critical aspects.

תמונה שמכילה טקסט, צילום מסך, עיגול

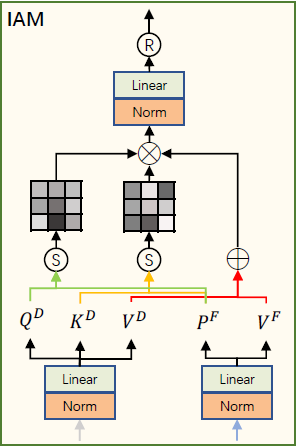
Description automatically generated

[**Figure 16.** fusing MR and CT images using local extreme map guided filtering, feature extraction, and feature fusion (FF) to enhance image quality by combining bright, dark, and base feature maps. [16]](#link16)

In the diffCOD model, FF processes the multi-scale features extracted by the Vision Transformer (ViT) backbone. The FF module has three branches, each handling one of the feature scales (Xp1, Xp2, Xp3). Each branch applies two convolution operations to enhance the features. The features from the three branches are then aggregated using a single convolution operation, resulting in a unified feature map. By leveraging FF, the diffCOD model can effectively combine multi-scale and complementary features to create richer and more robust representations. This enhanced representation significantly improves the model's performance in detecting and segmenting camouflaged objects.

### **Injection Attention Module (IAM)**

The Injection Attention Module (IAM) enhances the denoising process in the diffusion model by integrating texture and location information from the original features. The IAM, positioned in the middle of the U-Net-based denoising network and uses cross-attention mechanisms to merge multi-scale features effectively.



[***Figure 17****: IAM architecture. [17]*](#link17)

The IAM takes two inputs: the multi-scale fusion feature F from the Feature Fusion (FF) module and the deepest feature D from the diffusion model.

These inputs undergo the following process:

* 1. **Feature Transformation**  
     The deepest feature D is linearly projected to produce the query ​, key ​, and value ​. The fusion feature F is linearly projected to generate ​, and ​​. Unlike typical attention mechanisms, F does not generate queries and keys for direct similarity comparison, instead, it uses ​, ​ for this purpose. Those are formulated by:

- The query matrix derived from the deepest feature D. - The key matrix derived from the deepest feature D.

- The value matrix derived from the deepest feature D.

- The intermediary projection of F, used for similarity comparison.

- The value matrix derived from the fusion feature F.

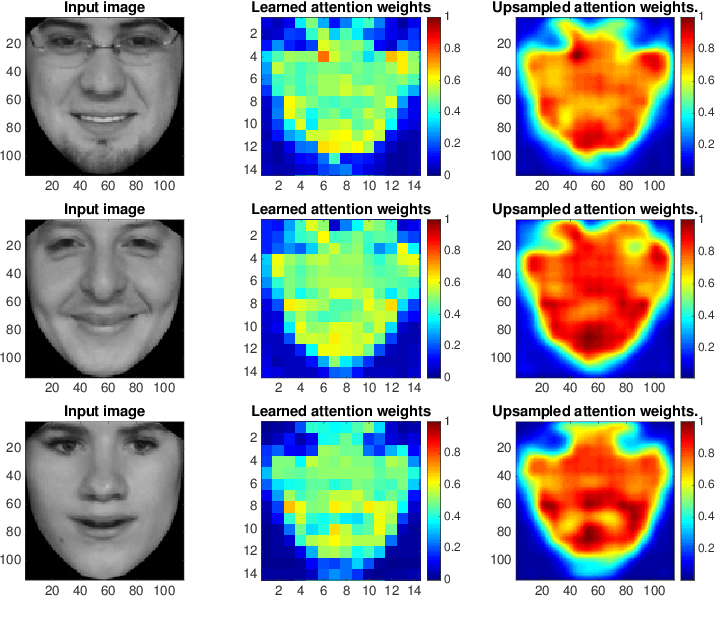
Where are the learned projection matrices, and d is the dimensionality of these projections.

* 1. **Similarity Computation**

The similarity between the query ​ and the intermediary projection ​ is as computed as followed:

Similarly, the similarity between the key ​ and the intermediary projection ​ is computed:

These similarity scores are normalized using the SoftMax function, resulting in attention maps ​ and ​(Attention maps are a core component of the attention mechanism in machine learning models, especially in tasks like image processing and natural language processing. They help the model understand which parts of the input data are most important for making accurate predictions or decisions).



***[Figure 18](#link18)****[: Heat map visualization of learned attention. [18]](#link18)*

* 1. **Attention Weight Application**The weighted values are combined to form the final output:

The output feature map serves as a refined representation that combines the strengths of both the multi-scale fusion features and the deep diffusion features. By leveraging cross-attention, effectively integrates texture and localization information, making it highly effective for tasks that require detailed and context-aware feature representations.

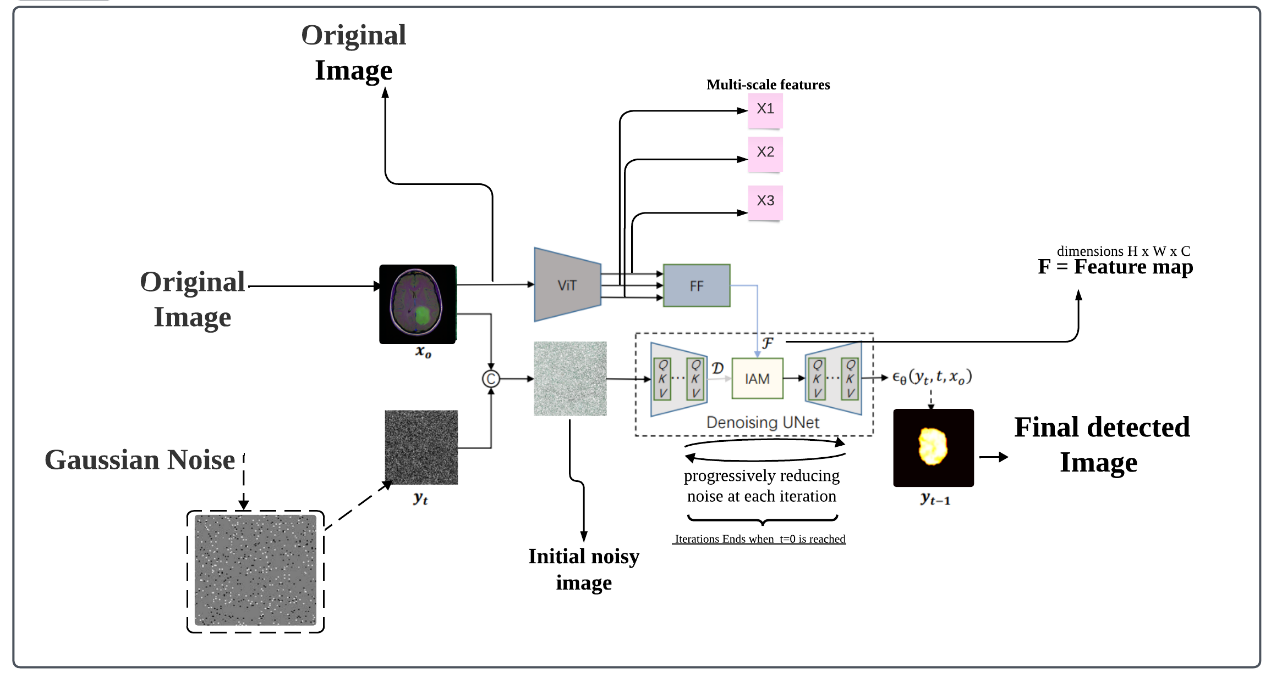
### **Product**

The product repurposes the existing Diffusion Model for Camouflaged Object Detection (diffCOD) to address the challenge of Tumors Detection in medical imaging. While diffCOD was originally designed to detect camouflaged objects, we utilize its core denoising diffusion process for a different application.

By leveraging its U-Net backbone and the Injection Attention Module (IAM), The mechanisms originally designed to detect subtle features in camouflaged objects have been adapted to identify intricate patterns and anomalies associated with tumors in brain MRI scans. This enables the model to effectively highlight and localize tumor regions for diagnostic purposes.

A key component in this adaptation is the IAM, which integrates semantic features derived from a Vision Transformer (ViT) and a Feed Forward (FF) module. This integration enables the IAM to enhance feature extraction and attention mechanisms, guiding the model to focus on tumor-specific anomalies in brain MRI scans. Through this approach, the model leverages its original design strengths while addressing the unique requirements of identifying and localizing tumors. This repurposing of diffCOD enables it to analyze intricate patterns in MRI data, making it a valuable tool for medical diagnostics and advancing its versatility beyond its original design.

To evaluate this approach, we use brain MRI datasets, which provide real-world examples of complex medical images where precise anomaly detection is critical. By applying diffCOD to this specialized domain, we demonstrate its effectiveness in identifying tumor regions and improving diagnostic accuracy.



**Figure 19.** Workflow of the model. The input feeds a given image concatenating with "Gaussian" noise into a denoising diffusion model with a U-Net architecture as the core component for denoising. An injection attention module (IAM) is designed to implicitly guide the diffusion process with the conditional semantic features that have gone through the Vision Transformer (ViT) and the Feed Forward (FF) module, emphasizing the importance of the FF in processing and refining features. This allows the model to take full advantage of the correspondence between image features and diffusion information, ultimately achieving effective tumor detection in brain MRI images.

### **Dataset**

For the project, the “LGG MRI Segmentation” [[20]](https://www.kaggle.com/datasets/mateuszbuda/lgg-mri-segmentation) Dataset from Kaggle was utilized, a collection of MRI scans designed for segmentation tasks. To tailor the dataset for the project’s specific goals, a thorough cleaning process was conducted. Images with empty ground truth (GT) were identified and removed, as they do not provide the necessary information for effective training. This step was crucial in ensuring that the dataset used was both relevant and high-quality.

Given the dataset's relatively small size, data augmentation techniques to increase diversity and improve model performance were initially explored. However, the experiments revealed that augmentation and duplication reduced the model's effectiveness, leading us to revert to the original dataset without additional modifications.

After these adjustments, the final dataset comprises 1,365 images, with 960 images allocated for training and 400 for testing. Each image is resized to a consistent dimension of 352x352 pixels, ensuring uniformity for input into the model. This cleaned dataset forms the basis for training and evaluating the model.

### **Research Process**

### **6.1 Process**

The main goal of this project is to adapt the CamoDiffusion model to handle medical images, specifically brain MRI scans, for tumor detection. This involves leveraging diffusion models to learn effective representations of brain structures and anomalies, allowing for accurate segmentation and detection of tumors. By extending the capabilities of CamoDiffusion, the goal is to improve its application to medical imaging, which poses unique challenges such as the variability in tumor shapes and sizes and the sensitivity required for precise detection.

In Part A of the project, the foundational aspects of diffusion models and their suitability for camouflaged object detection were explored. We then transitioned to researching how these models could be extended to medical imaging, with a particular focus on brain MRI scans. This phase included preprocessing medical images to normalize pixel intensities, segmenting regions of interest, and implementing initial versions of the CamoDiffusion pipeline for tumor detection.

In Part B, the emphasis shifted to refining the CamoDiffusion model and its components. This included a detailed study of the model’s architecture, integrating domain-specific knowledge of medical imaging, and applying noise-handling techniques. Key challenges included handling varying levels of image quality and noise in MRI scans and ensuring the model’s sensitivity to detect small or irregular tumors. To address these challenges, the training process was modified by introducing medical imaging augmentations and adjusting the diffusion process to better handle the nuances of MRI scan data.

The performance evaluation of the adapted CamoDiffusion model occurred by testing it on datasets containing brain MRI scans with varying tumor characteristics. These datasets were chosen to represent diverse conditions, including different tumor sizes, locations, and intensities. The primary focus of this phase was to assess how well the model performs in detecting tumors under various conditions and noise levels.

### **6.2 Research**

To begin the research, various methodologies and architectures, such as CamoDiffusion and other diffusion-based models, were investigated. The primary focus was on evaluating the effectiveness of the diffusion-based approach in the context of medical imaging, particularly for tumor detection in brain MRI scans. The model was trained and tested on brain MRI datasets, undergoing a systematic evaluation process to determine its suitability for the task.

To optimize the model for tumor detection, several training parameters and techniques were tested. Key adjustments included:

* **Hyperparameter Tuning**: Systematical adjustments of hyperparameters such as the batch size and optimizer settings. Additionally, different numbers of epochs were checked to observe their impact on the model’s convergence and performance.
* **Data Augmentation:** Techniques such as rotation, flipping, scaling, and intensity adjustments were applied to increase the diversity of the training data, thereby improving the model’s generalization capabilities.

Throughout the training process, the performance of the models was evaluated using two key metrics: Training Loss and Mean Absolute Error (MAE). These metrics were monitored under various training conditions to assess the impact of parameter adjustments, including noise types and hyperparameter configurations. By systematically analyzing Training Loss and MAE, insights were gained into the models' convergence behavior and their ability to generalize across noisy and clean datasets, ultimately optimizing the detection of tumors in brain MRI scans.

### **Models**

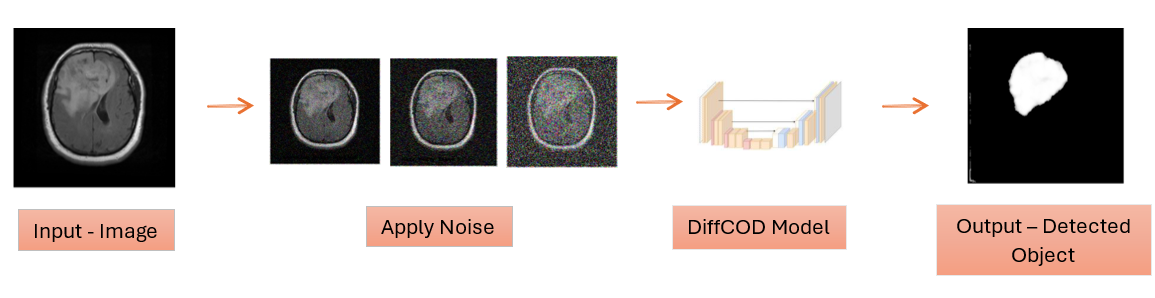
Workflow for Camouflaged Object Detection

Input: Grayscale or RGB Image

* The input image size used is 352×352×1 or 352×352×3 .

**Adding Synthetic Noise**  
In this phase, synthetic noise is artificially introduced to the dataset containing images. Various types of synthetic noise are added, and the model is trained for each one of them:

1. Gaussian Noise
2. Salt-and-Pepper Noise



***Figure .*** *Model Flow*

Model Architecture: CamoDiffusion

The CamoDiffusion model integrates conditioning mechanisms with a diffusion process, allowing precise reconstruction of camouflaged objects. It employs a Pyramid Vision Transformer (PVT-V2) backbone for feature extraction and a robust decoder for object segmentation. The model incorporates the following key components:

1. Conditioning Mechanisms:
   * Conditional UNet-based architecture for guiding the diffusion model using auxiliary inputs.
   * Vision Transformer (ViT) blocks process hierarchical feature maps with multi-scale attention.
2. Diffusion Process:
   * A cosine-based noise schedule is used to control forward and reverse diffusion.
   * Timestep embeddings enhance the model's understanding of diffusion stages.
3. Decoder:
   * Multi-scale feature fusion using residual blocks and up-sampling.
   * Integration of learned positional embeddings for spatial alignment.

Key Features

* The convolutional layers use a kernel size of (3×3) with zero padding to maintain spatial dimensions.
* Residual connections are introduced for stable gradient flow.
* Pixel Shuffle is used for efficient up-sampling without increasing the parameter count.

Implementation: Written in Google Colab, with limited GPU resources.

Hyperparameters

* Input size: 352
* Batch size: 16
* Optimizer: AdamW
* Learning rate: 0.0001
* Epochs: [3,10]
* Loss Function: Structure loss with IoU and weighted BCE components.
* Noise Augmentation:
  + Gaussian noise with standard deviation of 0.1.
  + Salt & Pepper noise - random max/min intensity pixels with probability 0.1.

Diffusion Model Parameters

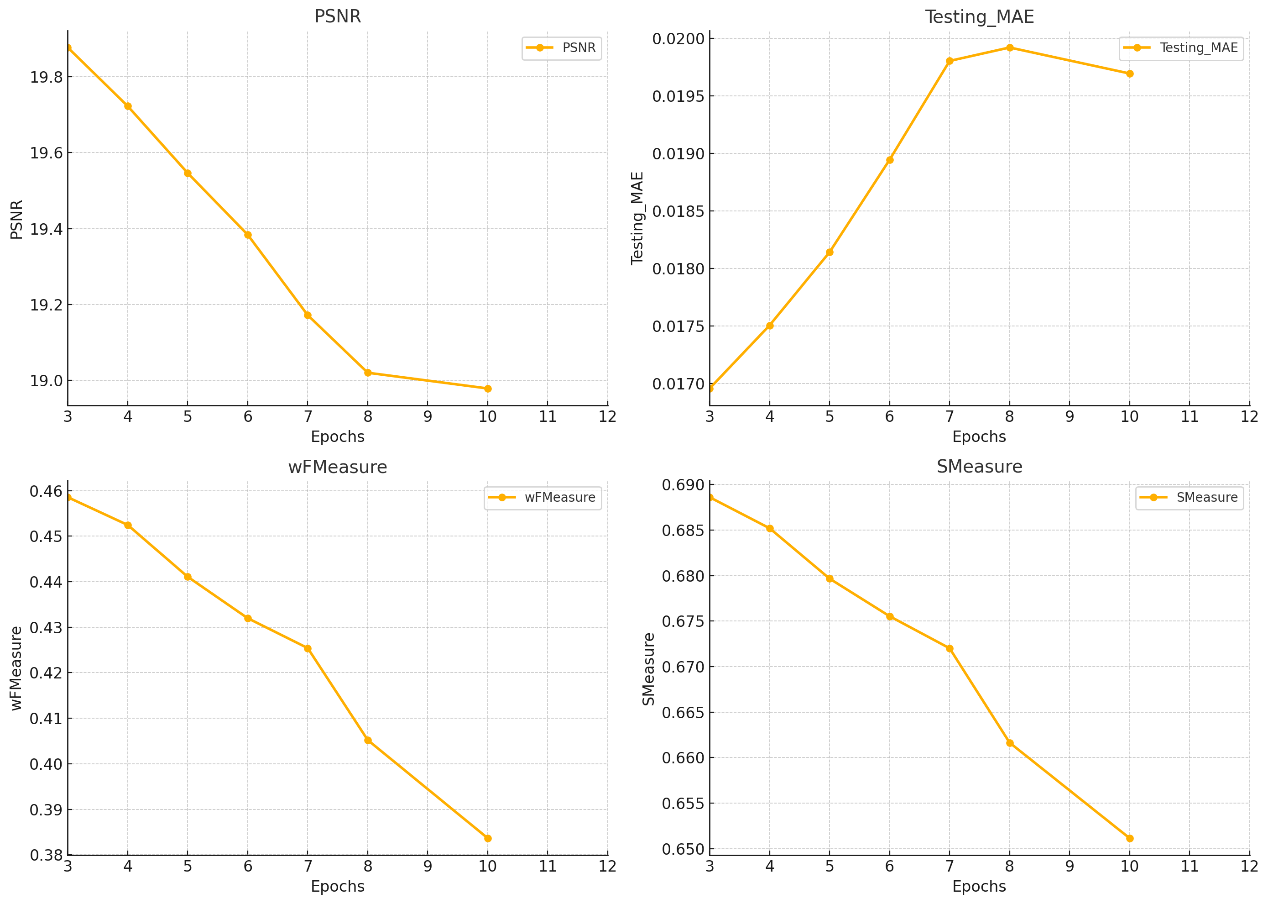
| **Parameter** | **Original Values** | **New Values Set 1** | **New Values Set 2** | **New Values Set 3** | **New Values Set 4** | **Parameter Explanation** |
| --- | --- | --- | --- | --- | --- | --- |
| Channels | 3 | 3 | 3 | 3 | 3 | Number of color channels in the input image. |
| Sampling Steps | 32 | 64 | 128 | 32 | 64 | Number of steps for image refinement during sampling. |
|  | 0.002 | 0.001 | 0.0005 | 0.01 | 0.01 | Minimum noise level in the diffusion process. |
|  | 80 | 50 | 40 | 50 | 50 | Maximum noise level in the diffusion process. |
|  | 0.5 | 0.6 | 0.7 | 0.7 | 0.6 | Standard deviation of the data distribution. |
|  | 7 | 5 | 4 | 5 | 7 | Controls the shape of the noise schedule curve. |
| ​ | -1.2 | -1.0 | -1.0 | -1.0 | -1.2 | Mean of the log-normal distribution for noise sampling. |
|  | 1.2 | 1.0 | 1.0 | 1.0 | 1.2 | Standard deviation of the log-normal distribution. |
|  | 80 | 50 | 30 | 30 | 100 | Controls stochastic sampling strength. |
|  | 0.05 | 0.1 | 0.2 | 0.2 | 0.1 | Minimum timestep for stochastic effects in sampling. |
|  | 50 | 40 | 30 | 20 | 40 | Maximum timestep for stochastic effects in sampling. |
|  | 1.003 | 1.001 | 1.001 | 1.0005 | 1.01 | Noise scale factor for stochastic sampling. |

Architecture Plot

| Layer | Function | Output Shape (Θ×W) | Channels (C) |
| --- | --- | --- | --- |
| 0 | Input layer | (352×352) | 1 or 3 |
| 1 | Overlap Patch Embedding | (176×176) | 64 |
| 2 | Transformer Block (Stage 1) | (176×176) | 64 |
| 3 | Transformer Block (Stage 2) | (88×88) | 128 |
| 4 | Transformer Block (Stage 3) | (44×44) | 320 |
| 5 | Transformer Block (Stage 4) | (22×22) | 512 |
| 6 | Decoder Fusion | (352×352) | 256 |
| 7 | Up-sampling | (352×352) | 96 |
| 8 | Prediction Head (Conv2D) | (352×352) | 1 |

### **Results**

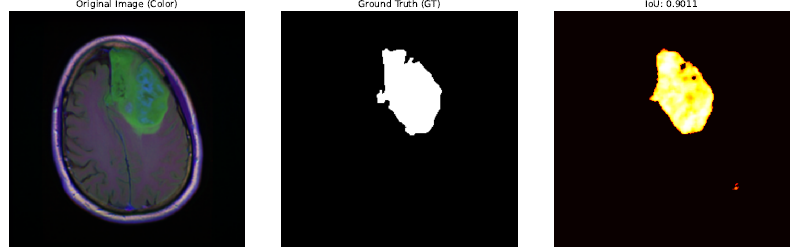
Evaluation metrics – Gaussian noise



***Figure .*** *Evaluation metrics- MAE, wFMeasure, PSNR and SMeasure.*

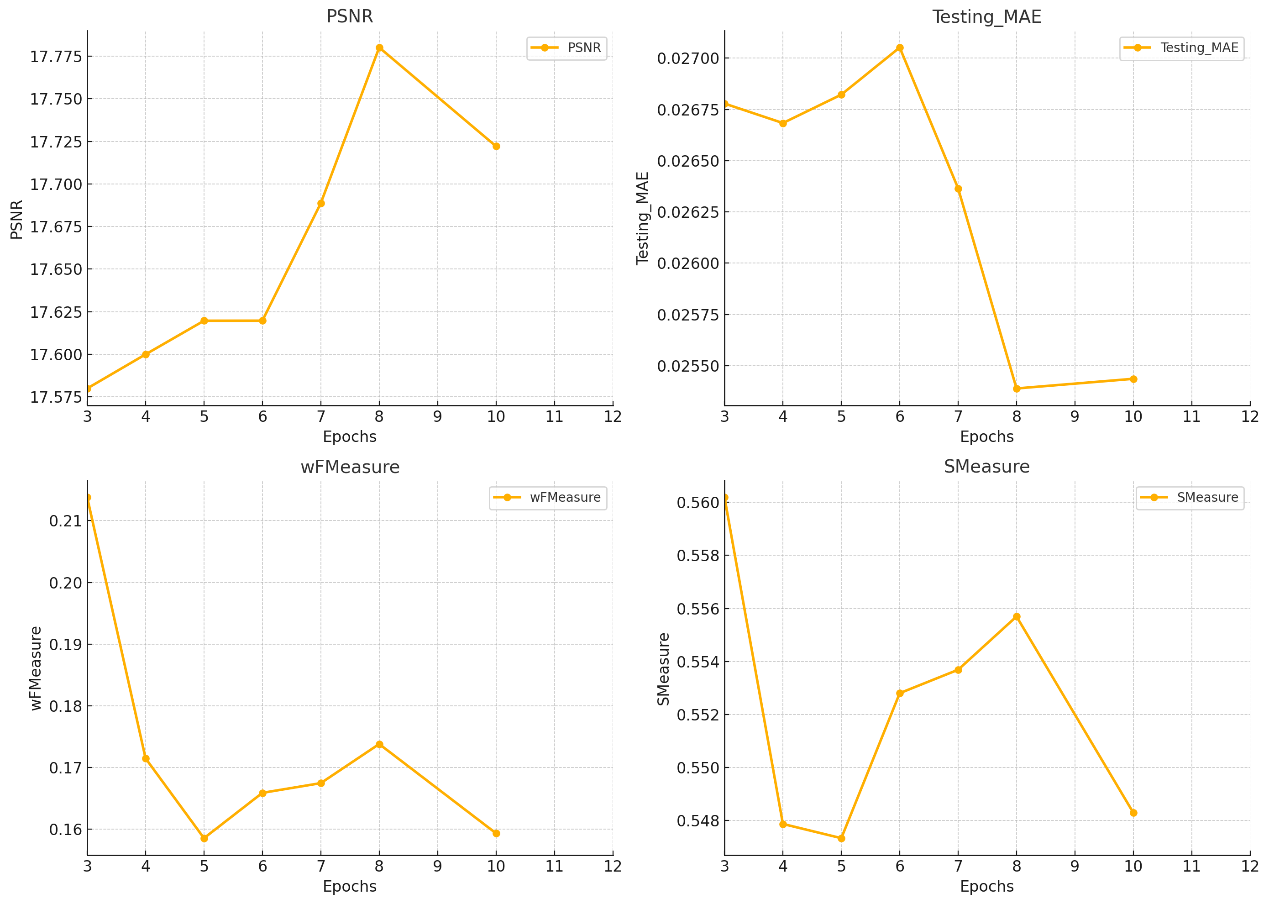
*תמונה שמכילה צילום מסך, עיגול

Description automatically generated*Examples:

**

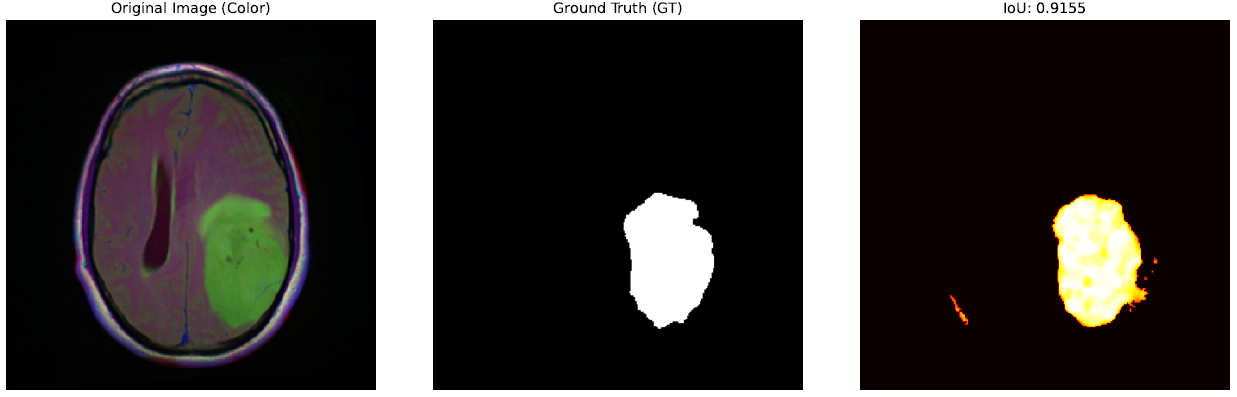
**

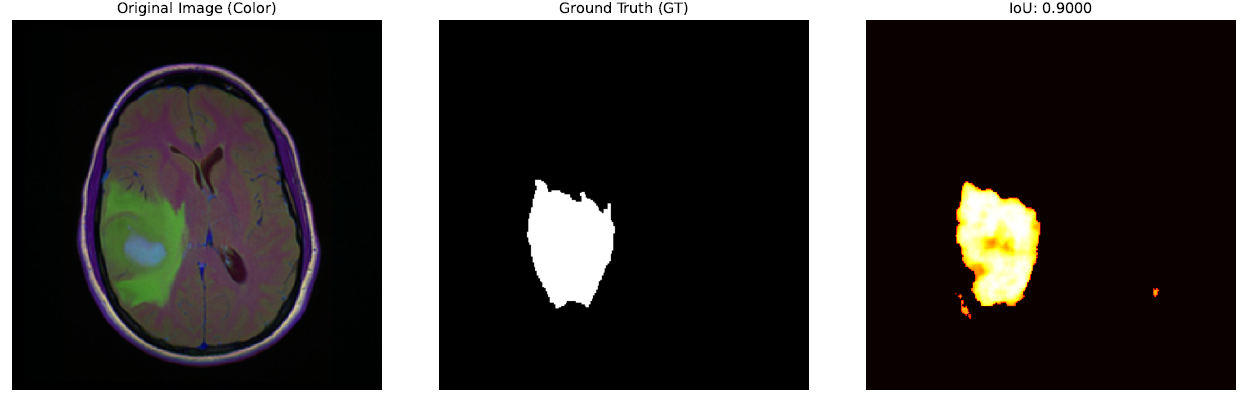
Evaluation metrics – Salt & Pepper noise



***Figure .*** *Evaluation metrics- MAE, wFMeasure, PSNR and SMeasure.*

Examples







### **Testing the model**

The performance of the CamoDiffusion model was evaluated using four key metrics:

**PSNR** (Peak Signal-to-Noise Ratio) - Measures the reconstruction quality of the output image compared to the ground truth. Higher PSNR indicates better image restoration and segmentation quality. This metric was used to assess how well the model preserved structural details under varying noise conditions.

**MAE** (Mean Absolute Error) - Evaluates the pixel-wise error between the predicted segmentation and the ground truth. A lower MAE value indicates better segmentation accuracy.

**wFMeasure** (Weighted F-measure) - Combines precision and recall into a single metric, weighted to account for segmentation balance. This metric is particularly useful for validating segmentation consistency, especially in boundary regions.

**Smeasure** (Structure Measure) - Balances region-based and object-based segmentation evaluation to provide a comprehensive assessment of the segmentation quality. It considers both pixel accuracy and structural similarity to the ground truth.

Gaussian noise experiments:



Salt & Pepper noise experiments:



### **Conclusions**

Considering the constraints of the dataset size and computational resources, the adapted diffusion-based model shows promising potential for restoring damaged images and reducing noise effectively, even when working with a limited number of training samples.

During evaluation, the variability in the model's performance depending on the extent and type of image degradation was observed. In some cases, particularly with severe distortions, the model faced challenges in almost fully detecting the original MRI image. Moreover, the model exhibited difficulties in accurately identifying the boundaries of objects, which impacted its ability to preserve finer structural details. This suggests that the model’s ability to generalize across varying levels of image degradation could be improved with more diverse and comprehensive training scenarios.

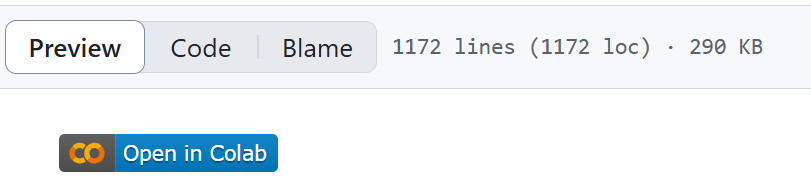
The model demonstrated results for most images in the dataset, effectively enhancing quality and clarity. However, its performance is inherently influenced by the dataset’s resolution (352x352), and scaling to significantly higher resolutions might require additional adjustments to the architecture or training process.

In conclusion, the primary limitations are tied to the size of the dataset and available computational resources. Expanding the dataset, incorporating more diverse degradation scenarios, increasing training iterations, and optimizing GPU usage would likely enhance the model's performance and robustness, enabling it to better address a wider range of restoration challenges, including the accurate delineation of object boundaries.

### **User Documents**

### **Instruction**

This guide outlines the steps for using the system provided in the Google Colab notebook, designed to adapt the DiffCOD model for tumor detection in brain MRI images. The system enables training, evaluation, and visualization of the model's performance. Follow the instructions below to successfully operate the system.

* **Clone the Project repository from GitHub:**
  + git clone [GitHub project link](https://github.com/HagarTibi/Image-Restoration-with-Diffusion-Model-24-2-R-5.git)
  + cd C:\your\path\to\Camouflaged-Object-Detection-With-Diffusion-Model-For-MRI-Images
* **Google Drive Setup**
  + Enter to the following google drive link: [**Source Code - DiffCOD**](https://drive.google.com/drive/folders/1gO6o32tK2__6XSiKExiXFUmo_Q0EpQ-O?usp=drive_link)
  + **Upload** the `CamoDiffusion-main` folder to your Google Drive.
  + Open the `diffCOD.ipynb` file in Google Colab. You can do this in two ways:
    - On GitHub web, go to the GitHub repository, click on `diffCOD.ipynb`, and select "Open in Colab" at the top of the page.
    - From ‘Camouflaged-Object-Detection-With-Diffusion-Model-For-MRI-Images’ folder, **upload** ‘diffCOD.ipynb’ to your google Drive.
  + In case you choose option 1: continue to “Getting Started” section
  + In case you choose option 2: go to Google Colab -> file -> Open notebook -> choose the ipynb you saved on option 2 -> go to “Getting Started” section
* **Getting Started** (The numbering of the headings aligns with the numbering assigned to each cell in the Colab notebook)
  1. Mount Google Drive

To ensure access to all required files and configurations, mount your Google Drive by running the following code snippet:

Navigate to the project directory on google drive and verify the current working directory:

תמונה שמכילה טקסט, צילום מסך, גופן, אלגברה

התיאור נוצר באופן אוטומטי

* 1. Install Requirements

Install the required Python packages for the project by running:

תמונה שמכילה טקסט, גופן, צילום מסך, לבן

התיאור נוצר באופן אוטומטי

Dependencies

- **PyTorch** (1.12 or higher)

- **NumPy** (1.21 or higher)

- **OpenCV** (4.5 or higher)

- **Matplotlib** (3.5 or higher)

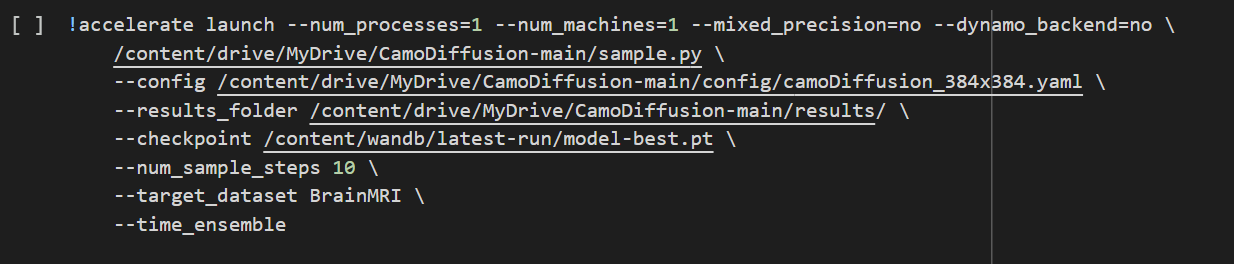
* 1. Training the Model cell

תמונה שמכילה טקסט, צילום מסך, גופן, קו

התיאור נוצר באופן אוטומטיStart training the model using the specified configuration file. Update the parameters such as **epochs** and **batch** size as needed:

* 1. Sample Test

Run the following command to generate sample results using the trained model:



* 1. Results - Metrics & Visualization

To visualize the model's outputs alongside the ground truth and original images, run the visualization script.

Visualizations are saved in the `/visualizations` folder on google colab notebook. These images clearly show the original input, ground truth, and restored results.

* 1. Maintenance - prepare folders for next run

Evaluate the model’s performance using metrics like **IoU** (Intersection over Union), **Smeasure**, and **PSNR** (Peak Signal-to-Noise Ratio) by

running the script for calculate Smeasure and PSNR (it computes the average metrics for the generated results)

The results, including average IoU, Smeasure, and PSNR values, provide insights into the model's performance.

* **Dataset Requirements**

The model uses the **BrainMRI** dataset, which must be structured correctly and stored in your Google Drive under the `/media/BrainMRI` folder. Ensure the dataset contains:

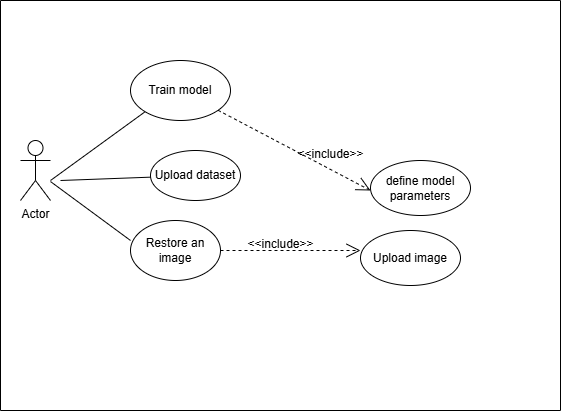
- Only image files (e.g., `.png`, `.jpg`).

- Folder paths and file names are in English.

By following these steps, you (the user) can successfully operate the image restoration system, train the model, generate results, and evaluate its performance.

### **Diagrams**

Use Case Diagram:



***Figure 20:*** *Use Case Diagram*

Sequence Diagram:

תמונה שמכילה תרשים, תוכנית, שרטוט טכני, קו

התיאור נוצר באופן אוטומטי

***Figure 21:*** *Sequence Diagram*

### **12. References**

[1]. Chen, Z., Gao, R., Xiang, T.-Z., & Lin, F. Diffusion Model for Camouflaged Object Detection. School of Informatics, Xiamen University, Xiamen, China; Department of Computer Science and Engineering, HKUST, Hong Kong, China; G42, Abu Dhabi, UAE.

[2]. Zhou, Y., Ge, R., McGrath, G., & Loianno, G. FENet: Fast Real-time Semantic Edge Detection Network. Presented at New York University, Tandon School of Engineering, in collaboration with Qualcomm Technologies Inc.

[3]. Chen, Z., Zhang, X., Xiang, T.-Z., & Tai, Y. Adaptive Guidance Learning for Camouflaged Object Detection. School of Intelligence Science and Technology, Nanjing University, Suzhou, China; College of Computer Science, Nankai University, Tianjin, China; G42, Abu Dhabi, UAE. Correspondence to: T.-Z. Xiang and Y. Tai

[4]. Kendall, A., Gal, Y., & Cipolla, R. (2017). Multi-Task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics

[5]. https://medium.com/@tech-gumptions/transformer-architecture-simplified-3fb501d461c8

[6]. <https://www.semanticscholar.org/paper/MedSegDiff-V2%3A-Diffusion-based-Medical-Image-with-Wu-Fu/3f77a62ae888c3b816eabd354a6dd0fc6b9528ea>

[7]. https://arxiv.org/abs/2303.04803

[8]. <https://www.researchgate.net/figure/Digital-image-representation-by-pixels-vii_fig2_311806469>

[9]. <https://www.telusinternational.com/insights/ai-data/article/guide-to-image-segmentation>

[10]. <https://www.researchgate.net/figure/Denoising-diffusion-model-forward-and-backward-process-qxtx-t-1-p-th-x-t-1-xt_fig1_369946317>

[11] 1D convolutional neural networks and applications: A survey Serkan Kiranyaz a,⇑ , Onur Avci b , Osama Abdeljaber c , Turker Ince d , Moncef Gabbouj e , Daniel J. Inman f

[[12](https://ebraude-my.sharepoint.com/personal/hagar_tibi_e_braude_ac_il/Documents/Capstone_Project_Phase_A-update.docx#Unet)] Ronneberger, Olaf, Fischer, Philipp, and Brox, Thomas. "U-net: Convolutional networks for biomedical image segmentation". MICCAI, 9351:234–241, 2015.

[13]. Enhancing SDO/HMI images using deep learning - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/An-example-of-a-convolution-with-a-filter-In-this-example-a-vertical-border-locating\_fig1\_317543623 [accessed 20 Jul 2024]

[14] Up and Down sampling - <https://mriquestions.com/upsampling.html>.

[15]. <https://www.researchgate.net/figure/Vision-transformer-model-adopted-for-classification-of-brain-tumors-from-MRI-MLP_fig2_364269781>

[16].<https://www.frontiersin.org/journals/neuroscience/articles/10.3389/fnins.2022.1055451/full>

[17] Chen, Z., Gao, R., Xiang, T.-Z., & Lin, F. Diffusion Model for Camouflaged Object Detection. School of Informatics, Xiamen University, Xiamen, China; Department of Computer Science and Engineering, HKUST, Hong Kong, China; G42, Abu Dhabi, UAE

[18]. <https://www.researchgate.net/figure/The-heat-map-visualization-of-the-learned-attention-weights-by-our-spatial-attention_fig4_321306916>