CTA SCAN BLOOD CLOT DETECTION

Final Presentation

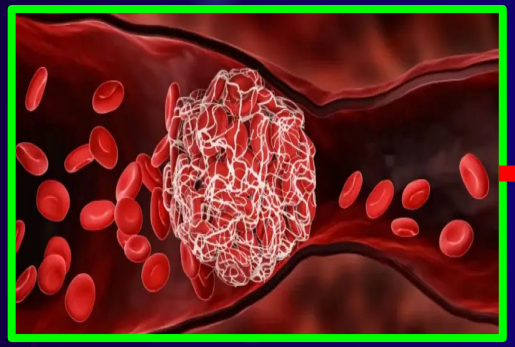
Hshmat Sahak
Nabil Mohamed
Haani Ahmed
Alaap Grandhi
Yawar Ashraf

CTA SCAN BLOOD CLOT DETECTION

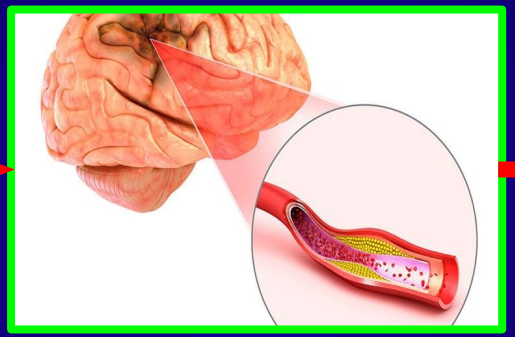
Final Presentation

Hshmat Sahak
Nabil Mohamed
Haani Ahmed
Alaap Grandhi
Yawar Ashraf

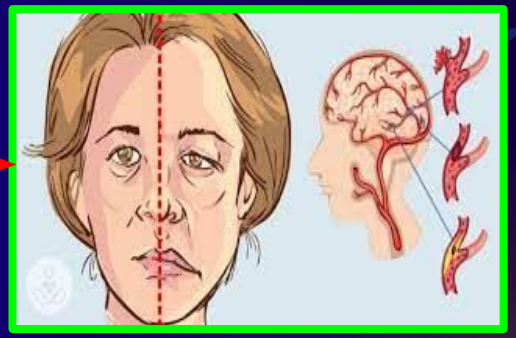
Strokes



Blood Clot blocks Blood Vessel

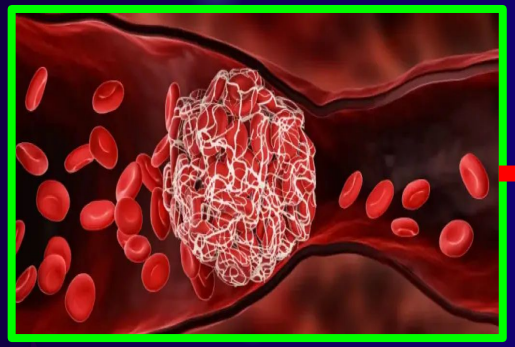


Parts of Brain Damaged

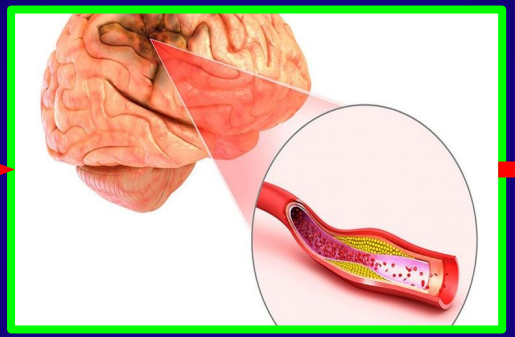


Disabilities like Premature Aging

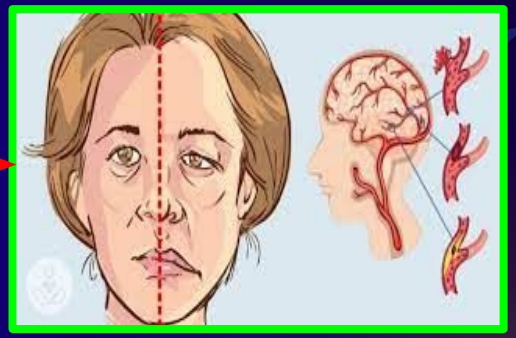
Strokes



Blood Clot blocks Blood Vessel

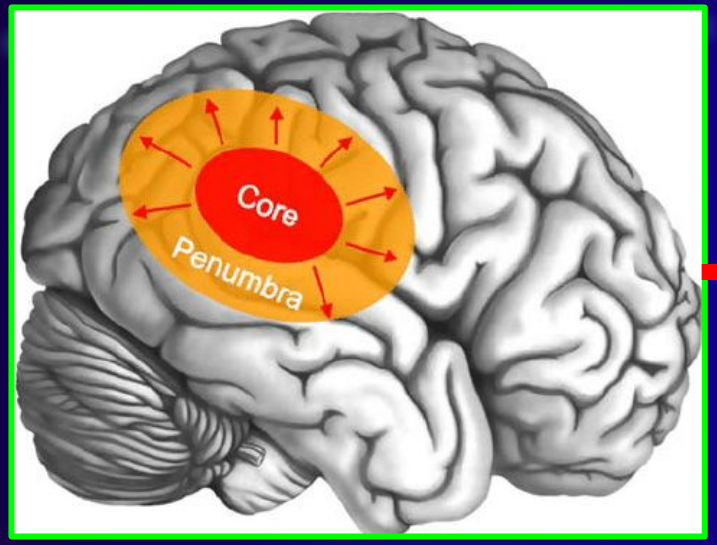


Parts of Brain Damaged

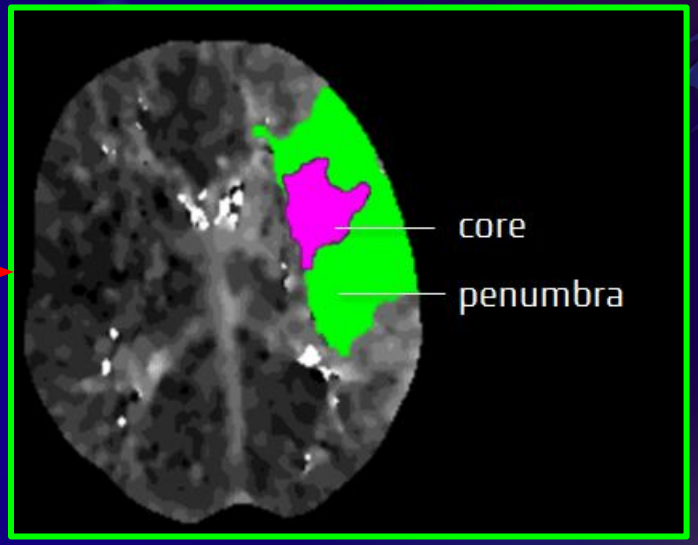


Disabilities like Premature Aging

Importance of Imaging



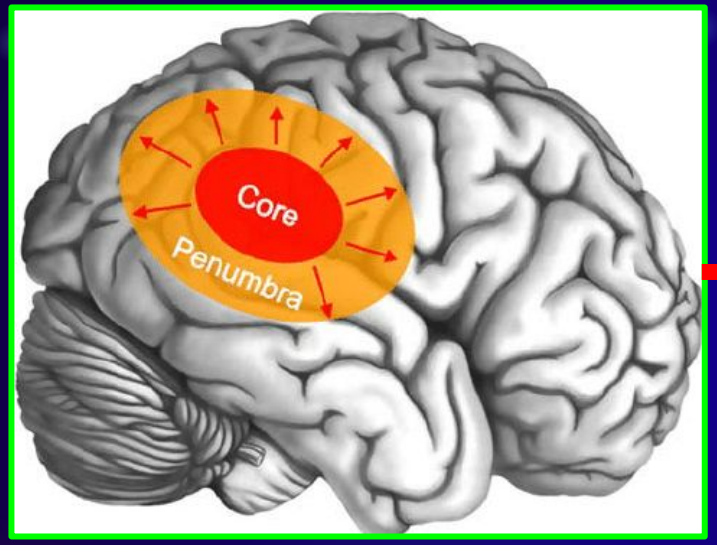
Brain with Blood Clot



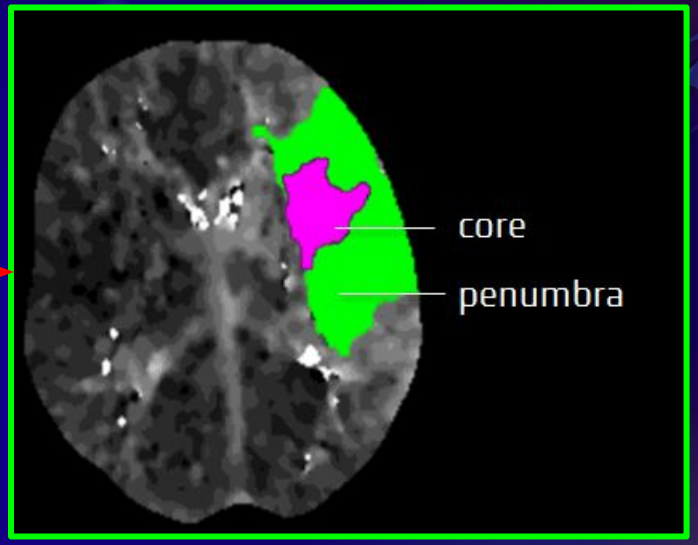
Result of Medical Imaging

- **Penumbral =** tissue affected but not yet dead
- **Core** = tissue that is already dead

Importance of Imaging



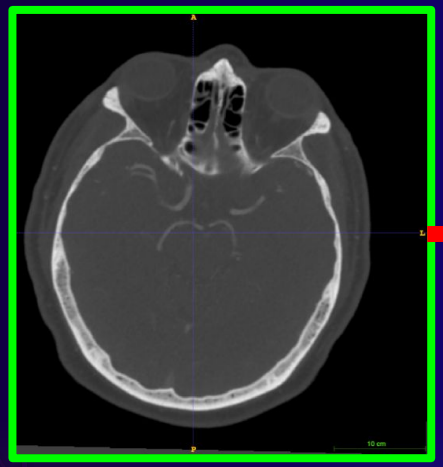
Brain with Blood Clot



Result of Medical Imaging

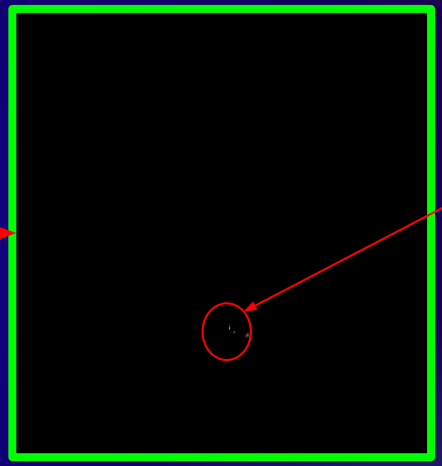
- **Penumbral =** tissue affected but not yet dead
- **Core** = tissue that is already dead

Motivation



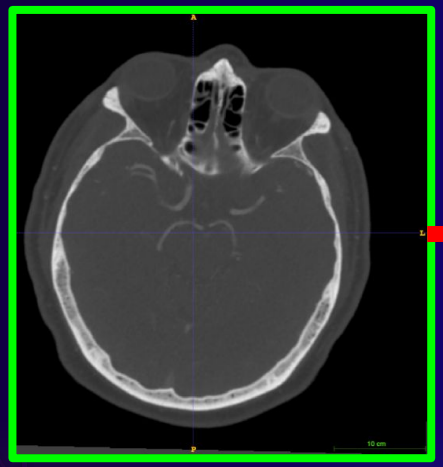
Data from CTA Scan

AUTOMATED method, meant for RURAL areas



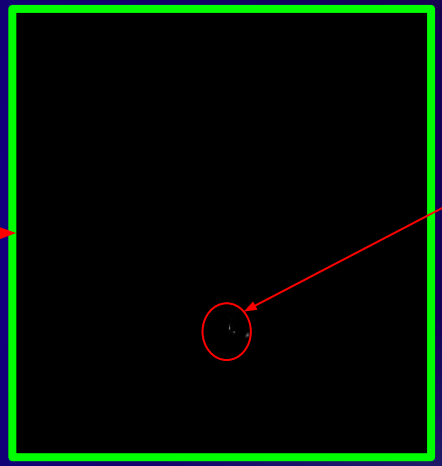
Goal: Identify Blood Clot Region

Motivation



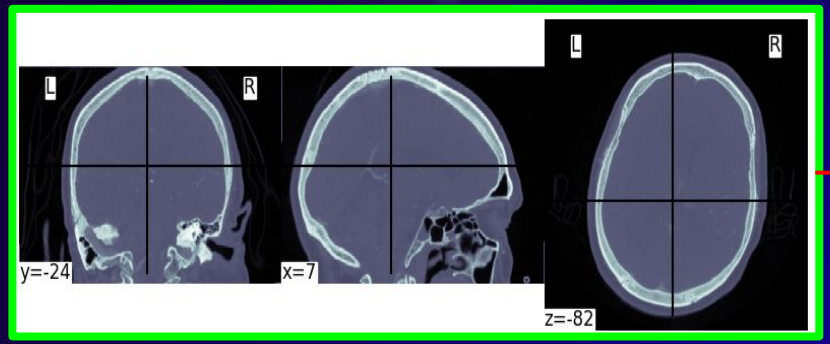
Data from CTA Scan

AUTOMATED method, meant for RURAL areas

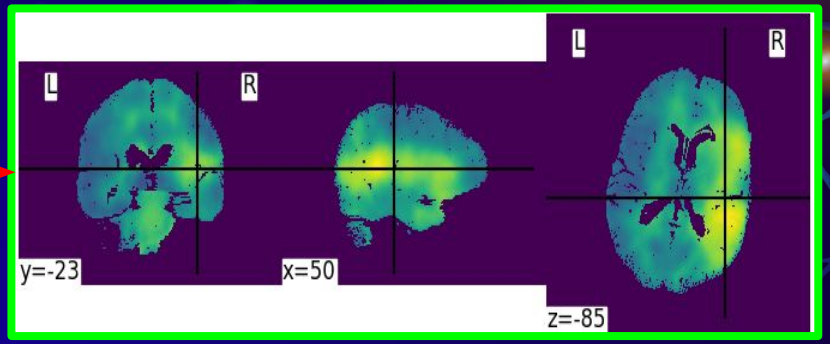


Goal: Identify Blood Clot Region

Problem Statement



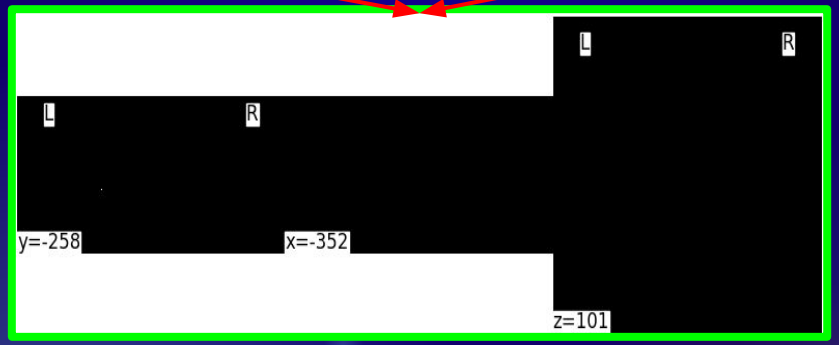
SPIRAL



Raw mCTA Scan

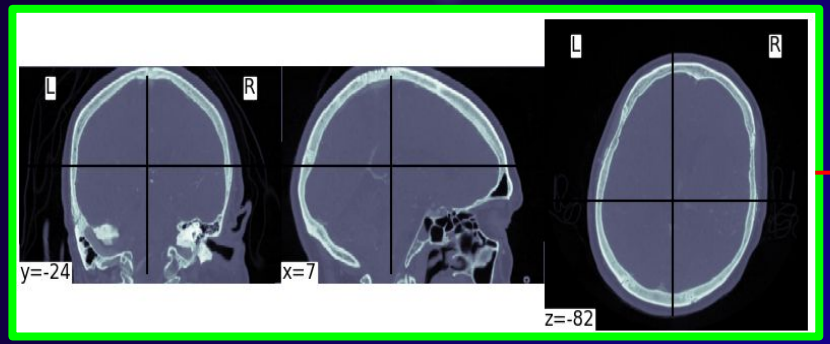
SPIRAL Output

AMI-Net

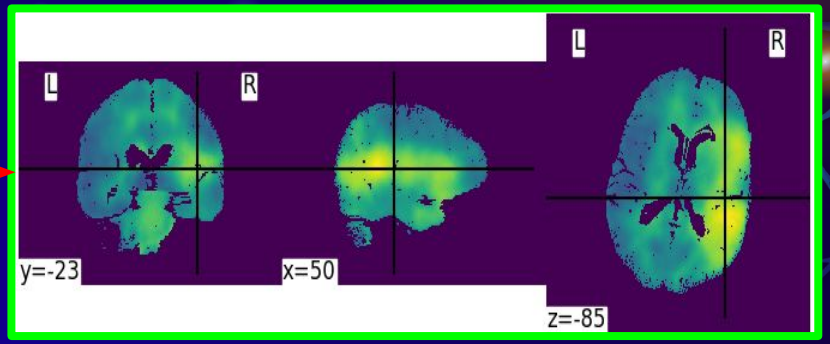


Blood Clot Region

Problem Statement



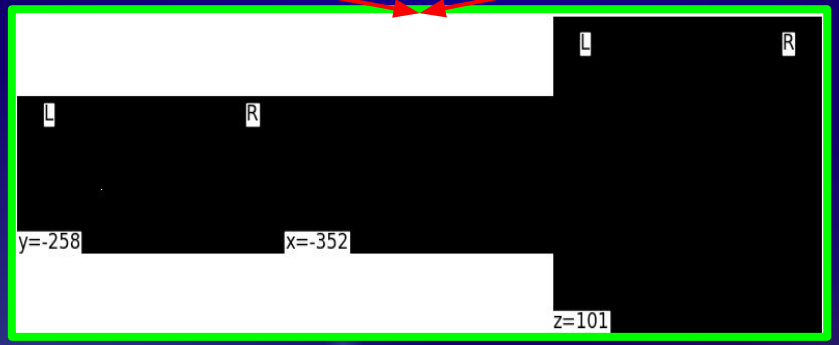
SPIRAL



Raw mCTA Scan

SPIRAL Output

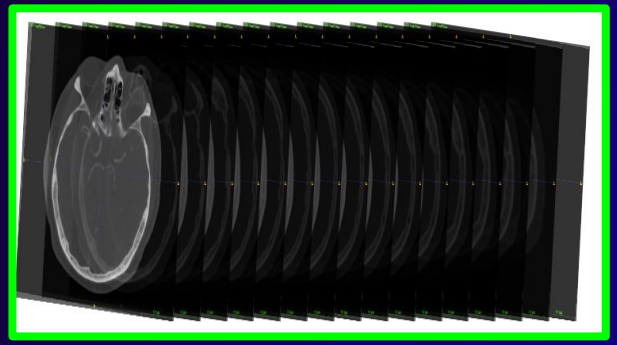
AMI-Net



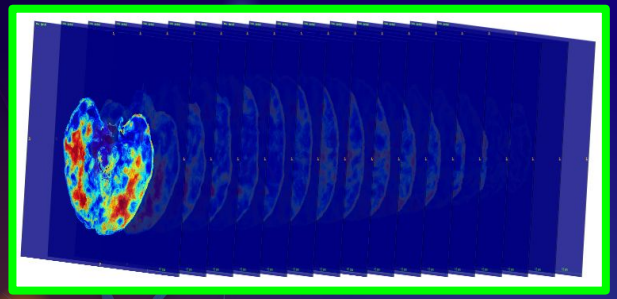
Blood Clot Region

Functional Requirements

Input Format

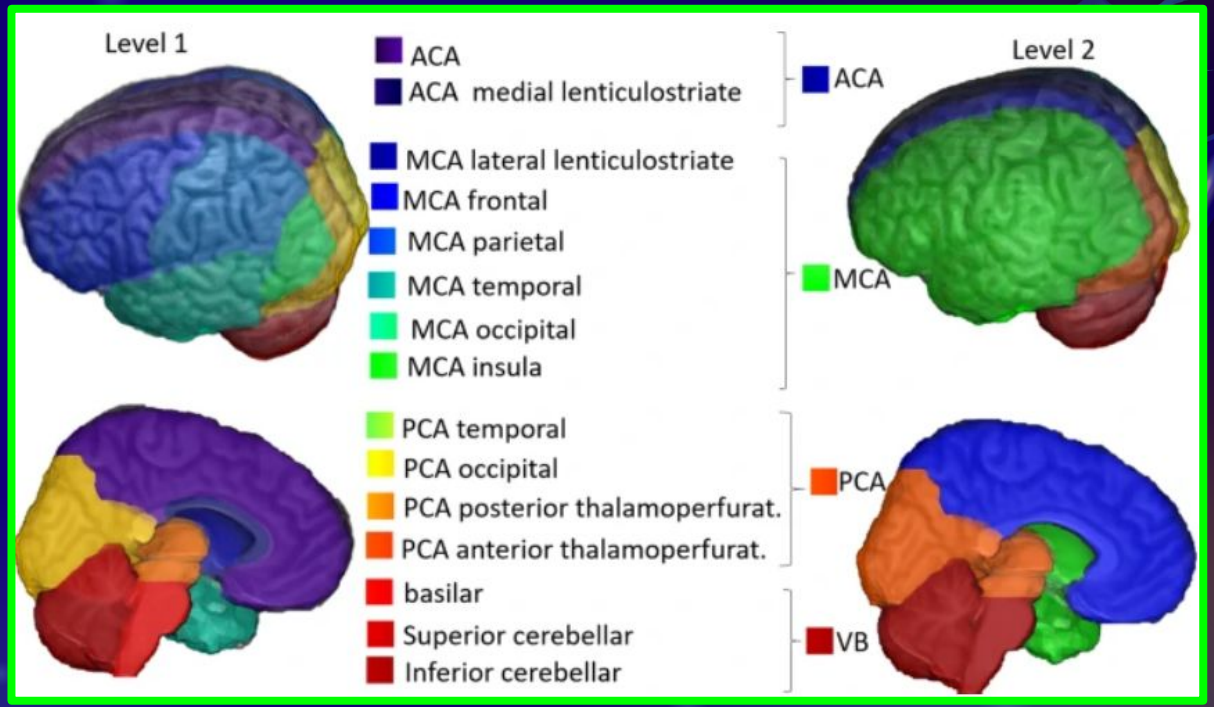


raw mCTA scans



generated SPIRAL images

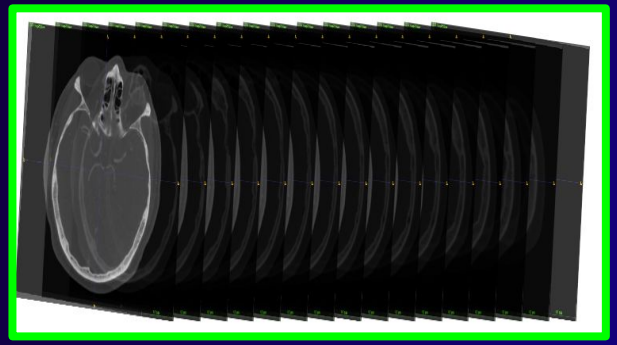
Output Format



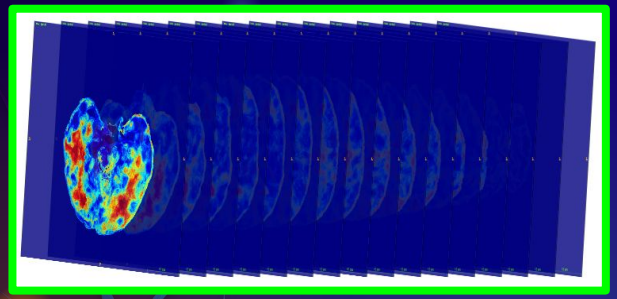
Must determine categorical site(s) of occlusion

Functional Requirements

Input Format

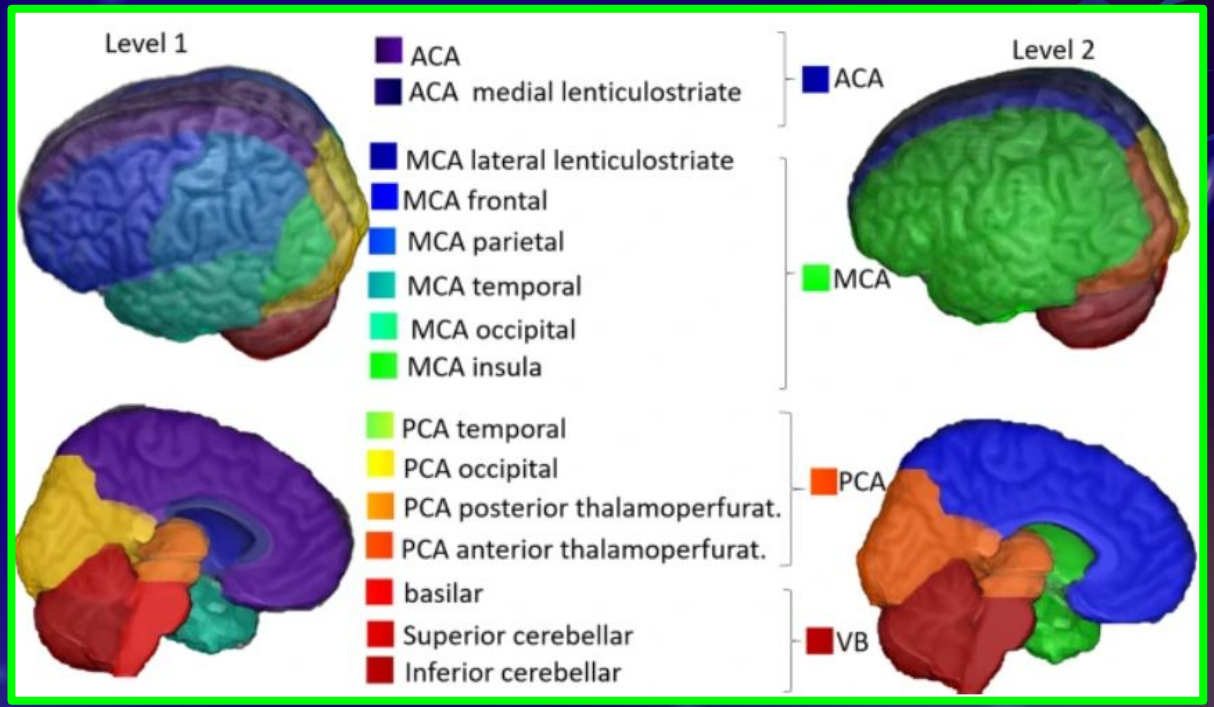


raw mCTA scans



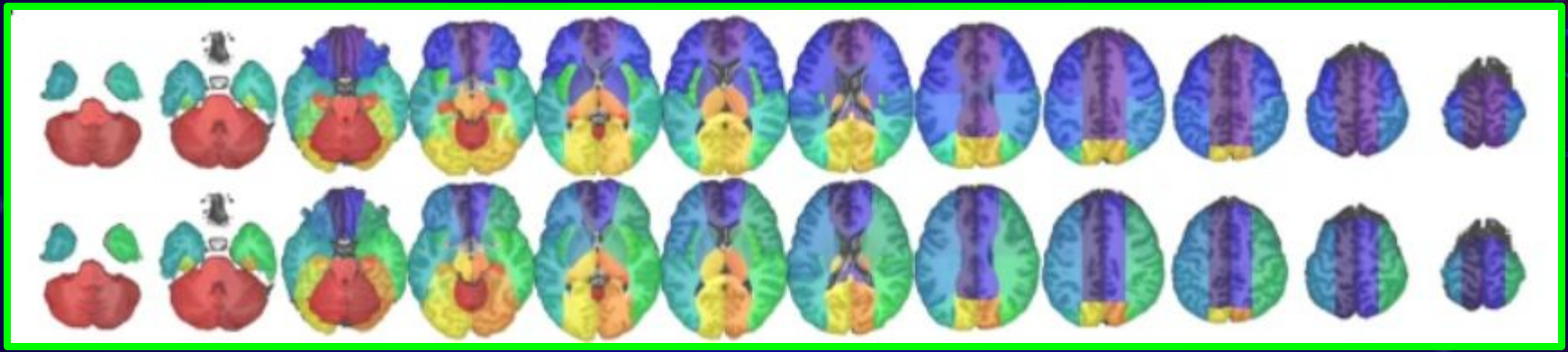
generated SPIRAL images

Output Format



Must determine categorical site(s) of occlusion

Objectives and Constraints



The blood clot will affect one or more of these 15 coloured regions

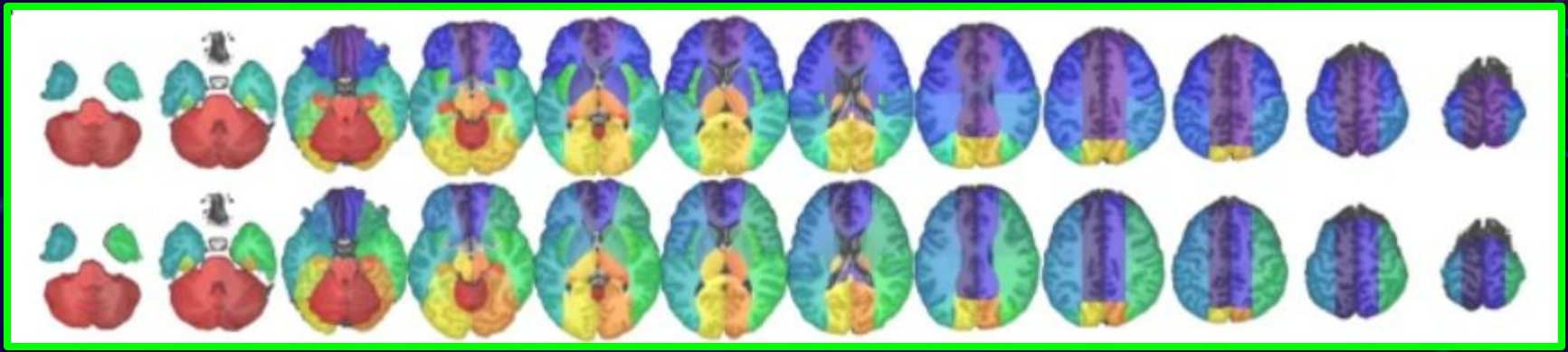
Objectives:

- ***Should outperform existing SPIRAL output in identifying cranial vessels affected by clot***
- ***Should identify the stroke site in a short amount of time***
- ***Should have a low memory footprint***

Constraints:

- ***Must run in under 5 minutes for a single patient***
- ***Must be able to run on a single CPU (8-16 GB)***

Objectives and Constraints



The blood clot will affect one or more of these 15 coloured regions

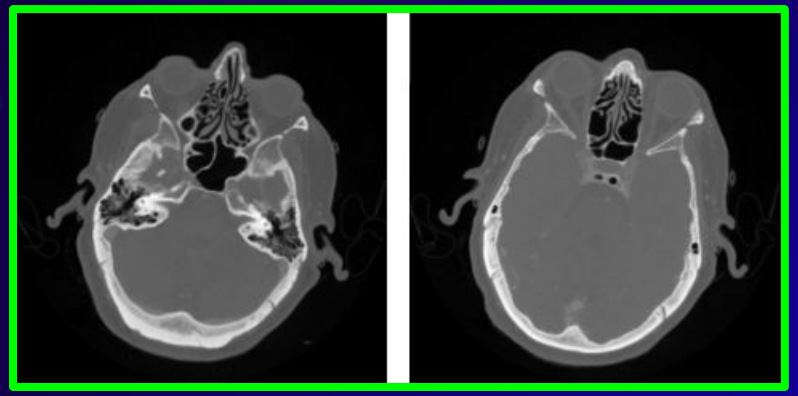
Objectives:

- ***Should outperform existing SPIRAL output in identifying cranial vessels affected by clot***
- ***Should identify the stroke site in a short amount of time***
- ***Should have a low memory footprint***

Constraints:

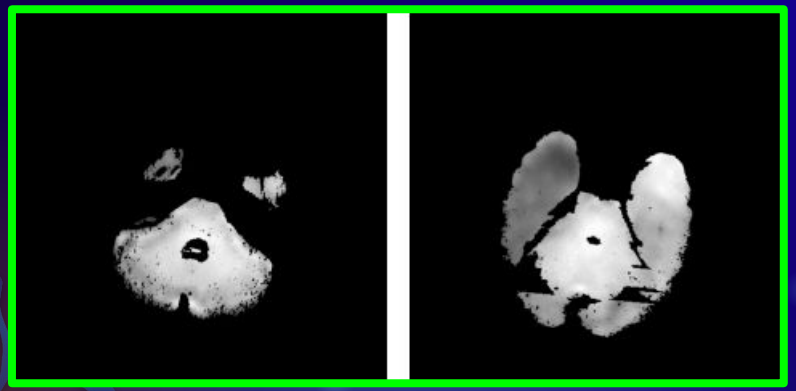
- ***Must run in under 5 minutes for a single patient***
- ***Must be able to run on a single CPU (8-16 GB)***

Data Visualization

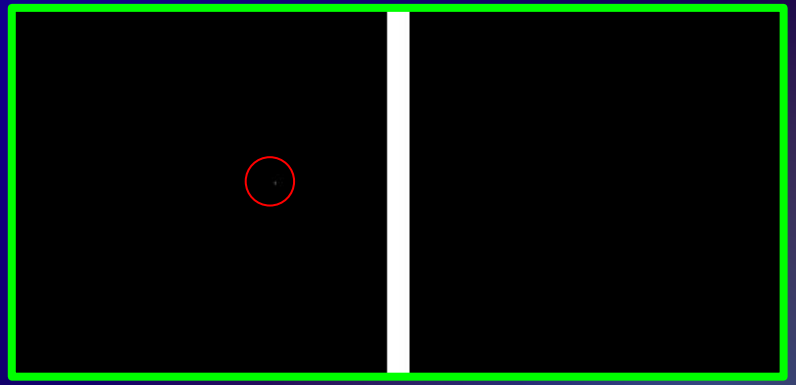


Raw CTA scan samples

ATLAS Map of Brain Regions [0-14]

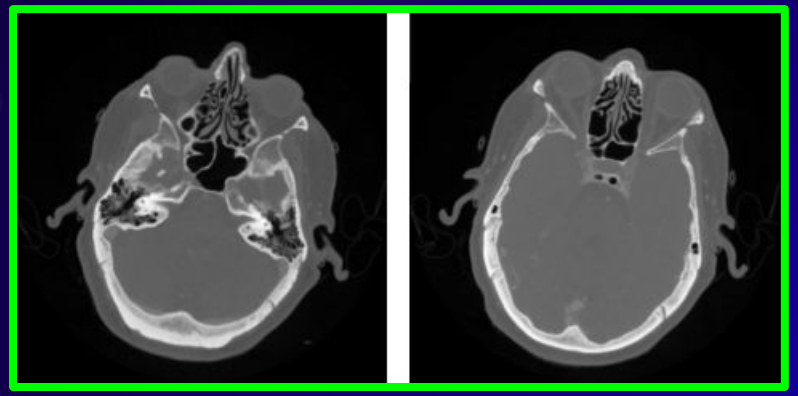


SPIRAL output - each pixel in [0,1]



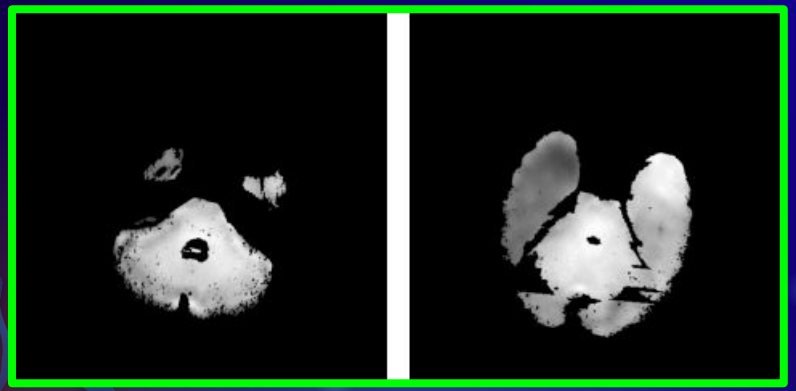
Blood Clot - each pixel in [0,2]

Data Visualization

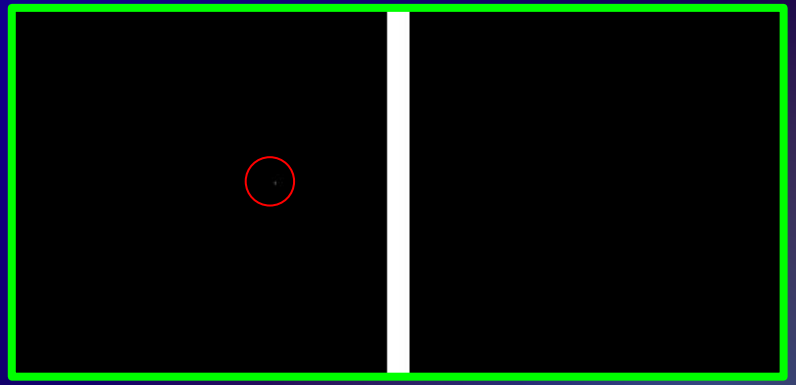


Raw CTA scan samples

ATLAS Map of Brain Regions [0-14]

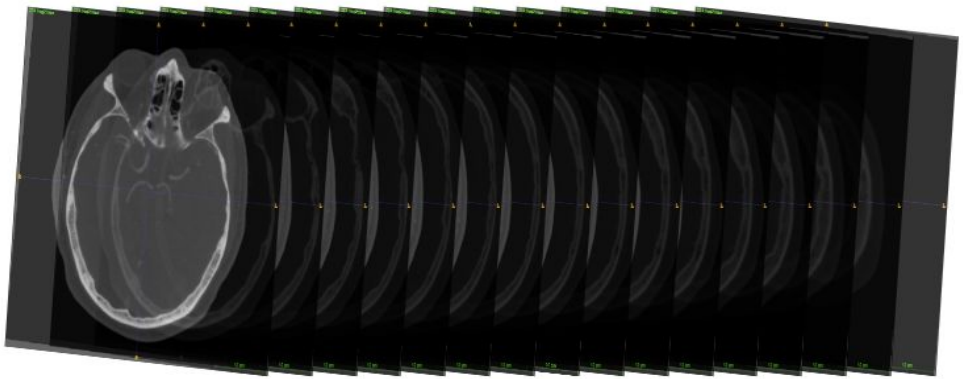


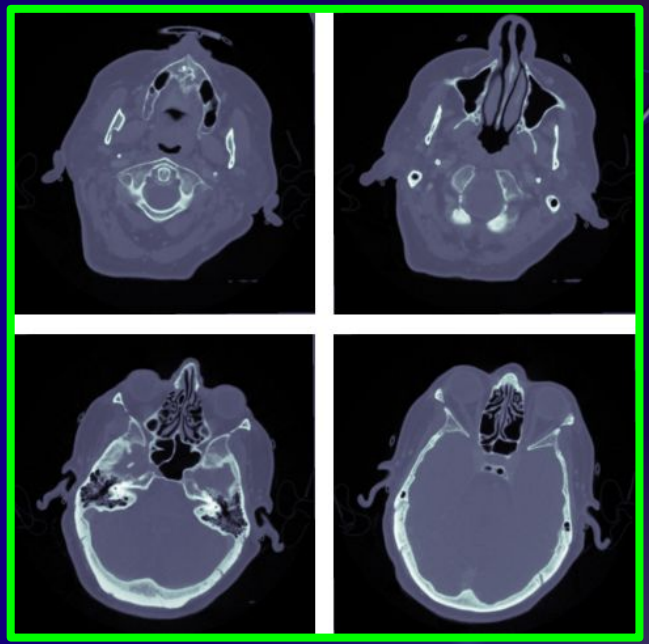
SPIRAL output - each pixel in [0,1]



Blood Clot - each pixel in [0,2]

Data Quantity and Dimensions

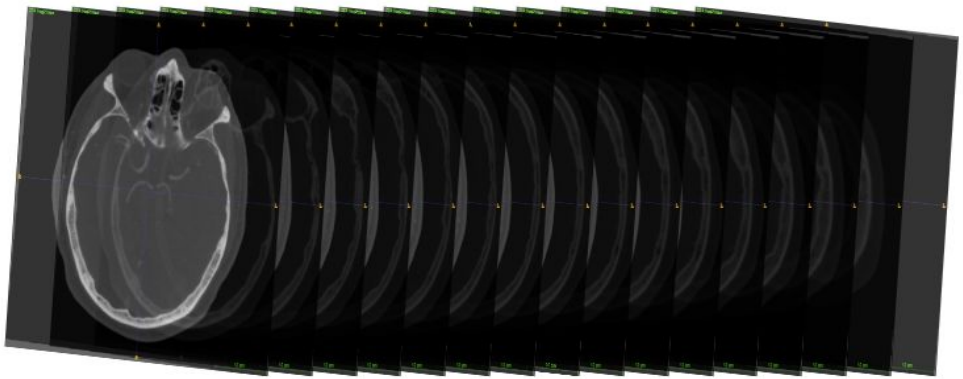


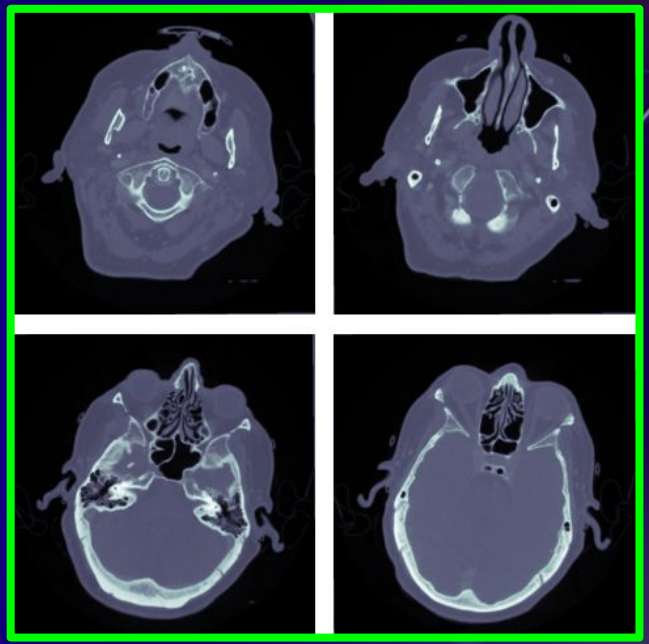


NiBabel
Access a cacophony of neuro-imaging file formats

- CTA scans are stacks of 2d images of brain at different depths
- These are in .NII (nifty) format, are 512 by 512 pixels in dimension, and vary in depth (usually ~200) by patient
- Each file is 50-70 MB

Data Quantity and Dimensions



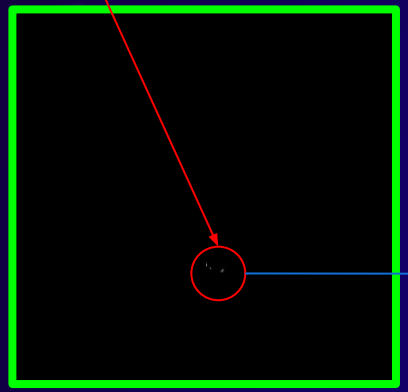


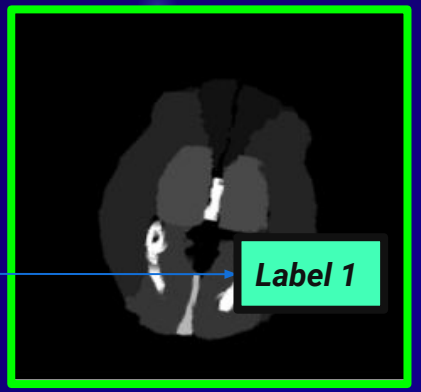
NiBabel
Access a cacophony of neuro-imaging file formats

- CTA scans are stacks of 2d images of brain at different depths
- These are in .NII (nifty) format, are 512 by 512 pixels in dimension, and vary in depth (usually ~200) by patient
- Each file is 50-70 MB

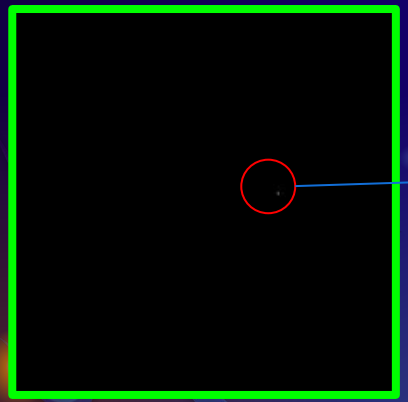
Data Labelling

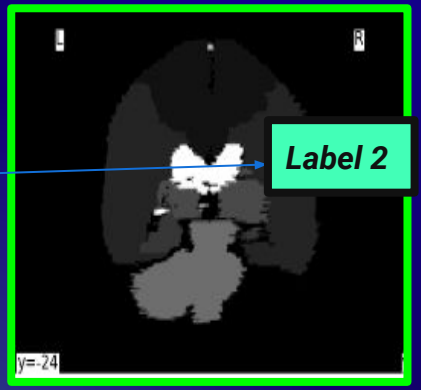
Clot in frame 1





Label 1



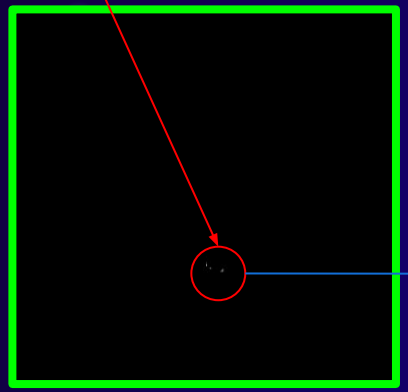


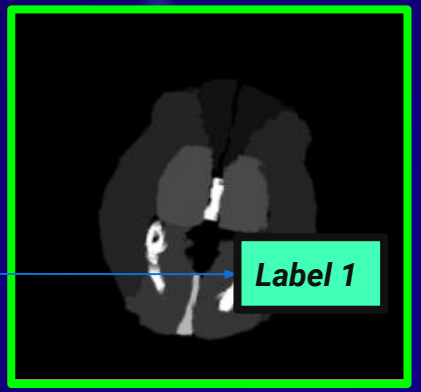
Label 2

Admire the clots

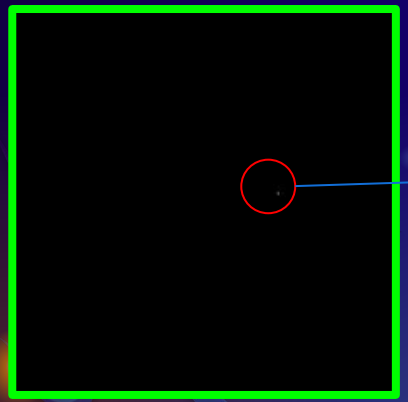
Data Labelling

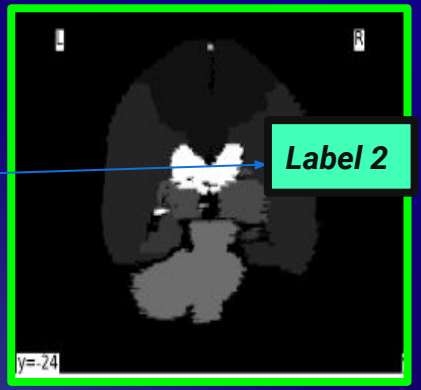
Clot in frame 1





Label 1

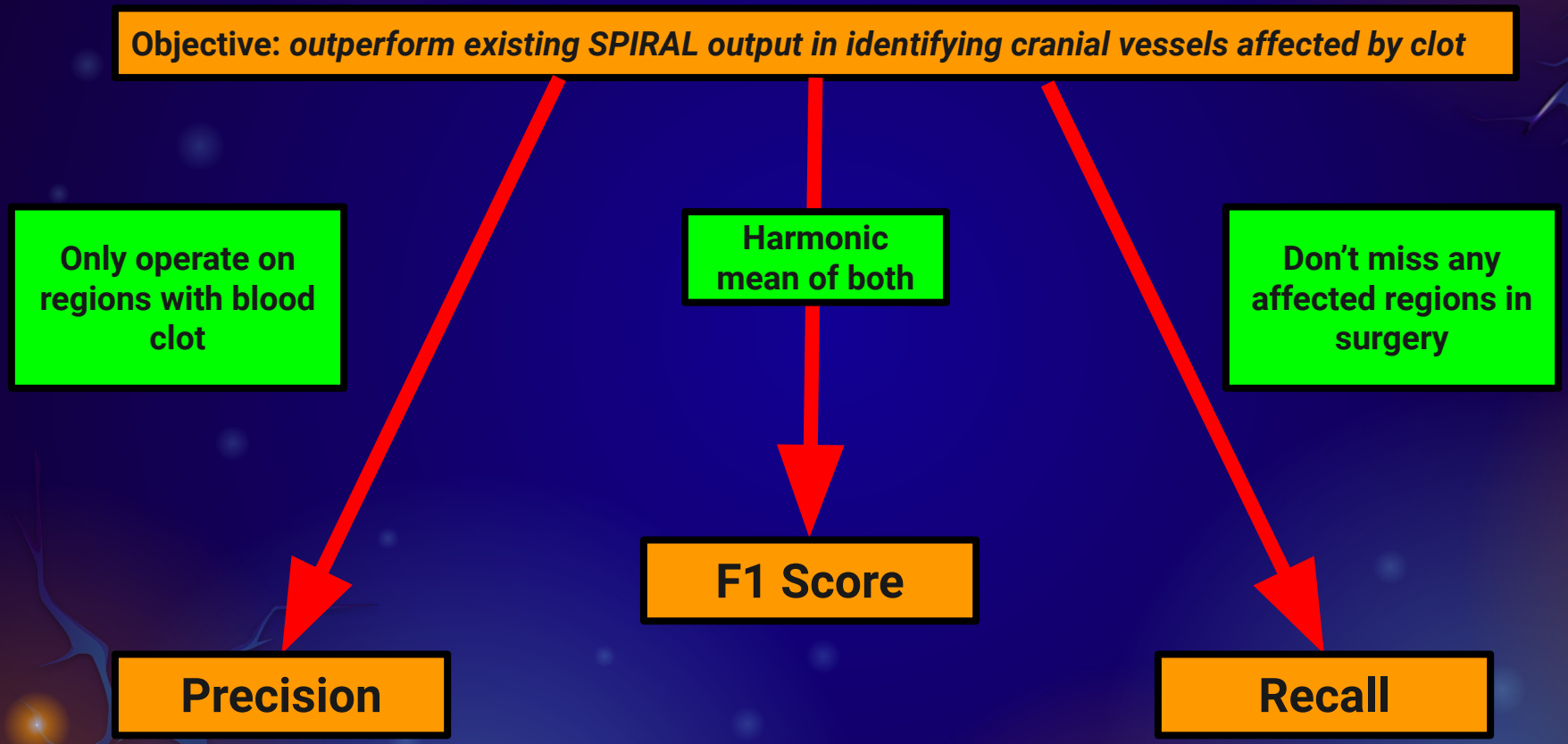




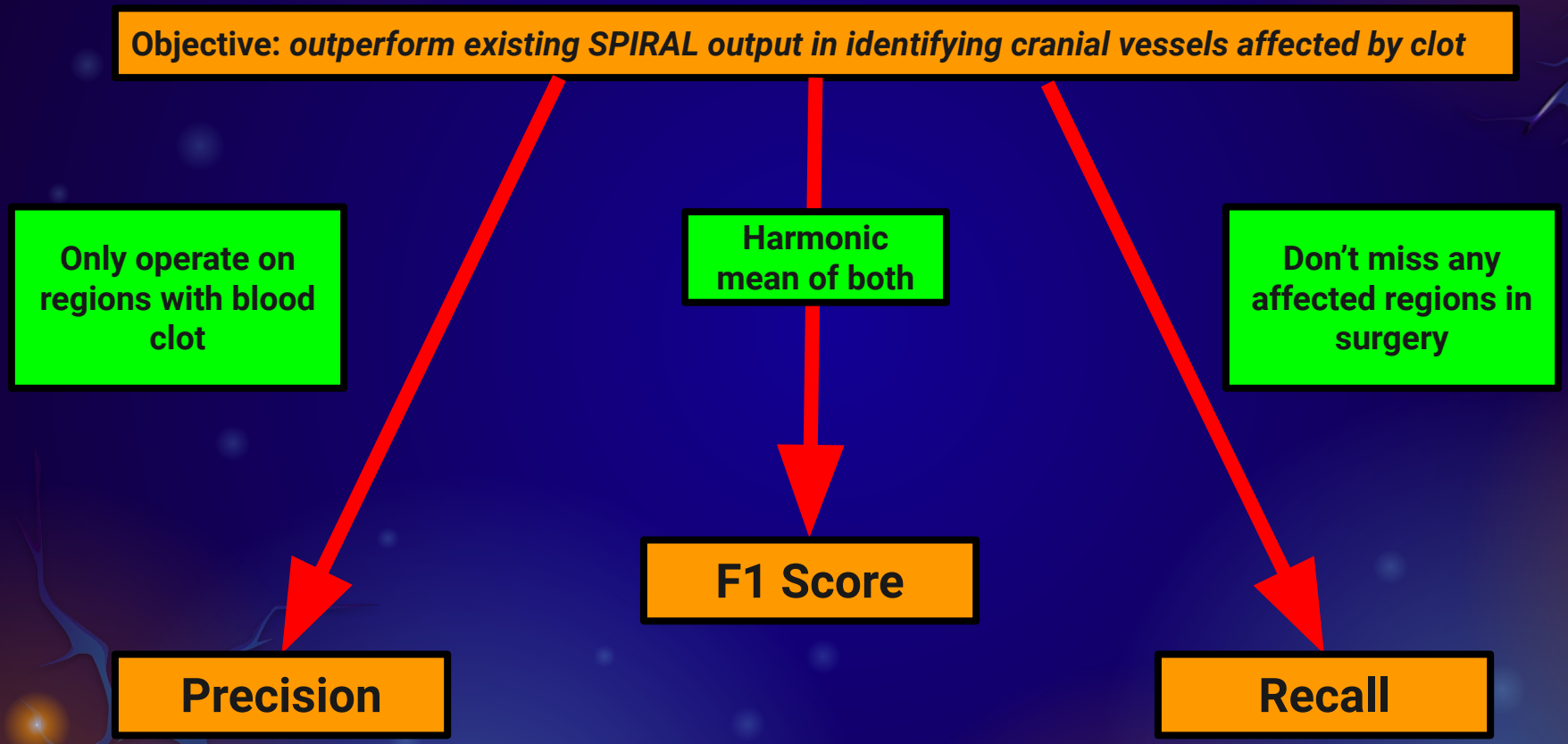
Label 2

Admire the clots

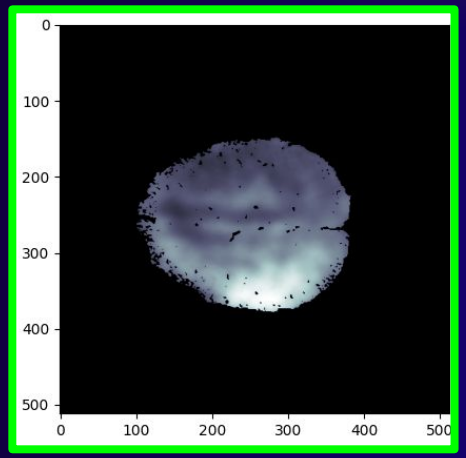
Success of Project



Success of Project

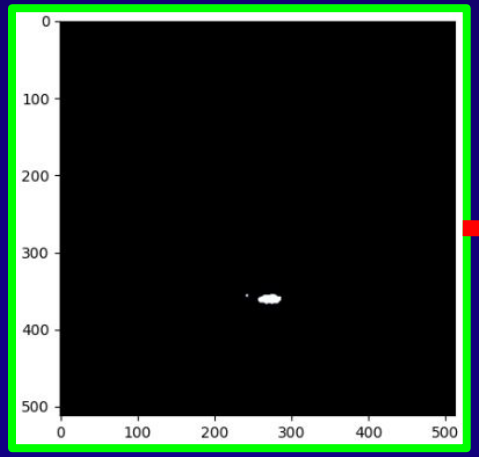


SPIRAL Data => Labels

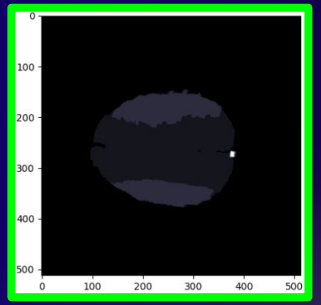


Start with SPIRAL Frame

Using Threshold



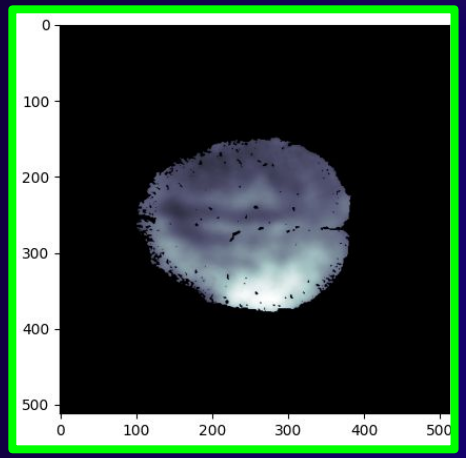
SPIRAL Mask (higher is more likely penumbra)



ATLAS Map

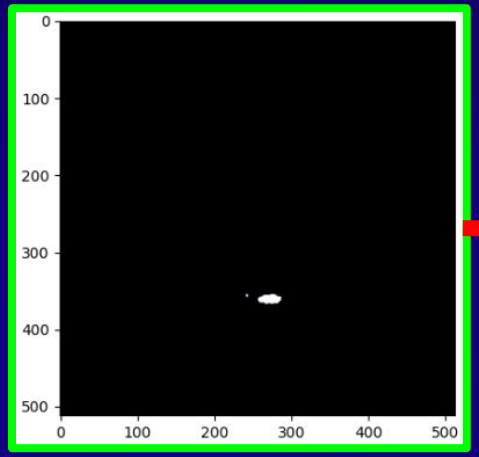
Corresponding ATLAS Regions: [0. 0. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

SPIRAL Data => Labels

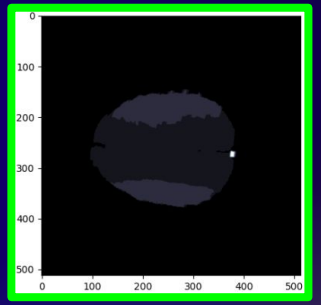


Start with SPIRAL Frame

Using Threshold



SPIRAL Mask (higher is more likely penumbra)



ATLAS Map

Corresponding ATLAS Regions: [0. 0. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

SPIRAL Baseline Results

```
Threshold: 0.0. Prec: 0.15765422696114564. Recall: 0.7089041095887983. Accuracy: 0.21386138613859973. F1 Score: 0.25794392520384624
Threshold: 0.1. Prec: 0.15765422696114564. Recall: 0.7089041095887983. Accuracy: 0.21386138613859973. F1 Score: 0.25794392520384624
Threshold: 0.2. Prec: 0.15765422696114564. Recall: 0.7089041095887983. Accuracy: 0.21386138613859973. F1 Score: 0.25794392520384624
Threshold: 0.3. Prec: 0.15765422696114564. Recall: 0.7089041095887983. Accuracy: 0.21386138613859973. F1 Score: 0.25794392520384624
Threshold: 0.4. Prec: 0.15765422696114564. Recall: 0.7089041095887983. Accuracy: 0.21386138613859973. F1 Score: 0.25794392520384624
Threshold: 0.5. Prec: 0.15765422696114564. Recall: 0.7089041095887983. Accuracy: 0.21386138613859973. F1 Score: 0.25794392520384624
Threshold: 0.6. Prec: 0.1558544303797345. Recall: 0.6746575342463443. Accuracy: 0.23300330033001762. F1 Score: 0.25321336757873314
Threshold: 0.7. Prec: 0.16593886462880647. Recall: 0.6506849315066265. Accuracy: 0.30231023102308235. F1 Score: 0.2644398051171984
Threshold: 0.8. Prec: 0.18794326241132528. Recall: 0.544520547945019. Accuracy: 0.45874587458742844. F1 Score: 0.27943760980362825
Threshold: 0.9. Prec: 0.33439490445849224. Recall: 0.3595890410957673. Accuracy: 0.7386138613860899. F1 Score: 0.34653465341529804
Threshold: 0.95. Prec: 0.5443037974676654. Recall: 0.1472602739725523. Accuracy: 0.8118811881187583. F1 Score: 0.23180592988549342
```

We find the optimal threshold on a train set

```
Threshold: 0.9. Prec: 0.3281356496258744. Recall: 0.3529586743659843. Accuracy: 0.7286597436512987. F1 Score: 0.3369857469852367
```

Assess the threshold found in training on test set

SPIRAL Baseline Results

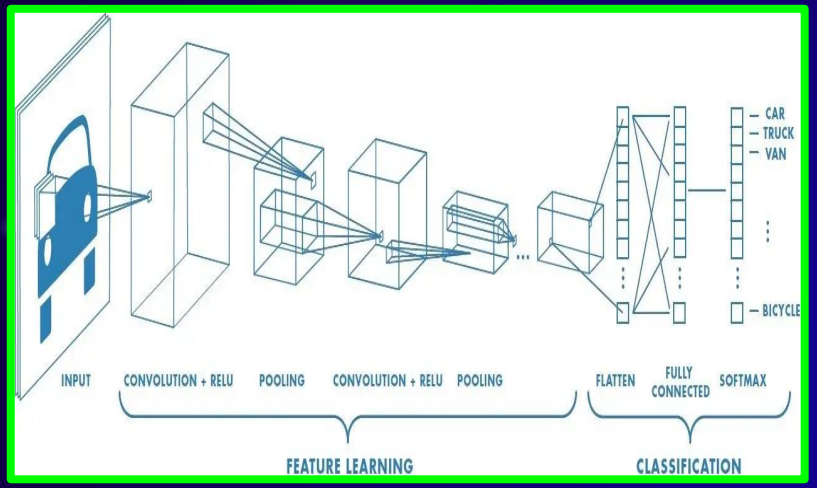
```
Threshold: 0.0. Prec: 0.15765422696114564. Recall: 0.7089041095887983. Accuracy: 0.21386138613859973. F1 Score: 0.25794392520384624
Threshold: 0.1. Prec: 0.15765422696114564. Recall: 0.7089041095887983. Accuracy: 0.21386138613859973. F1 Score: 0.25794392520384624
Threshold: 0.2. Prec: 0.15765422696114564. Recall: 0.7089041095887983. Accuracy: 0.21386138613859973. F1 Score: 0.25794392520384624
Threshold: 0.3. Prec: 0.15765422696114564. Recall: 0.7089041095887983. Accuracy: 0.21386138613859973. F1 Score: 0.25794392520384624
Threshold: 0.4. Prec: 0.15765422696114564. Recall: 0.7089041095887983. Accuracy: 0.21386138613859973. F1 Score: 0.25794392520384624
Threshold: 0.5. Prec: 0.15765422696114564. Recall: 0.7089041095887983. Accuracy: 0.21386138613859973. F1 Score: 0.25794392520384624
Threshold: 0.6. Prec: 0.1558544303797345. Recall: 0.6746575342463443. Accuracy: 0.23300330033001762. F1 Score: 0.25321336757873314
Threshold: 0.7. Prec: 0.16593886462880647. Recall: 0.6506849315066265. Accuracy: 0.30231023102308235. F1 Score: 0.2644398051171984
Threshold: 0.8. Prec: 0.18794326241132528. Recall: 0.544520547945019. Accuracy: 0.45874587458742844. F1 Score: 0.27943760980362825
Threshold: 0.9. Prec: 0.33439490445849224. Recall: 0.3595890410957673. Accuracy: 0.7386138613860899. F1 Score: 0.34653465341529804
Threshold: 0.95. Prec: 0.5443037974676654. Recall: 0.1472602739725523. Accuracy: 0.8118811881187583. F1 Score: 0.23180592988549342
```

We find the optimal threshold on a train set

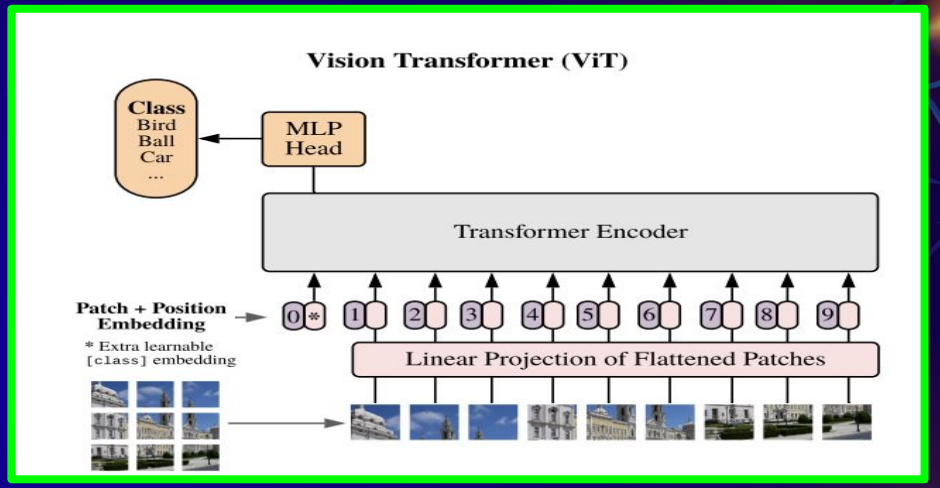
```
Threshold: 0.9. Prec: 0.3281356496258744. Recall: 0.3529586743659843. Accuracy: 0.7286597436512987. F1 Score: 0.3369857469852367
```

Assess the threshold found in training on test set

DL Computer Vision Models



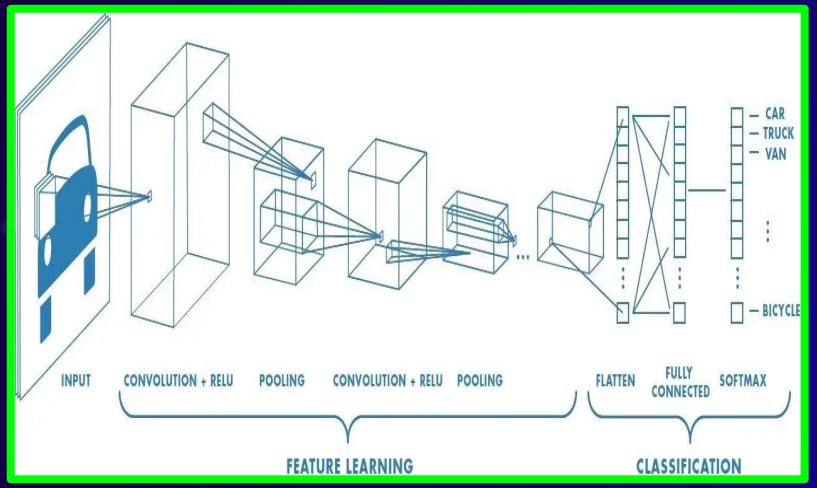
CNN



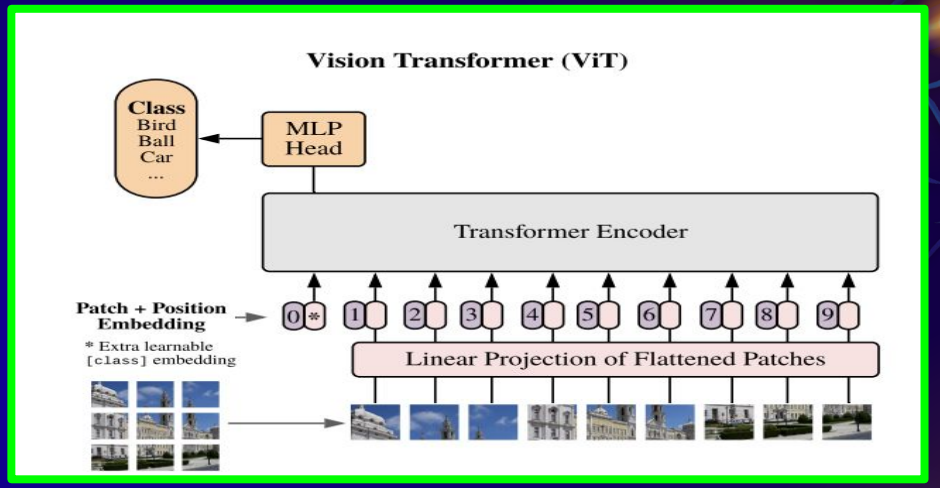
Vision Transformer

- CNNs and Vision Transformers: State-of-the-art in various tasks in medical imaging
- CNNs are efficient due to fewer (shared) parameters.
- Vision Transformers have achieved high performance in detecting retinal diseases and analyzing images for cancer diagnosis (similar domains) but require more data.

DL Computer Vision Models



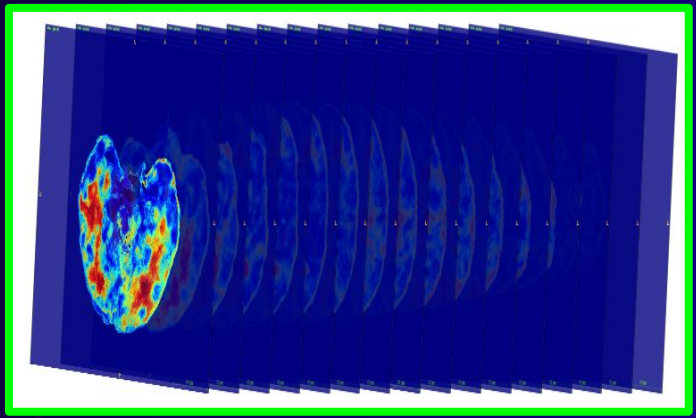
CNN



Vision Transformer

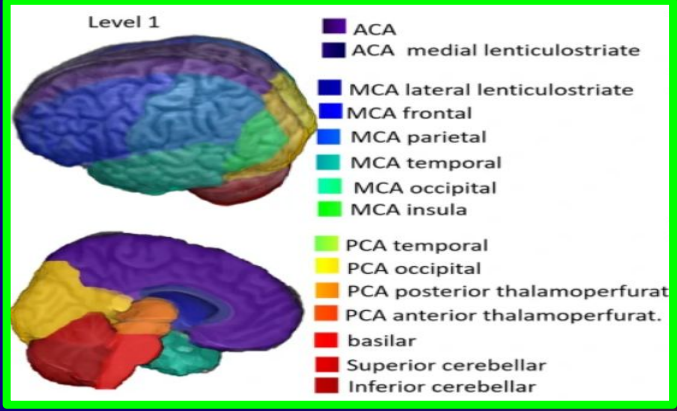
- CNNs and Vision Transformers: State-of-the-art in various tasks in medical imaging
- CNNs are efficient due to fewer (shared) parameters.
- Vision Transformers have achieved high performance in detecting retinal diseases and analyzing images for cancer diagnosis (similar domains) but require more data.

How do we apply DL here?



Model

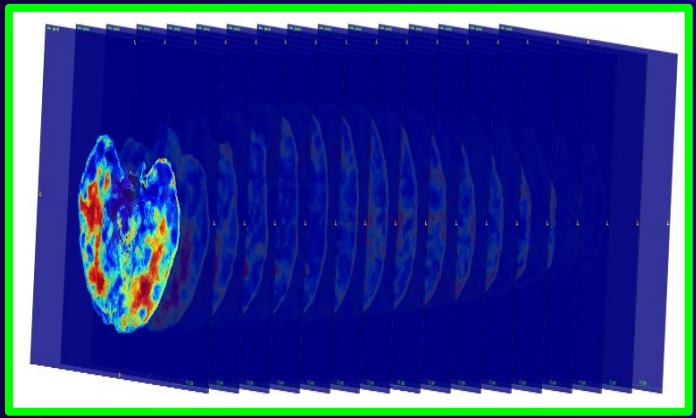
1 Baseline CNN
2 MobileNet
3 EfficientNet



Input can be i) frame, ii) all frames iii) sliding window of frames (e.g 5 frames per input)

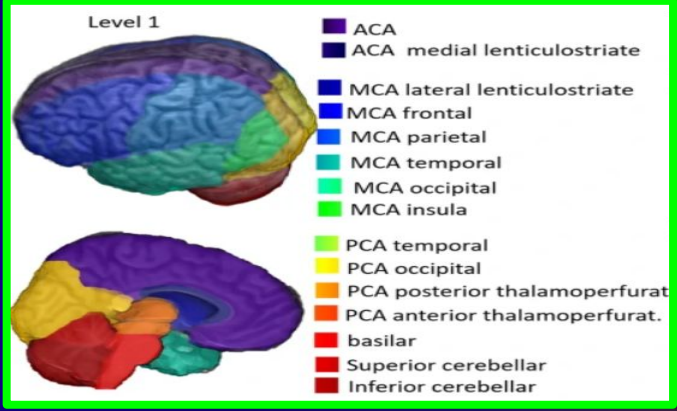
Output is length 15 array of probabilities of each region being affected

How do we apply DL here?



Model

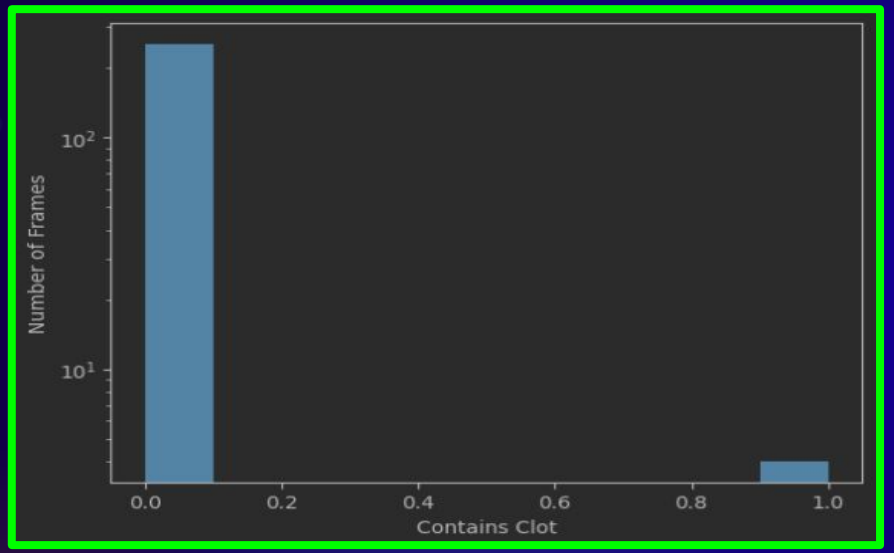
1 Baseline CNN
2 MobileNet
3 EfficientNet



Input can be i) frame, ii) all frames iii) sliding window of frames (e.g 5 frames per input)

Output is length 15 array of probabilities of each region being affected

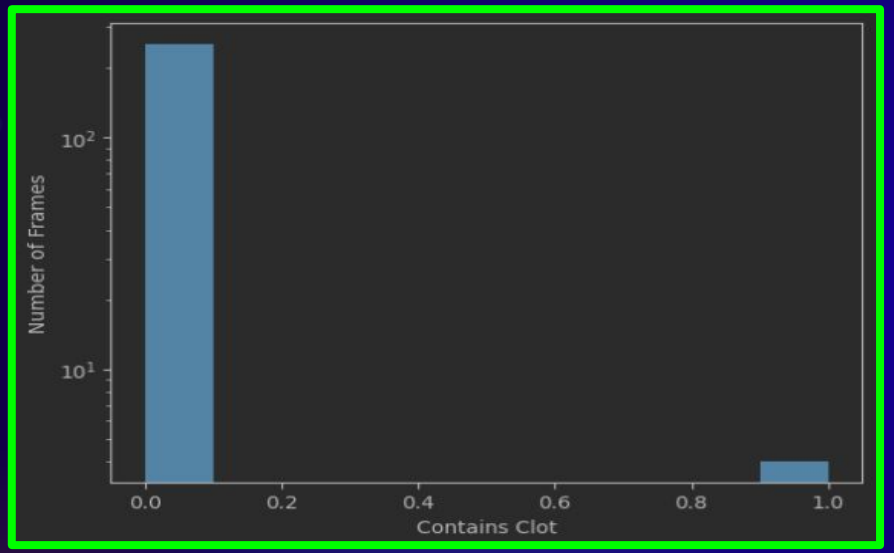
Potential Roadblocks



Imbalanced Dataset

CUDA Memory Issues

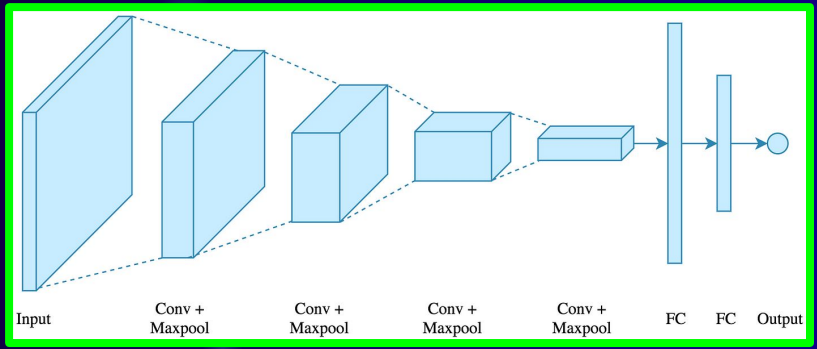
Potential Roadblocks



Imbalanced Dataset

CUDA Memory Issues

Model 1: Baseline CNN

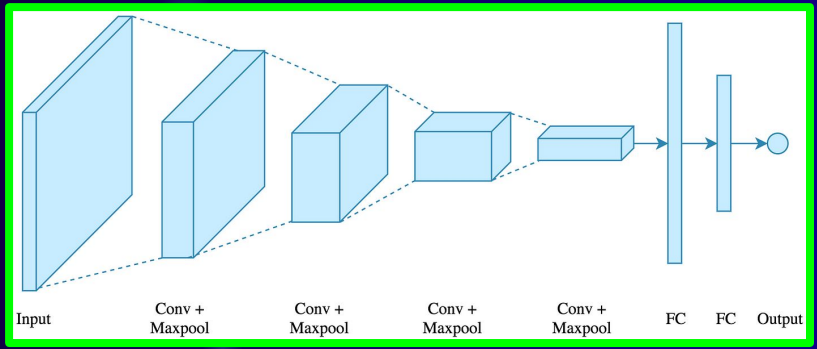


Layer	No. of Kernels	Kernel Size	Activation
Input	1	Input Shape	-
Convolution2D	32	5×5	ReLu
Maxpooling2D	-	2×2	-
Convolution2D	32	5×5	ReLu
Maxpooling2D	-	2×2	-
Fully Connected	1024	-	Dropout (0.25)
Fully Connected	2	-	ReLu
SoftMax	-	-	-

Model Architecture

The layers composing our model, taken from the paper (Zivkovic et al., *Hybrid CNN and XGBoost Model Tuned by Modified Arithmetic Optimization Algorithm for COVID-19 Early Diagnostics from X-ray Images*, 2022)

Model 1: Baseline CNN

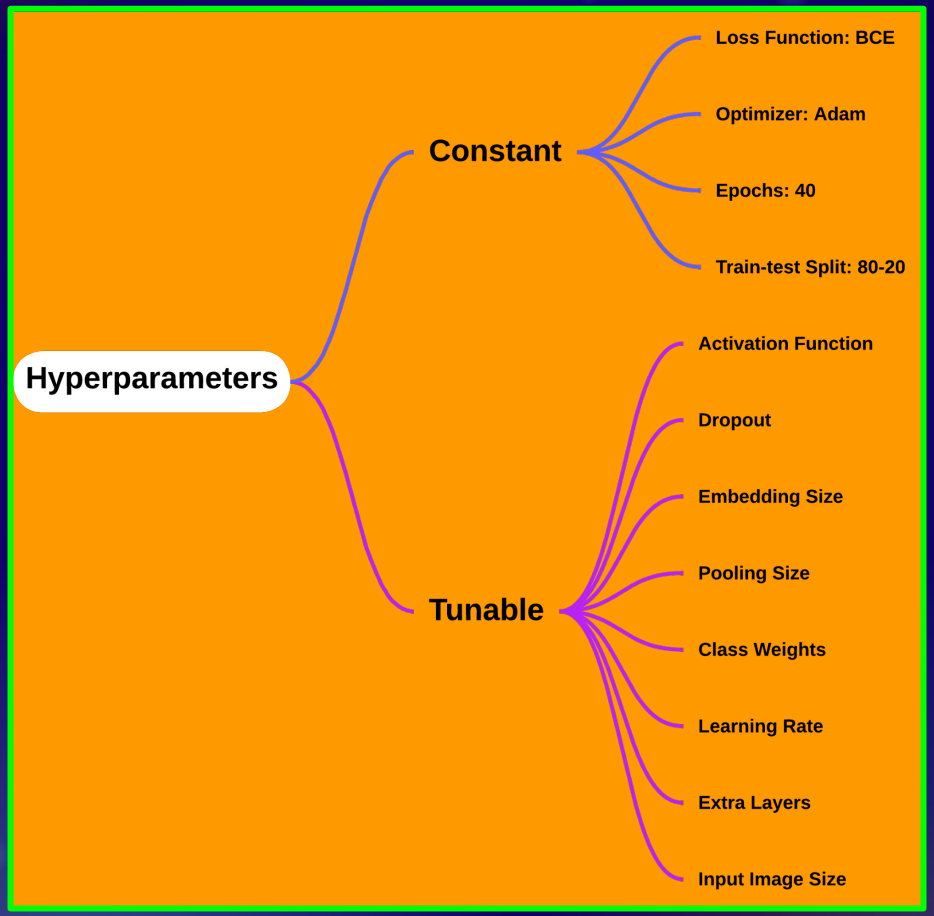


Layer	No. of Kernels	Kernel Size	Activation
Input	1	Input Shape	-
Convolution2D	32	5×5	ReLu
Maxpooling2D	-	2×2	-
Convolution2D	32	5×5	ReLu
Maxpooling2D	-	2×2	-
Fully Connected	1024	-	Dropout (0.25)
Fully Connected	2	-	ReLu
SoftMax	-	-	-

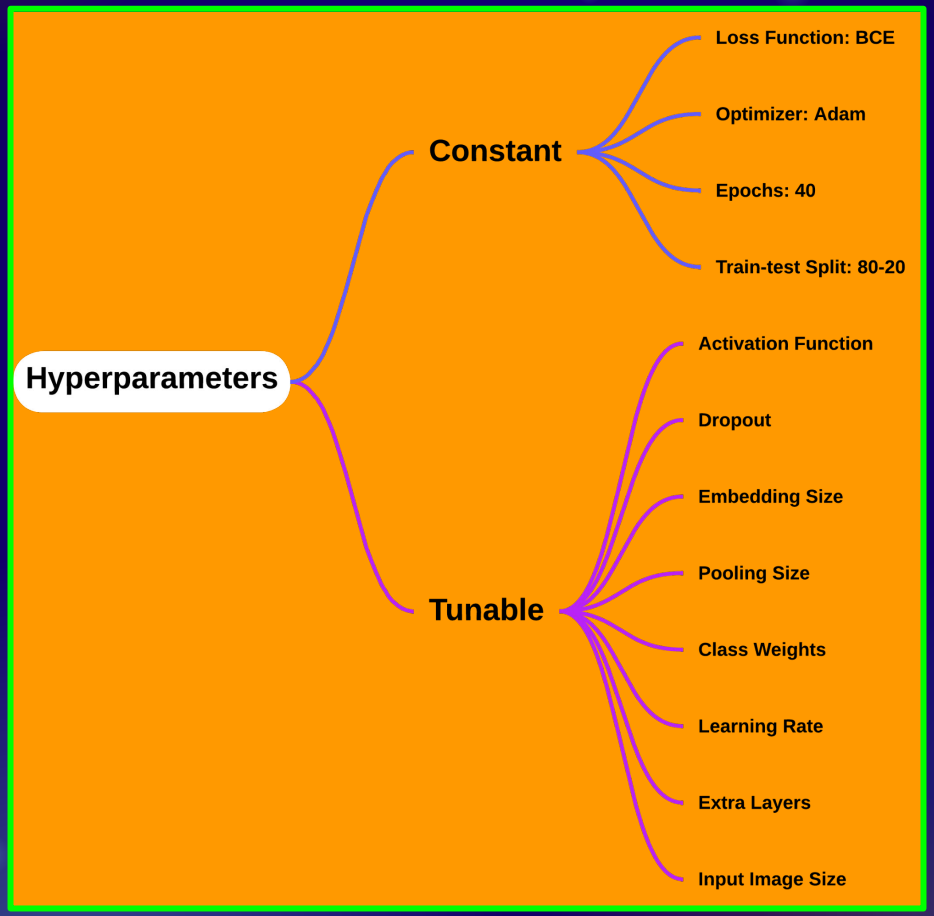
Model Architecture

The layers composing our model, taken from the paper (Zivkovic et al., *Hybrid CNN and XGBoost Model Tuned by Modified Arithmetic Optimization Algorithm for COVID-19 Early Diagnostics from X-ray Images*, 2022)

Model Optimization



Model Optimization



Hyperparameter Tuning

pool_dim	feat_dim	activation	lr	dropout_prob	pos_weight
4	256	relu	0.000512239	0.340438	17.428
4	256	sigmoid	0.00107452	0.181308	12.9439
4	256	tanh	0.00487844	0.307489	15.844
4	512	tanh	0.00142236	0.46978	13.3317
4	256	sigmoid	0.00234546	0.155405	13.8587
4	512	sigmoid	0.00094525	0.288003	14.2234
4	512	sigmoid	0.00067001	0.121338	15.4934
4	512	relu	0.00189603	0.123148	16.2624
4	512	relu	0.00350341	0.186474	15.2463
4	256	sigmoid	0.00210993	0.402286	16.6853

```yaml
hyperparameters:
  activation: sigmoid
  batch: 128
  feat_dim: 32
  lr: 0.0001
  model: CNN_Feature_Extractor
  num_channels: 210
  optimizer_name: adam
  pooling_size: 2
  drop_out_rate: 0.2
```

RayTune Library for Tuning

Tuned Hyperparameters

Hyperparameter Tuning

pool_dim	feat_dim	activation	lr	dropout_prob	pos_weight
4	256	relu	0.000512239	0.340438	17.428
4	256	sigmoid	0.00107452	0.181308	12.9439
4	256	tanh	0.00487844	0.307489	15.844
4	512	tanh	0.00142236	0.46978	13.3317
4	256	sigmoid	0.00234546	0.155405	13.8587
4	512	sigmoid	0.00094525	0.288003	14.2234
4	512	sigmoid	0.00067001	0.121338	15.4934
4	512	relu	0.00189603	0.123148	16.2624
4	512	relu	0.00350341	0.186474	15.2463
4	256	sigmoid	0.00210993	0.402286	16.6853

```yaml
hyperparameters:
  activation: sigmoid
  batch: 128
  feat_dim: 32
  lr: 0.0001
  model: CNN_Feature_Extractor
  num_channels: 210
  optimizer_name: adam
  pooling_size: 2
  drop_out_rate: 0.2
```

RayTune Library for Tuning

Tuned Hyperparameters

Hyperparameter Tuning Results

pool_dim	feat_dim	activation	lr	dropout_prob	pos_weight	extra_layers	resize_dims	iter	total time (s)	Val_F1	Val_Precision	Val_Recall	Val_Accuracy
4	512	tanh	0.00132562	0.318361	17.7171	0	(128, 128)	8	79.8011	0.327614	0.199568	0.914191	0.963328
4	512	relu	0.000608253	0.387794	15.2612	2	(128, 128)	20	106.681	0.598461	0.459725	0.857143	0.989873
4	512	sigmoid	0.00312816	0.196748	15.5842	2	(512, 512)	4	288.843	0.184887	0.353612	0.125162	0.986776
4	512	relu	0.00480434	0.131557	12.2852	4	(512, 512)	8	478.249	0.401463	0.271881	0.767075	0.978939
4	512	sigmoid	0.000681346	0.166743	13.9148	2	(128, 128)	4	67.417	0.168809	0.106594	0.405498	0.962522

Results using SPIRAL as Input

pool_dim	feat_dim	activation	lr	dropout_prob	pos_weight	extra_layers	resize_dims	iter	total time (s)	Val_F1	Val_Precision	Val_Recall	Val_Accuracy
4	512	tanh	0.00109909	0.248508	12.9139	2	(256, 256)	20	325.903	0.620511	0.514077	0.78254	0.990276
4	512	relu	0.00186225	0.308561	13.4125	2	(256, 256)	16	298.293	0.474085	0.322626	0.893617	0.98347
4	512	sigmoid	0.000671123	0.294085	14.9742	2	(512, 512)	4	252.012	0.191932	0.159434	0.241084	0.977052
4	512	tanh	0.000878918	0.274336	15.6468	2	(128, 128)	20	132.717	0.666351	0.514059	0.946879	0.988486
4	512	sigmoid	0.00141349	0.321913	14.2752	2	(128, 128)	4	52.2891	0.355104	0.229973	0.778963	0.970069

Results using raw CTA as Input

Hyperparameter Tuning Results

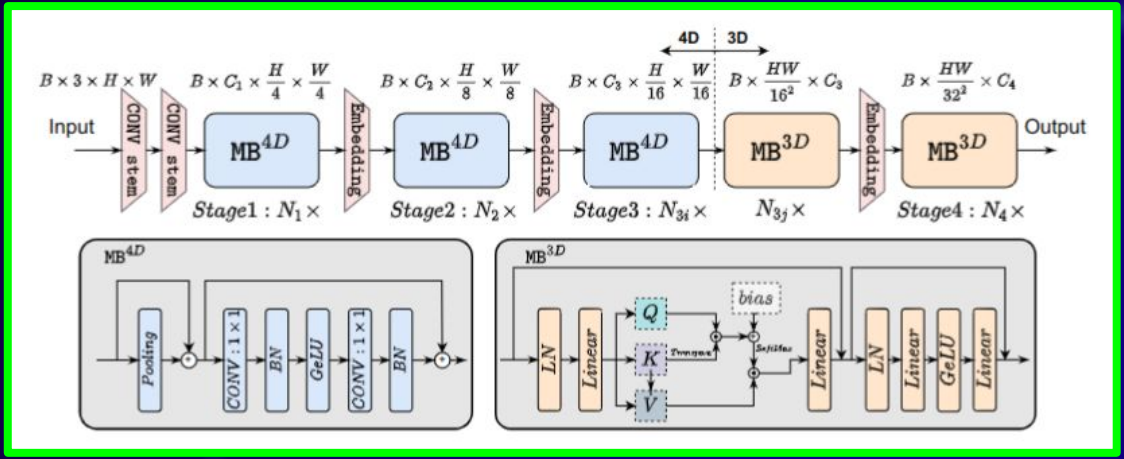
pool_dim	feat_dim	activation	lr	dropout_prob	pos_weight	extra_layers	resize_dims	iter	total time (s)	Val_F1	Val_Precision	Val_Recall	Val_Accuracy
4	512	tanh	0.00132562	0.318361	17.7171	0	(128, 128)	8	79.8011	0.327614	0.199568	0.914191	0.963328
4	512	relu	0.000608253	0.387794	15.2612	2	(128, 128)	20	106.681	0.598461	0.459725	0.857143	0.989873
4	512	sigmoid	0.00312816	0.196748	15.5842	2	(512, 512)	4	288.843	0.184887	0.353612	0.125162	0.986776
4	512	relu	0.00480434	0.131557	12.2852	4	(512, 512)	8	478.249	0.401463	0.271881	0.767075	0.978939
4	512	sigmoid	0.000681346	0.166743	13.9148	2	(128, 128)	4	67.417	0.168809	0.106594	0.405498	0.962522

Results using SPIRAL as Input

pool_dim	feat_dim	activation	lr	dropout_prob	pos_weight	extra_layers	resize_dims	iter	total time (s)	Val_F1	Val_Precision	Val_Recall	Val_Accuracy
4	512	tanh	0.00109909	0.248508	12.9139	2	(256, 256)	20	325.903	0.620511	0.514077	0.78254	0.990276
4	512	relu	0.00186225	0.308561	13.4125	2	(256, 256)	16	298.293	0.474085	0.322626	0.893617	0.98347
4	512	sigmoid	0.000671123	0.294085	14.9742	2	(512, 512)	4	252.012	0.191932	0.159434	0.241084	0.977052
4	512	tanh	0.000878918	0.274336	15.6468	2	(128, 128)	20	132.717	0.666351	0.514059	0.946879	0.988486
4	512	sigmoid	0.00141349	0.321913	14.2752	2	(128, 128)	4	52.2891	0.355104	0.229973	0.778963	0.970069

Results using raw CTA as Input

Model 2: EfficientFormerV2



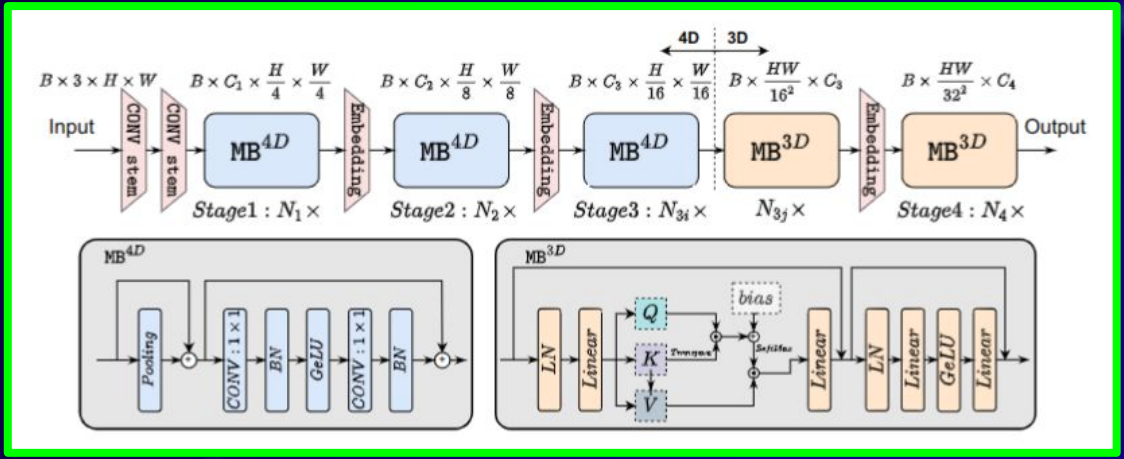
Model Architecture

```yaml
hyperparameters:
    batch_size: 32
    lr: 0.0015634
    pos_weight: 15.9876
```

Tuned Hyperparameters

- Similar to previous slide, use pre-trained EfficientFormerV2 as feature extractor
- Add classification layer on top
- Tune classifier from scratch and finetune architecture

Model 2: EfficientFormerV2



Model Architecture

```yaml
hyperparameters:
    batch_size: 32
    lr: 0.0015634
    pos_weight: 15.9876
```

Tuned Hyperparameters

- Similar to previous slide, use pre-trained EfficientFormerV2 as feature extractor
- Add classification layer on top
- Tune classifier from scratch and finetune architecture

Training Results

lr	pos_weight	resize_dims	iter	total_time (s)	Val_F1	Val_Precision	Val_Recall	Val_Accuracy
0.000855178	16.6069	(224, 224)	1	79.4564	0.0598473	0.154839	0.0370943	0.987841
0.00132556	13.5432	(224, 224)	20	862.509	0.600391	0.438462	0.951983	0.990211
0.00199256	12.2775	(224, 224)	20	859.569	0.57637536	0.42092352	0.91390368	0.95060256
0.00109437	16.3686	(224, 224)	20	849.987	0.4803128	0.3507696	0.7615864	0.96050467
0.00275423	13.6704	(224, 224)	20	865.269	0.3602346	0.2630772	0.5711898	0.97040678

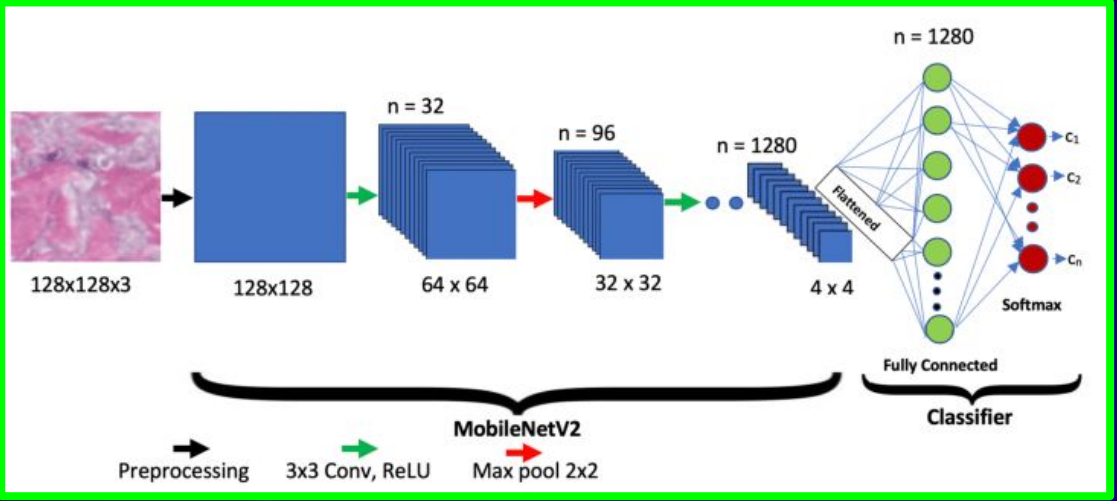
Results using raw CTA as Input

Training Results

lr	pos_weight	resize_dims	iter	total_time (s)	Val_F1	Val_Precision	Val_Recall	Val_Accuracy
0.000855178	16.6069	(224, 224)	1	79.4564	0.0598473	0.154839	0.0370943	0.987841
0.00132556	13.5432	(224, 224)	20	862.509	0.600391	0.438462	0.951983	0.990211
0.00199256	12.2775	(224, 224)	20	859.569	0.57637536	0.42092352	0.91390368	0.95060256
0.00109437	16.3686	(224, 224)	20	849.987	0.4803128	0.3507696	0.7615864	0.96050467
0.00275423	13.6704	(224, 224)	20	865.269	0.3602346	0.2630772	0.5711898	0.97040678

Results using raw CTA as Input

Model 3: MobileNetV2



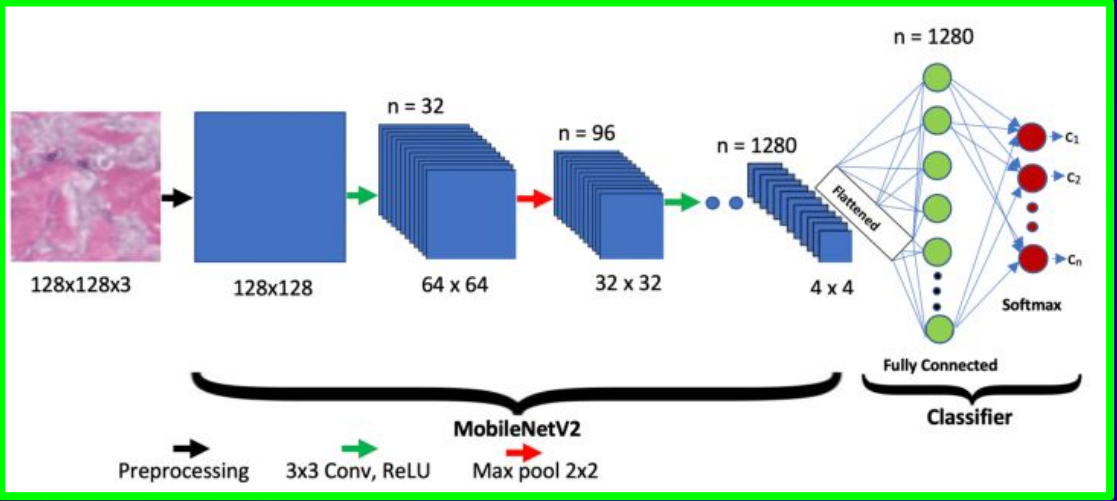

```yaml
hyperparameters:
    input_image_size: (224, 224)
    batch_size: 32
    lr: 0.0015634
    pos_weight: 15.9876
```

Model Architecture

Tuned Hyperparameters

- **Transfer learning: use pre-trained MobileNetV2 architecture trained on Imagenet as feature extractor**
- **Add classification layer on top**
- **Make all layers trainable; finetuning MobileNetV2 architecture**

Model 3: MobileNetV2




```yaml
hyperparameters:
    input_image_size: (224, 224)
    batch_size: 32
    lr: 0.0015634
    pos_weight: 15.9876
```

Model Architecture

Tuned Hyperparameters

- **Transfer learning: use pre-trained MobileNetV2 architecture trained on Imagenet as feature extractor**
- **Add classification layer on top**
- **Make all layers trainable; finetuning MobileNetV2 architecture**

Training Results

lr	pos_weight	resize_dims	iter	total_time (s)	Val_F1	Val_Precision	Val_Recall	Val_Accuracy
0.000674431	12.7152	(224, 224)	20	589.63	0.770024	0.650438	0.943503	0.993566
0.000700611	15.5943	(224, 224)	20	580.24	0.789864	0.678954	0.955524	0.994589
0.00101701	14.1608	(224, 224)	20	583.94	0.7315228	0.6179161	0.89632785	0.9438877
0.00327364	16.4515	(224, 224)	20	586.57	0.6930216	0.5853942	0.8491527	0.9438877
0.00332455	17.0229	(224, 224)	20	596.54	0.5390168	0.4553066	0.6604521	0.8942094

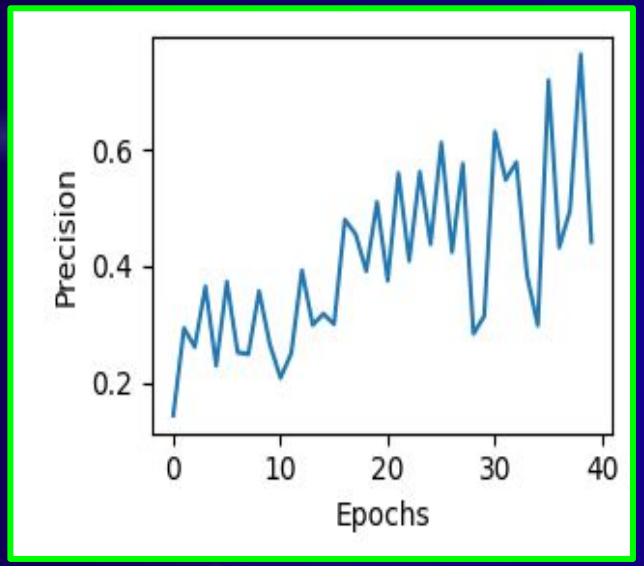
Results using raw CTA as Input

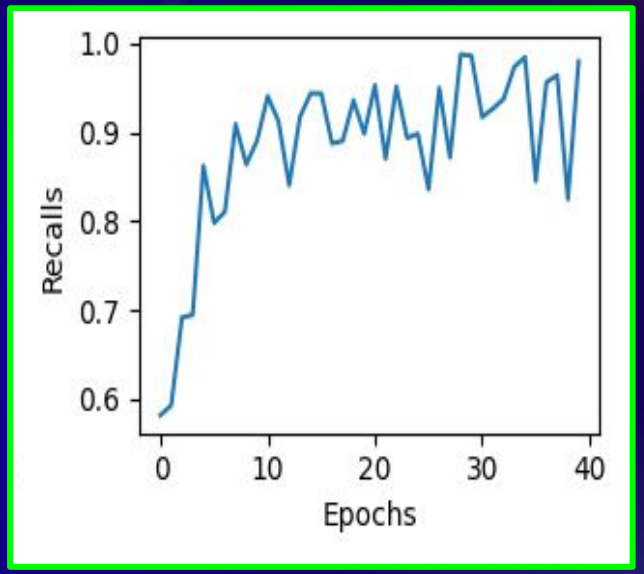
Training Results

lr	pos_weight	resize_dims	iter	total_time (s)	Val_F1	Val_Precision	Val_Recall	Val_Accuracy
0.000674431	12.7152	(224, 224)	20	589.63	0.770024	0.650438	0.943503	0.993566
0.000700611	15.5943	(224, 224)	20	580.24	0.789864	0.678954	0.955524	0.994589
0.00101701	14.1608	(224, 224)	20	583.94	0.7315228	0.6179161	0.89632785	0.9438877
0.00327364	16.4515	(224, 224)	20	586.57	0.6930216	0.5853942	0.8491527	0.9438877
0.00332455	17.0229	(224, 224)	20	596.54	0.5390168	0.4553066	0.6604521	0.8942094

Results using raw CTA as Input

Analysis of Results: Precision & Recall

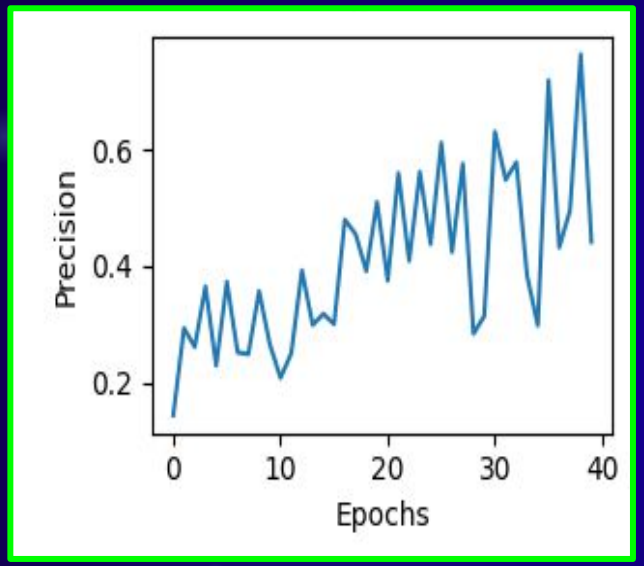


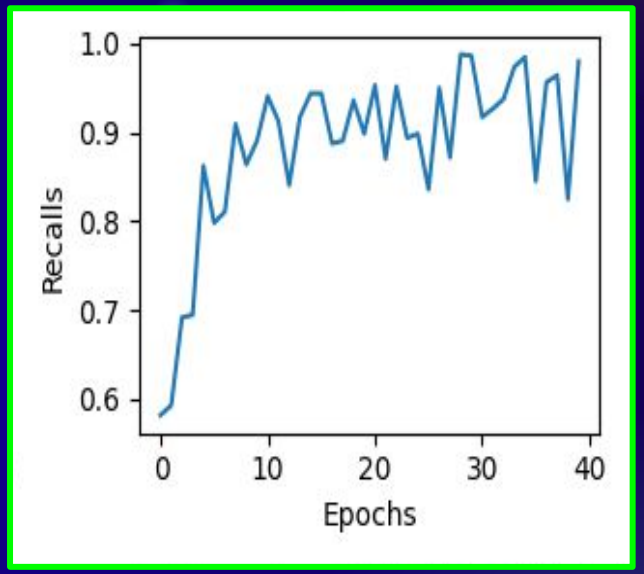


Precision

Recall

Analysis of Results: Precision & Recall

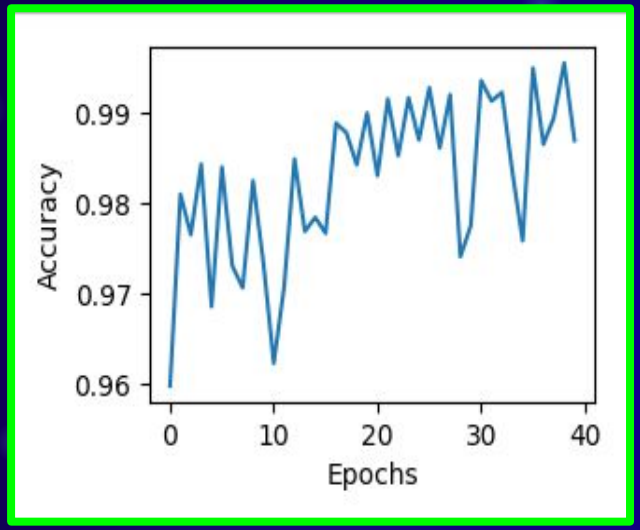


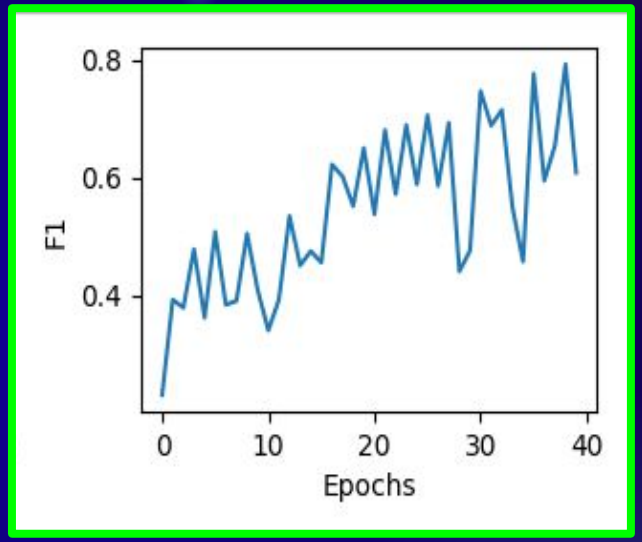


Precision

Recall

Analysis of Results: Accuracy & F1 Score

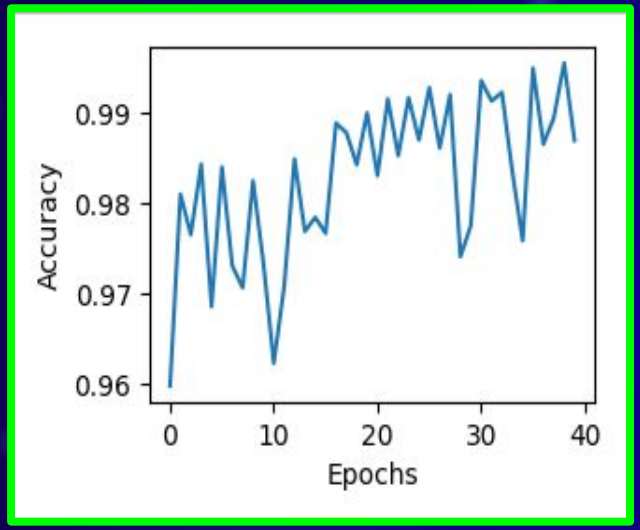


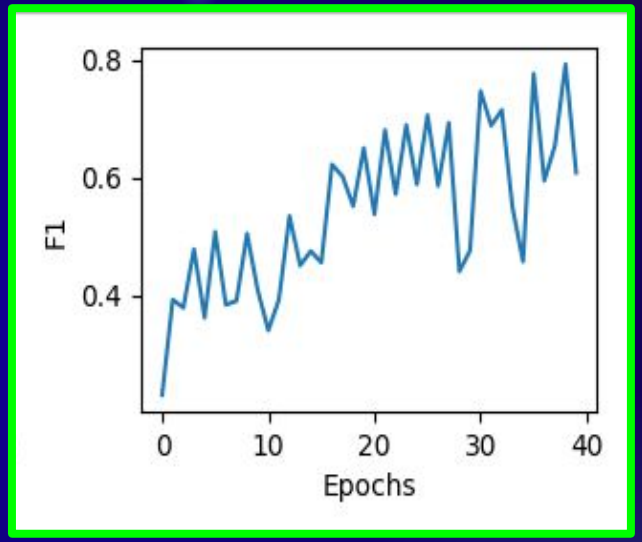


Accuracy

F1 Score

Analysis of Results: Accuracy & F1 Score

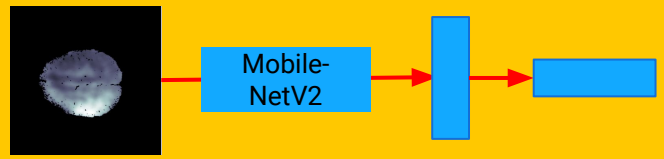
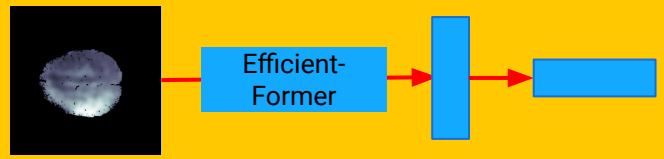




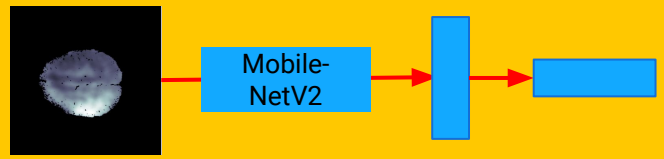
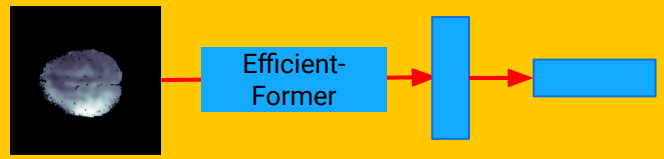
Accuracy

F1 Score

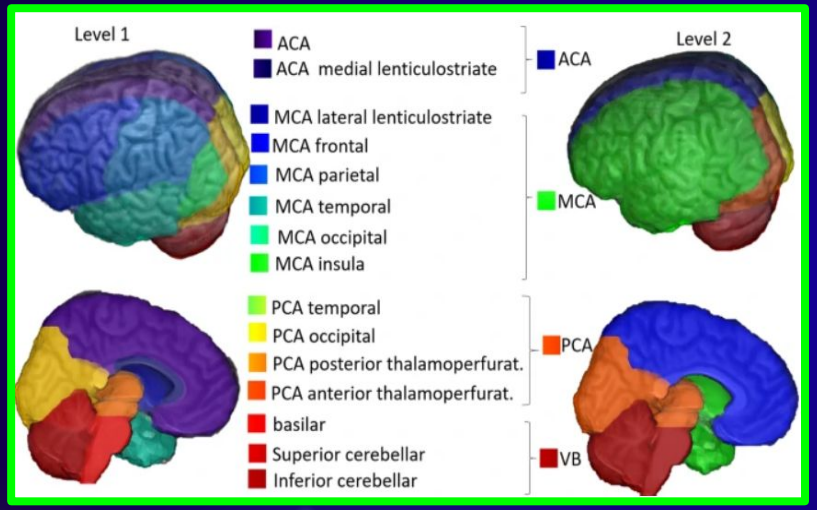
Summary of Results

Approach	Diagram	Best Result
Thresholding SPIRAL and extracting corresponding brain regions		**F1: 33.69%, Prec: 32.8%, Rec: 35.29%**
Train a Simple CNN		**F1: 66.64%, Prec: 51.41%, Rec: 94.69%**
Fine-tuning MobileNetV2 as feature extractor		**F1: 78.99%, Prec: 67.90%, Rec: 95.55%**
Fine-tuning EfficientFormerV2 as feature extractor		**F1: 60.03%, Prec: 43.85%, Rec: 95.20%**

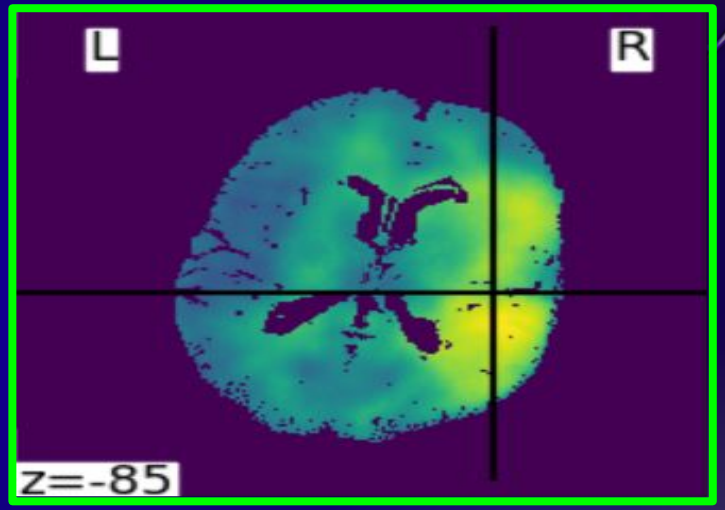
Summary of Results

Approach	Diagram	Best Result
Thresholding SPIRAL and extracting corresponding brain regions		**F1: 33.69%, Prec: 32.8%, Rec: 35.29%**
Train a Simple CNN		**F1: 66.64%, Prec: 51.41%, Rec: 94.69%**
Fine-tuning MobileNetV2 as feature extractor		**F1: 78.99%, Prec: 67.90%, Rec: 95.55%**
Fine-tuning EfficientFormerV2 as feature extractor		**F1: 60.03%, Prec: 43.85%, Rec: 95.20%**

Discussion: Usefulness

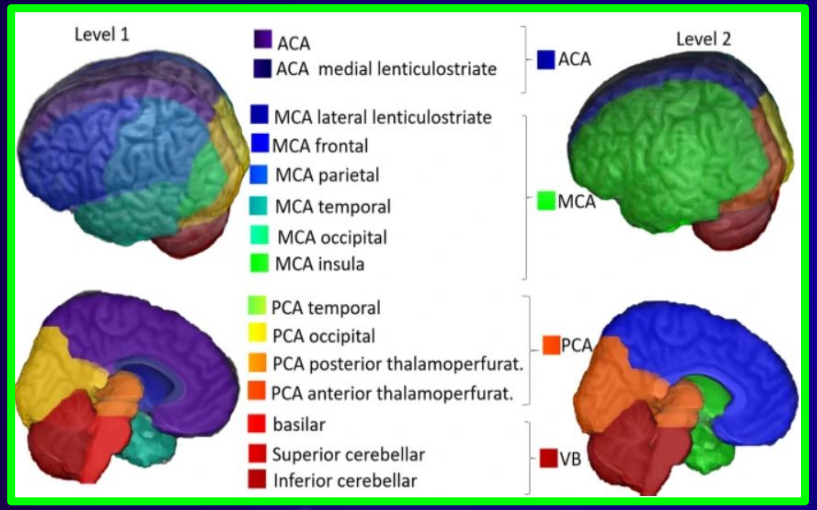


Output = regions

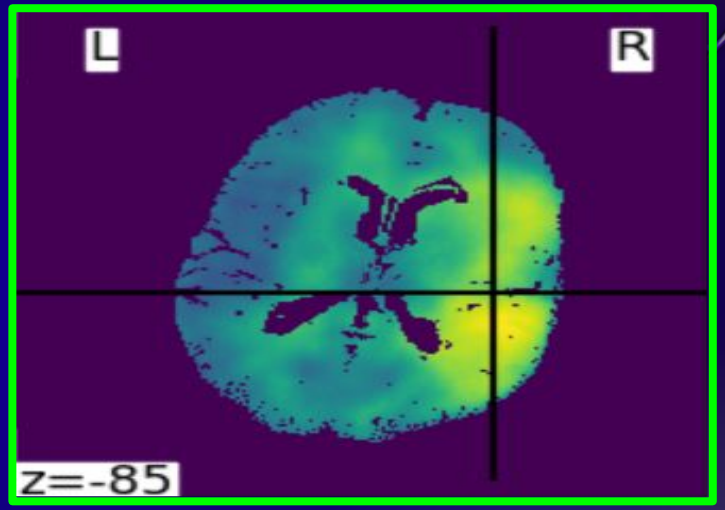


Spiral Scans (AMI)

Discussion: Usefulness

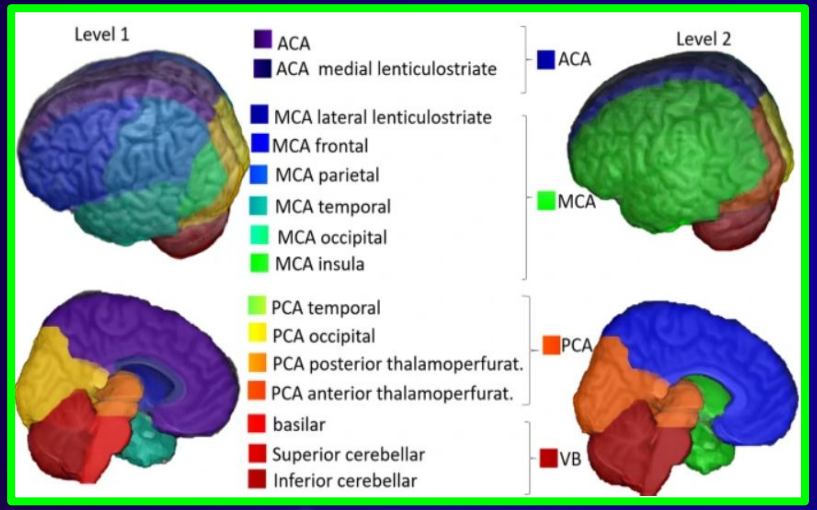


Output = regions

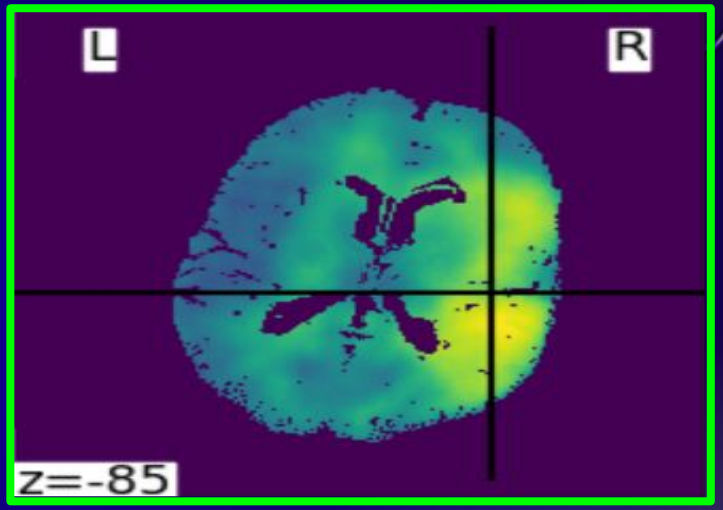


Spiral Scans (AMI)

Discussion: Limitations

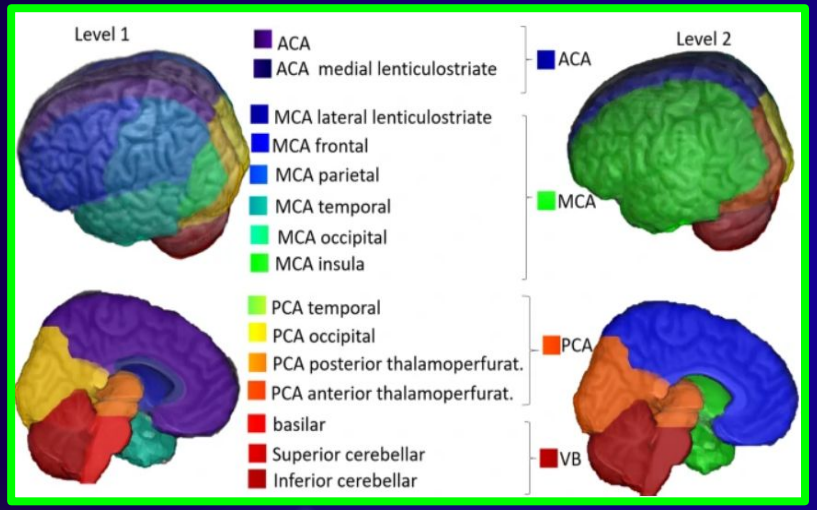


Output = regions

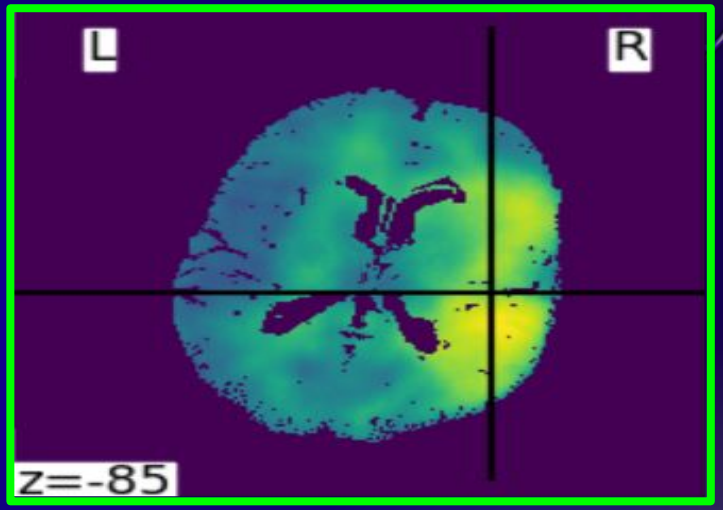


Spiral Scans (AMI)

Discussion: Limitations



Output = regions



Spiral Scans (AMI)

Discussion: Industry Competitors

Discussion: Industry Competitors

THANKS FOR WATCHING