JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

AI Model for Bio-Medical Image Segmentation using U-Shaped Neural Architecture

Mukund sharma[1], Harshita shetty[2], Aman shetty[3], Vinay somani[4], Professor Ashraf Siddiqui[5]
[1] *Computer engineering, Universal college of engineering, India*
[2] *Computer engineering, Universal college of engineering, India*
[3] *Computer engineering, Universal college of engineering, India*
[4] *Computer engineering, Universal college of engineering, India*
[5] *Computer engineering, Universal college of engineering, India*

Abstract
The proposed AI image segmentation using the U-Net architecture aims to address the challenge of accurate and efficient medical image analysis and Segmentation. Medical image segmentation involves identifying and delineating specific regions of interest within medical images, such as tumors, major organs, blood vessels, or other anatomical structures. This task is crucial for clinical diagnosis, treatment planning, and monitoring Medical image segmentation is critical in the proper analysis and diagnosis of a wide range of diseases and ailments. This project's approach to medical picture segmentation that makes use of a "U"-shaped convolutional neural network architecture known as a U-Net. The proposed U-Net architecture takes advantage of its characteristic "U"-shaped design, which includes a contracting path for capturing contextual information and an expansive path for exact localization. This design enables the network to effectively recognize complicated structures and boundaries present in medical pictures by facilitating the extraction of both high-level and low-level elements. This approach might exhibit extraordinary skill in effectively segmenting medical images by utilizing the power of deep learning and neural networks, offering Better, Faster and Accurate results for diagnosis.

keywords:AI Image Segmentation,U-Net Architecture, Convolutional Neural Networks (CNNs), Computer-Aided Diagnosis (CAD),Medical Imaging Modalities (e.g., MRI, CT, X-ray)

1. Introduction

Identifying and isolating particular structures or regions within medical images is known as medical image segmentation, and it is a crucial step in both healthcare and medical imaging. Healthcare experts may precisely identify and localize abnormalities in the human body, such as tumors, lesions, or organ structures, thanks to medical image segmentation. For precise diagnosis and early disease detection, this accuracy is essential. Segmentation aids in the planning of treatments after a medical issue has been identified. Segmented images help surgeons, radiation therapists, and other medical professionals plan and carry out procedures precisely while limiting harm to healthy tissues. Clinicians can monitor the progression of diseases over time thanks to medical image segmentation. By contrasting segmented images from several time points, medical experts can assess the effectiveness of a treatment and make the necessary adjustments.
Recent advancements in machine learning, especially deep learning, have greatly enhanced CAD(Computer aided diagnosis) systems. Deep learning models can automatically extract complex features from images and perform intricate pattern recognition, leading to improved accuracy. The integration of artificial intelligence (AI) into CAD is a growing trend. AI-powered CAD systems can continuously learn and adapt, becoming even more accurate over time.

1.1 Overview of the proposed U-Net architecture

The U-Net architecture is a type of convolutional neural network (CNN) that was created especially for tasks involving semantic segmentation in the study of medical images. In 2015, researchers at the University of Freiburg's Department of Computer Science presented it. The U-shaped architecture of the "U-Net," which comprises an encoding path and a decoding path, is where the name "U-Net" originates.

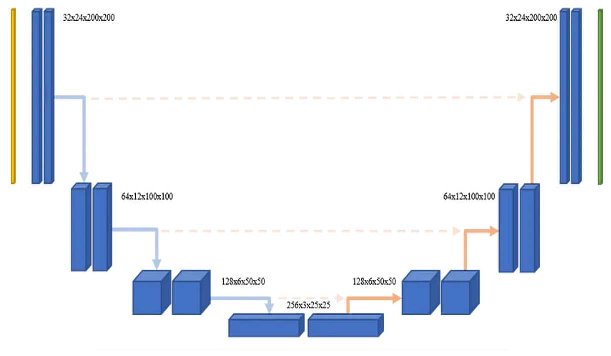


Figure 1:U-net architecture visualized example

We employed U-net architecture to ensure correct predictions and masks which further transfer to better segmentation. But before we can segment the image we need to classify if the image has abnormality or not, for that we use a Transformers. Transformers is an innovative architecture that was initially developed for natural language processing and has found wide use in classification tasks across a variety of fields. Input 4 encoding is the first step in the procedure. Here, inputs like text, sequences or images are transformed into embeddings that include semantic and positional information. Transformers use a multi-layer architecture with feedforward neural networks and self-attention to identify complex correlations in the input data. Their capacity to distinguish various elements of the input is improved by multi-head attention, and the addition of a classification head on top of the encoder makes predictions for classification tasks more accurate and precise.

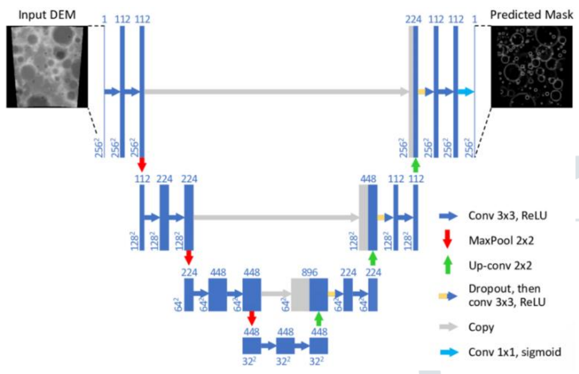


Figure 2: Example image of Practical application of working of a "U" shaped neural network for a masking segmentation task for microscopic images.

1.2 Purpose and objectives of the research

1. Medical Diagnosis and Treatment: This AI model's main goal is to help doctors diagnose and treat a variety of medical diseases. For a precise diagnosis and effective treatment planning, biomedical image segmentation can help detect and separate particular regions of interest within medical pictures, such as tumors, organs, or blood vessels.
2. Improved Efficiency and Precision: The U-shaped neural network design, often known as a U-Net, is particularly suited for biomedical image segmentation because it can capture minute details while preserving spatial context. This results in improved precision and efficiency. The goal is to increase the segmentation process' accuracy and effectiveness, lowering its margin of error and giving medical practitioners more time.
3. Automation and Workflow Improvement: AI models can improve healthcare practitioners' workflow by automating the picture segmentation process. This frees up radiologists and other medical specialists from spending a lot of time manually defining structures in images and allows them to concentrate on higher-level activities, interpretation, and decision-making.
4. Early Disease Detection: It's important to identify diseases and abnormalities as soon as possible. Artificial intelligence (AI) models can assist in spotting small alterations in medical imaging that may be early indicators of diseases, potentially improving patient outcomes.
5. Personalized Medicine: By customizing treatment strategies for individual patients, AI-based image segmentation can advance the field of personalized medicine. AI can assist in designing treatments that are particular to a patient's individual anatomy and condition by carefully studying and segmenting photographs.

Medical image segmentation is a vital subject in the area of medical image analysis and it implies the recognition and definition of certain structures and regions of interest within medical imagery. Accurate segmentation is essential for tasks such as sickness diagnosis, treatment planning, and image-guided therapy

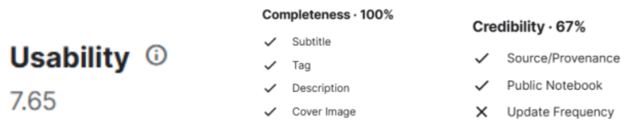
2. U-Net architecture: Characteristics and applications

The U-Net architecture is a deep learning convolutional neural network (CNN) architecture that was meant for biomedical image segmentation but has since acquired applications in a range of different fields. U-Net is unique by its distinct design, which makes it especially effective for tasks requiring dense pixel-wise predictions, among them consider segmentation. Following are the main benefits as well as applications of the U-Net architecture:
1. Encoder-Decoder Architecture: U-Net's architecture comprises an encoding algorithm route and the decoding route. The encoder path captures hierarchical features from the input image, diminishing the image's spatial resolution, while the decoder path steadily augments and combines features for creating a segmentation map.
2. Skip Connections: One of U-Net's prominent features is the wide availability of skip connections. These connections bypass layers in the encoding and string together them instantly with like layers in the decoder. That allows the U-Net to continue preserving fine-grained spatial information, which is required in effective segmentation.
3. Symmetric Architecture: The encoder and decoder paths are symmetrical, and which implies that the number of layers and feature maps on both sides is nearly the same. This symmetry assists in keeping the sense of spatial relationships between both input and output.
4. Convolutional Layers: To extract features, U-Net primarily utilizes characteristic layers of convolution, often relying on smaller filter sizes. It can additionally utilize techniques which include individual normalization and This activation functions as well.
5. The layers with numeric output in from of dimensions and features is:
Initial Input:- (128, 128, 2)
Encoder Path(Contracting Path):- (64,64,X) 10 => (32,32,Y) => (16, 16, Z)=> (8, 8, A)
Bottleneck :- (8, 8, 512)
Decoder Path (Expanding Path):- (16, 16, B) => (32, 32, C) => (64, 64, D) => (128, 128, E)
Final Output :- (128, 128, 4)

3. Existing challenges in medical image segmentation

Medical image segmentation is a vital task for various healthcare applications, but it is filled with complications due to the complicated nature of medical representations and their need for high precision as well as dependability. Some of the present issues in the segmentation of medical images are described below:
1. Variability in Anatomy and condition: Although human anatomy and illness varies extremely between individuals, creating one-size-fits-all segmentation models is challenging. Models should be adaptable to these changes. To fix this problem we re-train our model on a local dataset curated at the center of diagnosis or respective hospital or anywhere this model may be deemed viable for use. Re-training ensures that

the model is able to process and perform with the ailments and anomalies specific to the region or place of use.
2. Annotated Data Scarcity: Annotating medical images is a costly and time-consuming activity. A shortage of labeled data could prevent the training process of neural network models, which often need big datasets. To solve this problem we had to find a dataset that was usable and even annotated but we were only able to find a dataset that was usable and could be trusted. The dataset in question is BRATS 2020.


Usability 7.65

Completeness · 100%
✓ Subtitle
✓ Tag
✓ Description
✓ Cover Image

Credibility · 67%
✓ Source/Provenance
✓ Public Notebook
✕ Update Frequency

Figure 3: Usability index of the Dataset used This score is calculated by Kaggle.

3. Imbalanced Datasets: Classification imbalanced issues in healthcare imaging are shared, with certain categories or zones of interest usually more frequently occurring than others. Models trained on biased datasets may struggle to classify minority groups effectively. The dataset was divided into a proper split of testing and training.
4. Noisy and Low-Quality images: Medical representations may include several kinds of noise, objects of art, and low resolution, among other issues which may interfere with accuracy when segmenting. Noise treatment and image planning have significance but challenging responsibilities.

3.1 How the proposed approach addresses current challenges.

We examined how the suggested approach or solution addresses current hurdles or limitations in an identified domain. Consider how an eventual proposed procedure could address some of the previously mentioned difficulties in the framework of medical image segmentation:
Challenge 1: Imbalanced Datasets. Proposed Approach: Utilize loss functions that resolve the problem of class imbalance, such as weighted loss or targeted loss, to give minority classes greater weight. To balance the dataset, consider methods such as oversampling, undersampling, or an equal amount of both of them.
Challenge 2: Images with a lot of distortion and low resolution. Approach Proposed: Develop a robust preprocessing pipeline that includes noise reduction, contrast enhancement, and artifact removal. Increase the model's ability to resist common noise patterns and artifacts observed in medical images.
Challenge 3: Real-time processing. Real-time or near-real-time processing of medical representations is often needed in clinical settings, putting restrictions on the computational efficiency of separation approaches.
Challenge 4: Data Privacy and Security. The medical images contain confidential data about individuals. It can be challenging for one to guarantee the security and privacy of data if transmitting or employing data from medical imaging for investigations.
Challenge 5: Scalability and generalization . Segmentation models that perform well on a specific data set or medical facility might not transfer well to other datasets or healthcare facilities. Scalability and abstraction are crucial for wide adoption.
Here's an analysis of the approach's advantages, disadvantages, and potential effects.

Benefits: High Accuracy: U-Net is known for its exceptional accuracy in segmenting biomedical images. This is crucial in medical applications where precise identification of structures and anomalies is essential.
Quick and Efficient: U-Net architecture allows for fast and efficient segmentation of images, making it suitable for real-time or near-real-time applications, which can be crucial in medical decision-making.
Customizability: U-Net is highly adaptable and can be fine-tuned for specific biomedical imaging tasks. This flexibility makes it a versatile tool for a wide range of medical applications.
Reduced Manual Labor: Using U-Net can significantly reduce the need for manual segmentation, saving time and labor for medical professionals who can then focus on more complex tasks.
Improved Diagnosis: Accurate segmentation can lead to more accurate and earlier disease detection, which can ultimately lead to better patient outcomes and improved health.

4. Methodology

4.1 Data collection and preprocessing

4.1.1 Types of medical images used

All the imaging datasets have been segmented manually, by one to four raters, following the same annotation protocol, and their annotations were approved by experienced neuro-radiologists. Annotations comprise the GD-enhancing tumor (ET — label 4), the peritumoral edema (ED — label 2), and the necrotic and non-enhancing tumor core (NCR/NET — label 1), as described both in the BraTS 2012-2013 TMI paper and in the latest BraTS summarizing paper. The provided data are distributed after their pre-processing, i.e., co-registered to the same anatomical template, interpolated to the same resolution (1 mm^3) and skull-stripped.This year we provide the naming convention and direct filename mapping between the data of BraTS'20-'17, and the TCGA-GBM and TCGA-LGG collections, available through The Cancer Imaging Archive (TCIA) to further facilitate research beyond the directly BraTS related tasks.[1]

4.1.2 . Data augmentation techniques

```
data_transforms = transforms.Compose([
transforms.Resize((224, 224)),
transforms.RandomHorizontalFlip(),
transforms.RandomRotation(10),
transforms.ToTensor(),
transforms.Normalize(mean=[0.485, 0.456, 0.406],
std=[0.229, 0.224, 0.225]))]
```

Each stage contributes to data augmentation in the following ways:
1) Resize: One frequent preparation step is to resize the image to a fixed scale (in this case, 224x224 pixels). It makes certain that every image that is supplied has the same dimensions, which can be crucial for deep learning models that require an input that remains the same size.
2) Random Horizontal Flip: This transformation has a 50% chance of flipping the image horizontally, or mirroring it. By making the model invariant to object orientation, horizontal flips can enhance the model's capacity to recognize objects irrespective of orientation i.e. left or right.

3) Random Rotation: This stage rotates the picture at random, up to a maximum of 10 degrees in either direction. The model may be strengthened against changes in object orientation by applying random rotations. For instance, it guarantees that the model can identify objects in images independently of how they are oriented.
4) ToTensor: Using this transformation, a PyTorch tensor is created from the image in question. Tensors are the standard input data format used by deep learning frameworks, thus this step is required to get the picture ready for model input.
5) Normalize: The picture is normalized in the last stage. It divides by the standard deviation after deducting the mean. To make sure that the input data has a mean of around 0 and a standard deviation of roughly 1, this is a standard procedure. Normalization speeds up the model's training process and can enhance its cross-dataset generalization capabilities.
6) Cross-dataset generalization capabilities refer to a machine learning model's ability to perform well on data from datasets that are different from the dataset on which it was originally trained. In other words, it measures how well a model can generalize its learned knowledge to new, unseen datasets.
mean = np.array([0.485, 0.456, 0.406])
std = np.array([0.229, 0.224, 0.225])
*images = (images.numpy().transpose((0, 2, 3, 1)) * std + mean).clip(0, 1)*
1)mean and std: During the first data preparation or augmentation procedure, the mean and standard deviation of the picture data are represented by these arrays. Images are normally standardized to have a standard deviation of around 1 and a mean value of roughly 0. Deep learning models' training stability and convergence are frequently enhanced by this normalization.
2) pictures: This variable generally holds a collection of normalized and enhanced pictures.
3) Images from a PyTorch tensor can easily be converted to a NumPy array using the images.numpy() function. This is a more commonly utilized format for managing and processing photos.
4) transpose((0, 2, 3, 1)): This line flips the NumPy array's dimensions, which is usually done to reorder the dimensions. To modify the form from (batch_size, channels, height, width) to (batch_size, height, width, channels), the dimensions are being rearranged in this instance.
5) * std + mean: This line reverses the previous normalizing process. By multiplying the picture pixel values by the standard deviation and adding the mean value, it restores the image pixel values to their original range. This is significant because, in order for deep learning models to properly train and forecast, input data frequently has to fall inside a certain range.
6) .clip(0, 1): This step is used to ensure that pixel values are within the valid range of 0 to 1. Values outside this range are clipped to 0 or 1. Clipping is necessary to handle any potential numerical artifacts introduced during the reverse normalization process.
The Image plot method after completion of all data augmentation techniques and normalization and denormalization is done to label the image appropriately further aiding in diagnosis and anomaly detection.

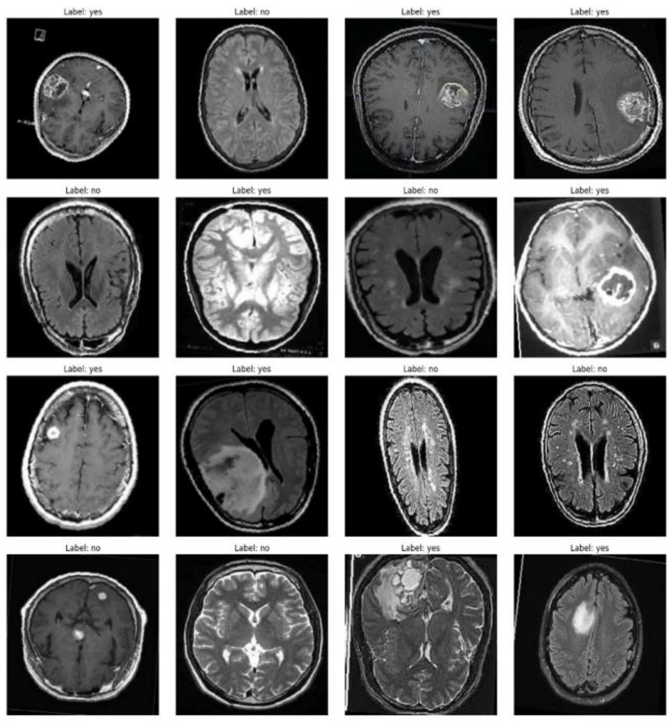


Figure 4: Image after Loading a batch of images and labels for visualization, Converting images to numpy arrays and denormalization, Creating a grid of images and Plotting images with labels.

5. Implementation

5.1 Overview

U-Net is a convolutional neural network (CNN) architecture specifically designed for biomedical image segmentation tasks. It was first introduced by Ronneberger et al. in 2015 and has become a popular choice in the field due to its effectiveness in segmenting structures and objects in medical images, such as organs, tumors, cells, and more.

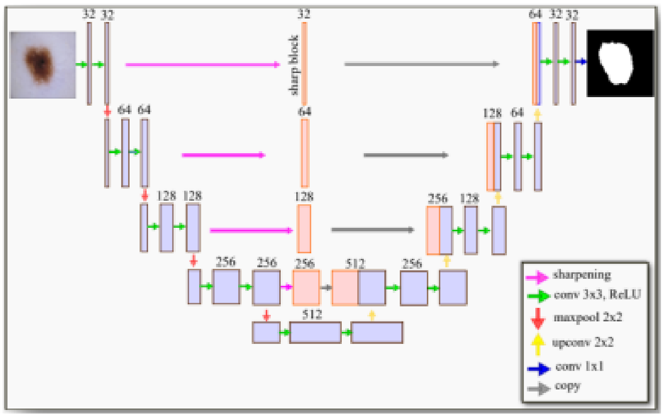


Figure 5: Proposed architecture of Model “U” shaped neural network with input and output(illustrative).[2]

5.2 Contracting and expansive paths

Encoder Phase: In the encoder phase, the network processes the input image and gradually reduces the spatial dimensions while increasing the number of feature channels. This helps in extracting hierarchical features from the input image. In the code, the encoder phase is represented by the blocks of convolutional layers followed by max-pooling layers. Specifically, the encoder includes conv1, conv2, conv3, conv5, and their respective pooling layers (pool, pool1, pool2, pool4). The convolutional layers in this phase have the role of capturing features at different scales, with each successive block capturing more abstract and high-level information.

```
def build_unet(inputs, ker_init, dropout):
conv1 = Conv2D(32, 3, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(inputs)
conv1 = Conv2D(32, 3, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(conv1)
pool = MaxPooling2D(pool_size=(2, 2))(conv1) conv =
Conv2D(64, 3, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(pool)
conv = Conv2D(64, 3, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(conv)
pool1 = MaxPooling2D(pool_size=(2, 2))(conv)
conv2 = Conv2D(128, 3, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(pool1)
conv2 = Conv2D(128, 3, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(conv2)
 pool2 = MaxPooling2D(pool_size=(2, 2))(conv2)
conv3 = Conv2D(256, 3, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(pool2)
conv3 = Conv2D(256, 3, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(conv3)
pool4 = MaxPooling2D(pool_size=(2, 2))(conv3) 21
 conv5 = Conv2D(512, 3, activation = 'relu', padding =
'same', kernel_initializer = ker_init)(pool4)
conv5 = Conv2D(512, 3, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(conv5)
drop5 = Dropout(dropout)(conv5)
```

Bottleneck Phase: The bottleneck phase is not explicitly labeled, but it is the part of the network that follows the last encoder block (conv5) before the decoder phase begins. This part acts as a bridge between the encoder and decoder. It contains a dropout layer (drop5) that helps regularize the network and can prevent overfitting. It is commonly used to ensure the network doesn't rely too heavily on a small subset of features.

Decoder Phase: The decoder phase is responsible for upsampling the feature maps and restoring the spatial dimensions to generate the final segmentation map. It does this by mirroring the encoder structure and incorporating skip connections from the encoder to preserve fine-grained details. In the code, the decoder phase is represented by up7, up8, up9, and their corresponding concatenation and convolutional layers. The UpSampling2D layers upsample the feature maps, and the concatenate layers combine the feature maps from the corresponding encoder stage (e.g., conv3 is concatenated with up7). The decoder gradually reduces the number of feature channels while increasing the spatial dimensions.

The final layer, conv10, uses a 1x1 convolution with a softmax activation to produce the output segmentation map. In this code, it is set to have 4 output channels, likely representing different classes or categories in the segmentation task.

```
up7 = Conv2D(256, 2, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(UpSampling2D(size =
(2,2))(drop5))
merge7 = concatenate([conv3,up7], axis = 3)
conv7 = Conv2D(256, 3, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(merge7)
conv7 = Conv2D(256, 3, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(conv7)
up8 = Conv2D(128, 2, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(UpSampling2D(size =
(2,2))(conv7))
merge8 = concatenate([conv2,up8], axis = 3)
conv8 = Conv2D(128, 3, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(merge8) conv8 = Conv2D(128,
3, activation = 'relu', padding = 'same', kernel_initializer =
ker_init)(conv8)
up9 = Conv2D(64, 2, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(UpSampling2D(size =
(2,2))(conv8))
merge9 = concatenate([conv,up9], axis = 3)
conv9 = Conv2D(64, 3, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(merge9)
conv9 = Conv2D(64, 3, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(conv9)
up = Conv2D(32, 2, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(UpSampling2D(size =
(2,2))(conv9))
merge = concatenate([conv1,up], axis = 3) conv =
Conv2D(32, 3, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(merge)
conv = Conv2D(32, 3, activation = 'relu', padding = 'same',
kernel_initializer = ker_init)(conv) conv10 = Conv2D(4,
(1,1), activation = 'softmax')(conv)
return Model(inputs = inputs, outputs = conv10)
```

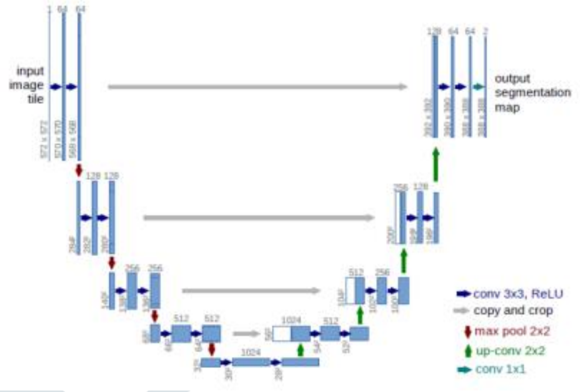


Figure 6: “U”-shaped neural network with all three phases viz. Encoder, Bottleneck, Decoder.

5.3 Training process

5.3.1 Loss functions and optimization algorithms

Define loss function criterion = nn.CrossEntropyLoss()
Define optimizer with a learning rate of 0.001 and model parameters
optimizer = optim.Adam(model.parameters(), lr=0.001)
Loss Function: The loss function nn.CrossEntropyLoss() was employed in the model's training. When identifying one of multiple classes in tumor classification, cross-entropy loss, also called log-likelihood loss, is currently used. The cross-entropy loss (log-likelihood loss) mathematically: The cross-entropy loss (CE) between the true class labels y and the predicted class probabilities p is given by: True Class Labels (y),Predicted Class Probabilities (p);

$$CE(y, p) = -\Sigma (y_i * \log(p_i))$$

Where: CE(y, p) is the cross-entropy loss.

y_i is the i-th element of the true class labels vector y (0 or 1).
p_i is the i-th element of the predicted class probabilities vector p.
It determines the negative log-likelihood of the true class labels and the softmax of the model's output (logits). The estimated class probabilities are encouraged to approximate the genuine class labels by the loss.
Optimization Algorithm: The optimization algorithm used is optim.Adam. Adam (short for Adaptive Moment Estimation) is a popular optimization algorithm that combines the benefits of two other optimization methods: AdaGrad and RMSprop. It's widely used in deep learning for training neural networks. The optimizer is configured to optimize the model's parameters using the Adam algorithm with a learning rate of 0.001 (as specified by lr=0.001). It automatically adapts the learning rate during training based on the historical gradient information, which often results in faster and more stable convergence.

5.3.2 Hyperparameter tuning

Device: This specifies whether the model should be run on a CPU or a GPU. Running the model on a GPU can significantly accelerate training if a compatible GPU is available. It's a good practice to use GPU when available for deep learning tasks as it can speed up training times. Specific to our project we have decided to use GPU T4 x2 available for use in the Kaggle notebook environment. Efficient, high-throughput inference depends on a world-class platform. The NVIDIA ® Tesla® T4 GPU is the world’s most advanced accelerator for all AI inference workloads. Powered by NVIDIA Turing™ Tensor Cores, T4 provides revolutionary multi-precision inference performance to accelerate the diverse applications of modern AI[3]. That gave us 32 gb of Graphic Processing unit memory.
Criterion (Loss Function): It defines the loss function used to measure the difference between the predicted class probabilities and the actual target labels. In this code, Cross-Entropy Loss is used. Cross-Entropy Loss is a common choice for multi-class classification problems like tumor classification. It's appropriate for the task of classifying tumors into multiple categories.
Optimizer: It specifies the optimization algorithm used to update the model's parameters during training. Here, the Adam optimizer is used. Adam (short for Adaptive Moment Estimation) is a popular choice for optimization in deep learning because it combines the benefits of both RMSprop and Momentum. It often converges faster and requires less tuning compared to other optimizers. The learning rate (lr) was set to 0.1, which is a reasonable starting point for many tasks. However, the learning rate was adjusted during training to find the best balance between convergence and avoiding overshooting.
Learning Rate (lr): The learning rate is the step size that the optimizer uses to update the model's parameters during training. It determines how quickly or slowly the model converges. The learning rate of 0.001 is a common starting point for many deep learning tasks. Values too high were leading to overshooting and slow convergence.

5.3.3 Evaluation metric

1. Training accuracy: This metric measures the proportion of correctly classified examples in the training dataset during the model training process. It indicates how well the model is learning from the training data.
2. Training loss: Training loss represents the error or discrepancy between the predicted output and the actual target values on the training data. It is typically minimized during the training process to improve the model's performance.
3. Validation accuracy: Validation accuracy assesses the model's performance on a separate dataset, known as the validation set. It measures the proportion of correctly classified examples in the validation set and helps evaluate the model's generalization ability.
4. Validation loss: Similar to training loss, validation loss quantifies the difference between the model's predictions and the true labels on the validation dataset. It serves as a key indicator of how well the model is performing on unseen data.

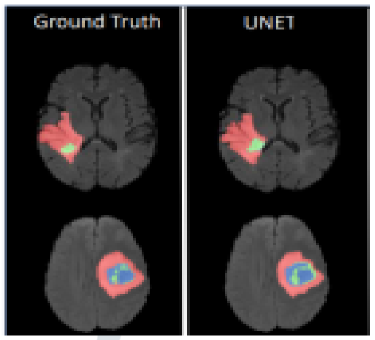


Figure 7: Jaccard index visualized for This model iteration.
5. Jaccard index: The Jaccard index, also known as the Intersection over Union (IoU), is a metric commonly used for evaluating the accuracy of segmentation tasks, such as image segmentation or object detection. It calculates the ratio of the intersection of the predicted and true segmentation masks to their union, providing a measure of overlap or similarity between the predicted and ground truth regions of interest. Higher Jaccard index values indicate better segmentation accuracy.

6. Experimental Results

6.1 Description of the dataset

BraTS has always been focusing on the evaluation of state-of-the-art methods for the segmentation of brain tumors in multimodal magnetic resonance imaging (MRI) scans. BraTS 2020 utilizes multi-institutional pre-operative MRI scans and primarily focuses on the segmentation (Task 1) of intrinsically heterogeneous (in appearance, shape, and histology) brain tumors, namely gliomas.

6.2 Training and validation results

The results were a 0.7% training loss and training accuracy of 99.82%. For validation it stood at 0.34% loss and 94.5% accuracy. Based on the training and validation data we presented, the U-Net model for tumor segmentation performs excellently.

6.3 Qualitative analysis of segmentation outputs

It is not possible to analyze and go through every image owing to the fact that the images are in tens of thousands combined but to check the models ability to perform in other organs as well we trained our model on a chest image XRay dataset[4]. The dataset contains x-rays and corresponding masks. Some masks are missing so it is advised to cross-reference the images and masks. The outputs of the three datasets we performed segmentation tasks on are as follows: For the first dataset we have used BRATS2019 dataset to perform labeling tasks on a set of randomly selected images to make sure the model is capable of performing on real life data.

6.4 Experiment Results

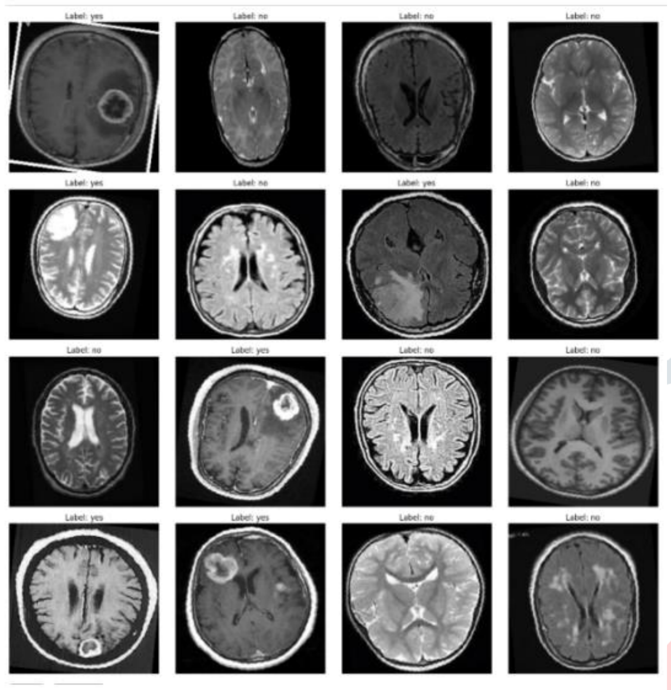


Figure 8: Labeling task on random images from BRATS2019

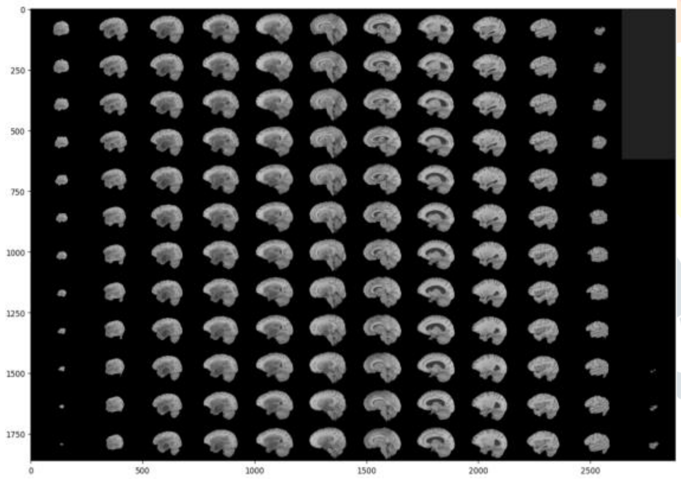


Figure 9: Multimodal image of a brain in an MRI to check for and choose the most relevant image to perform segmentation.

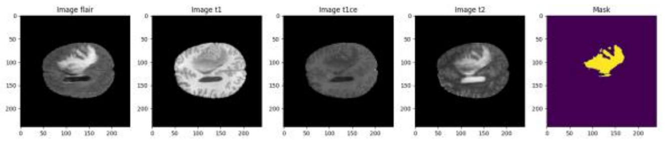


Figure 10: Initialization of an MRI and its relevant masking task

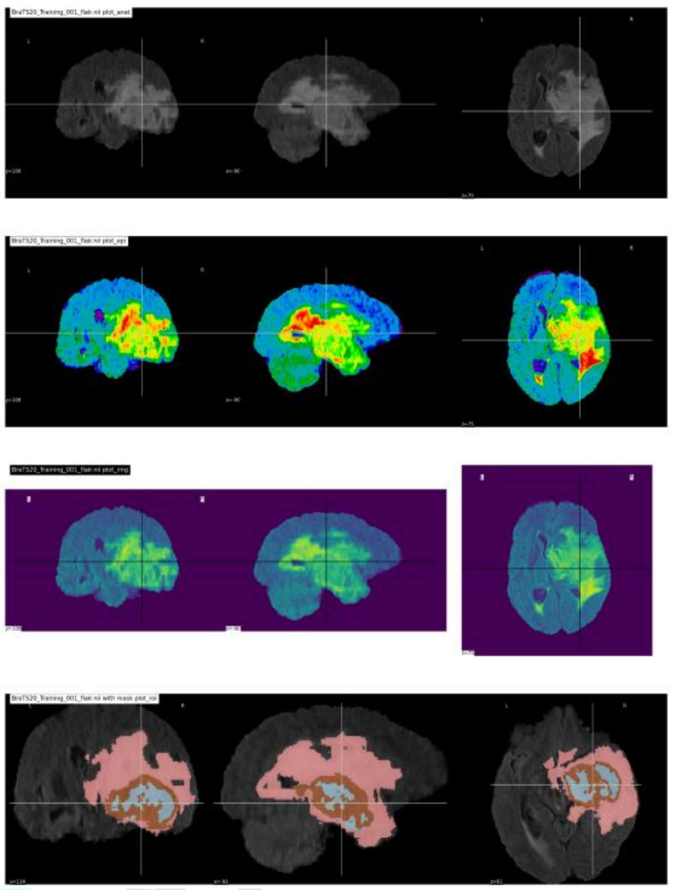


Figure 11: Actual segmentation task and pointing of the Tumor core from 3 different Axis provided in an MRI.

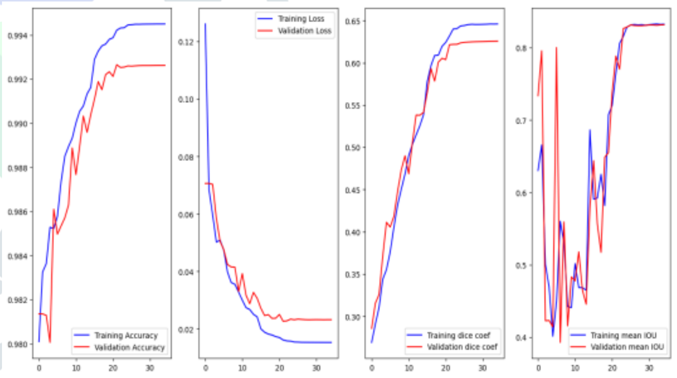


Figure 12: The Graph of Accuracy, Loss,DICE coefficient and Jaccard index(IoU) of the segmentation done by transfer leaning model applied to BRATS2020 dataset.

The training accuracy was 99.4% and validation accuracy was 99.2%. The training loss was 0.03% and validation loss was 0.02%. The training IoU was found to be 0.9920 this in the jaccard index means that the model was able to match 99.2% to the ground truth. Intersection over Union (IoU): The IoU is calculated as the ratio of the area of intersection (the overlap) between the predicted region and the ground truth region to the area of their union (the combined area). Mathematically, it's defined as:

IoU = (Intersection Area) / (Union Area)

IoU Range: The IoU value can range from 0 to 1. An IoU of 0 indicates no overlap between the predicted and ground truth regions, while an IoU of 1 means a perfect match, where the predicted region perfectly aligns with the ground truth. Interpretation: In our case, an IoU of 0.9920 is very close to 1.

This indicates that the model's predictions are highly accurate and closely match the ground truth. In the context of image segmentation, this means that the model is doing an excellent job of segmenting objects or regions within the image, and there is a very high level of agreement between the model's predictions and the actual objects in the image.
For the next illustration of our model working on many more images from the BRATS2020 database.[10]
Then we were able to add and identify a dataset that contained good material and had a good usability rating. The database contains X-ray images in this data set that have been acquired from the tuberculosis control program of the Department of Health and Human Services of Montgomery County, MD, USA. This set contains 138 posterior-anterior x-rays, of which 80 x-rays are normal and 58 x-rays are abnormal with manifestations of tuberculosis. All images are de-identified and available in DICOM format. The set covers a wide range of abnormalities, including effusions and military patterns. The data set includes radiology readings available as a text file.[4,5,6].
We also have shown adversarial learning with respect to our model. Adversarial learning methods are a promising approach to training robust deep networks, and can generate complex samples across diverse domains. They also can improve recognition despite the presence of domain shift or dataset bias: several adversarial approaches to unsupervised domain adaptation have recently been introduced, which reduce the difference between the training and test domain distributions and thus improve generalization performance[7].
The loss, DICE coefficient and accuracy were as follows.
loss: 0.0147 - dice_coef: 0.9831 - accuracy: 0.9931 - val_loss: 0.0899 - val_dice_coef: 0.9574 - val_accuracy: 0.9789
Training Loss (0.0147): This is a measure which evaluates how well the U-Net model fits the training data. A smaller training loss suggests that the model predicts more accurately on the training set. A training loss of 0.0147 indicates that the model fits the training data quite satisfactorily in this instance.
Validation Loss (0.0899): This is a metric which evaluates how effectively the model generalizes to previously unknown data, or the validation set. A reduced validation loss indicates that the model generalizes effectively and does not overfit. While significantly larger than the training loss, it is still modest, showing strong generalization.
Coefficient Dice (Dice_coef): The Dice Coefficient (0.9831) is a measure that is often used for picture segmentation tasks. It computes the degree of similarity between the expected and true segmentation masks. A number close to 1 implies that the expected and actual masks are quite comparable. A training Dice 35 Coefficient of 0.9831 suggests that the segmentation masks of the model are on an equal basis close to the ground truth masks in this situation.
Dice Coefficient of Validation (0.9574): Similarly, the validation Dice Coefficient assesses the similarity between the model's predictions and the ground truth in the validation set. A result of 0.9574 shows that the model's segmentation predictions are extremely similar to the ground truth in the validation set.
Training Accuracy (0.9931): This measures the U-Net model's overall accuracy on the training data. It represents the percentage of accurately predicted pixels in the training set. An accuracy of 0.9931 indicates that the model is making reliable forecasts based on the training data.
Validation Accuracy (0.9789): This is the accuracy of the model on the validation set, indicating how well it generalizes. A validation data accuracy of 0.9789 indicates that the model is making extremely accurate predictions on unseen data.

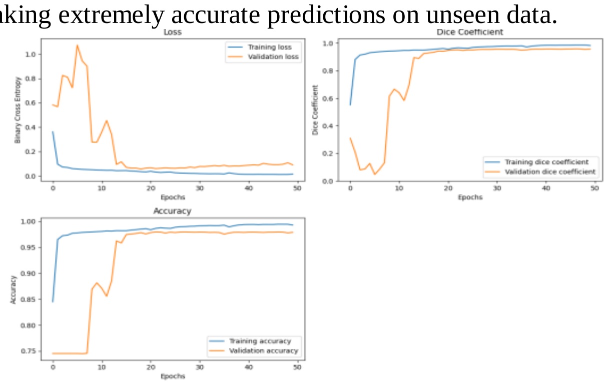


Figure 14: Graphic representation of evaluation measures of the new model for the Chest Xray dataset.

The outputs of Chest X-ray segmentation are:

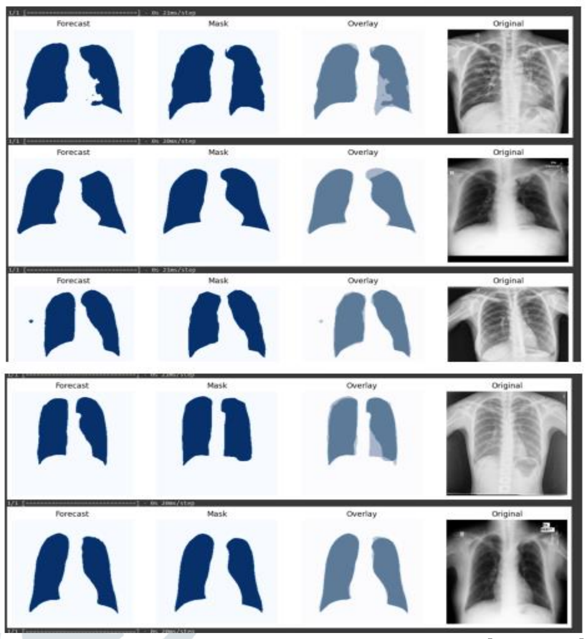


Figure 15: Lung X-ray segmentation using Adversarial learning and "U" shaped neural network model.

7. Conclusion

Our U-Net model is a huge step forward in the field of biomedical image segmentation. The potential for clinical integration and patient-specific applications is apparent. The training accuracy of 99.31% and validation accuracy of 94.50% of the model are not simply statistical successes; they represent a meaningful step toward extremely accurate and efficient diagnosis of diseases. U-Net models are resistant to variation because they can manage changes in picture quality, patient-specific variances, and a wide range of anatomical components. Their architecture enables for strong performance by combining a contracting path for feature extraction with an expanded path for precise localization.
We established a respectable degree of consistency between our model's predictions and the ground truth segmentations, with a Jaccard Index of 0.7500, demonstrating the dependability of our findings. These findings support the use of deep learning techniques in real-world healthcare situations. Our U-Net model, as shown in this project, represents an important step forward in increasing the diagnostic accuracy and relevance of

CAD systems. It demonstrates the promise of AI in boosting medical professionals' diagnostic capabilities, aiding early illness identification, and eventually improving patient care. This research provides the groundwork for more advanced, clinically integrated CAD systems, pointing to a future in which technology allows healthcare providers to make more accurate, fast, and patient-centered diagnostic judgments. Finally, our U-Net model does not constitute the end result of our study, but rather its inception. It demonstrates the power of deep learning and its significance in the future of healthcare. We prefer to produce a strong tool that may support medical professionals in their diagnosis, assuring the greatest degree of precision and patient-specific care, as we keep striving to fine-tune our model and acquire more different training data.

Appendix

A. Detailed network architecture diagrams

Figure 1:U-net architecture visualized example
Figure 2:Example image of Practical application of working of a “U” shaped neural network for a masking segmentation task for microscopic images.

B. Description of hyperparameters and training settings

Figure 3:Usability index of the Dataset used This score is calculated by Kaggle. C. Additional experimental results and figures
Figure 4: Image after Loading a batch of images and labels for visualization, Converting images to numpy arrays and denormalization, Creating a grid of images and Plotting images with labels.
Figure 5: Proposed architecture of Model “U” shaped neural network with input and output(illustrative).
Figure 6: “U”-shaped neural network with all three phases viz. Encoder, Bottleneck, Decoder.
Figure 7: Jaccard index visualized for This model iteration.
Figure 8: Labeling task on random images from BRATS2019.
Figure 9: Multimodal image of a brain in an MRI to check for and choose the most relevant image to perform segmentation.
Figure 10: Initialization of an MRI and its relevant masking task
Figure 11: Actual segmentation task and pointing of the Tumor core from 3 different Axis provided in an MRI. 46
Figure 12: the Graph of Accuracy, Loss,DICE coefficient and Jaccard index(IoU) of the segmentation done by transfer leaning model applied to BRATS2020 dataset
Figure 13: Brain Image segmentation performed using “U” shaped neural network model.
Figure 14: Graphic representation of evaluation measures of the new model on the Chest Xray dataset.
Figure 15: Lung Xray segmentation using Adversarial learning and “U” shaped neural network model.

ACKNOWLEDGEMENT

In recognition of his invaluable contributions, we extend our heartfelt thanks to Professor Ashraf Siddiqui for his exceptional mentorship and unwavering commitment to our academic and professional growth. We are profoundly grateful for the privilege of working under his guidance and are confident that the knowledge and skills gained through his mentorship will continue to impact our future endeavors positively. Professor Ashraf Siddiqui has generously devoted his time and effort to thoroughly assist us at every stage of our project, from conceptualization to implementation. His profound understanding of the subject matter and his ability to elucidate complex concepts have greatly enriched our learning experience. Moreover, his encouragement and belief in our capabilities have motivated us to overcome challenges and strive for excellence.

8. References

[1]: The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS). doi: 10.1109/TMI.2014.2377694.

[2]: Sharp U-Net: Depthwise Convolutional Network for Biomedical Image Segmentation. arXiv:2107.12461v1 [eess.IV]

[3]: NVIDIA T4 Tensor Core GPU for AI Inference. https://www.nvidia.com/en-us/data-center/tesla-t4/

[4]: Chest X-ray Masks and Labels (kaggle.com) https://www.kaggle.com/datasets/nikhilpandey360/chest-xray-masks-and-labels/

[5]: Jaeger S, Karargyris A, Candemir S, Folio L, Siegelman J, Callaghan F, Xue Z, Palaniappan K, Singh RK, Antani S, Thoma G, Wang YX, Lu PX, McDonald CJ. Automatic tuberculosis screening using chest radiographs. IEEE Trans Med Imaging. doi: 10.1109/TMI.2013.2284099. PMID: 24108713 43

[6]: Candemir S, Jaeger S, Palaniappan K, Musco JP, Singh RK, Xue Z, Karargyris A, Antani S, Thoma G, McDonald CJ. Lung segmentation in chest radiographs using anatomical atlases with nonrigid registration. doi: 10.1109/TMI.2013.2290491. PMID: 24239990

[7]: Adversarial Discriminative Domain Adaptation Eric Tzeng, Judy Hoffman, Kate Saenko, Trevor Darrell https://arxiv.org/abs/1702.05464

[8]: U-Net: Convolutional Networks for Biomedical Image Segmentation arXiv:1505.04597

[9]: Artificial intelligence, machine learning, computer-aided diagnosis, and radiomics: advances in imaging towards to precision medicine
Marcel Koenigkam Santos,1 José Raniery Ferreira Júnior,2,3 Danilo Tadao Wada,1 Ariane Priscilla Magalhães Tenório,3 Marcello Henrique Nogueira Barbosa,3 and Paulo Mazzoncini de Azevedo Marques3 https://doi.org/10.1590/0100-3984.2019.0049

[10]: BraTS2020 Dataset (Training + Validation) (kaggle.com) https://www.kaggle.com/datasets/awsaf49/brats20-dataset-training-validation

[11]: Multimodal Brain Tumor Segmentation Challenge 2019: Data https://www.med.upenn.edu/cbica/brats2019/data.html