nnU-Net for Brain Tumor Segmentation

Connor Cole, Shen Zhu, Zoraiz Qureshi - Group No. 5
cdc3hf, sz9jt, zce5py

Training Setup & Parameters

Pre-processing

1. **Maximal pessimistic bounding box crop:** All of the training data images had excess empty space around their border. This empty space unnecessarily increases the size of each training data point and slows down the training speed. To remedy this, the training images were indiscriminately and pessimistically (empirically found 15 pixels from each side or 30 per axis) cropped along the edges of the frame by computing a maximal bounding box to reduce the overall size of the image, without eliminating any of the key data: the brain being scanned and annotated. We assume the test set follows a similar image distribution due to affine registration as the training images so should stay within these crop bounds. This operation omitted extraneous data and sped up the training process by simplifying the training data.

2. **Rescaling to 128*128 with nearest-neighbor interpolation:** Following the structure of the CNN of nnU-Net, and to reduce the overall dimensions of the training images as done in the original paper for computational efficiency, the images were rescaled to 128 x 128 pixels. This was done using the nearest neighbor interpolation method for resizing images and corresponding labels. This increased the computational efficiency of the network's training due to the smaller size of the overall structure and individual training images, enabling us to do a thorough evaluation with multiple training folds.

3. **Mean-std normalization**: The normalization process we used when preprocessing the data was the mean-std normalization method. This method involved subtracting by the mean of the data and dividing by the standard deviation of the data to normalize the data consistently across the dataset.

4. **Label change**: In our implementation, we changed labels 4 to 3 for consistency and nnUNet data processing requirements. This is because the original annotation labels were {0,1,2,4}, so we changed them to {0,1,2,3}

We also evaluate with and without pre-processing steps 1 and 2 (Task500, Task501 respectively in code setup).

Network architecture

Network parameters for final pre-processed 128x128 images:

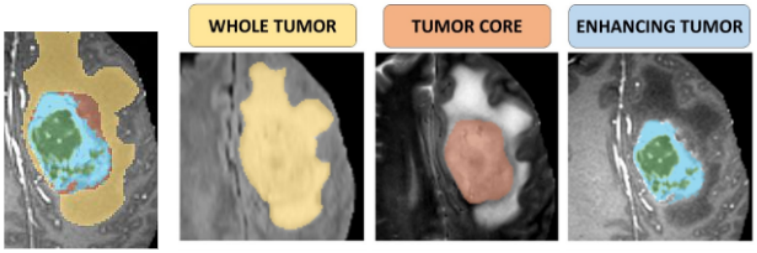
- Generic U-Net - with 32 base filters, 3 convolutional layers (3x3 kernels, 2x2 max pool) - based on nnUNet rule that features dimensions cannot be smaller than 4x4 in latent space
- Dice+BCE hybrid weighted loss, different loss class weights (based on class sample pixels)
- Batch size - 64
- Decaying LR 1e-2, SGD, weight_decay 3e-5
- 100 epochs

- default affine data augmentation (scaling, rotation, gamma noise)

For the architecture without pre-processing (240x240 images), batch size was fixed to 50 automatically by nnUNet because of GPU limitations, and an additional convolutional layer was added in the encoder-decoder. We also edited the main trainer class from which other classes are extended to restrict epochs to 100 consistently instead of 1000 (due to time constraints, training can go to almost 300+ epochs otherwise due to SGD optimizer and decaying learning rate).

Training improvement modules

Region-based training (R) - dice scores on labels evaluated by the BraTS backend with partially overlapping classes whole tumor, tumor core, and enhanced tumor with sigmoid instead of softmax and separate dice for training.



Batch normalization (BN) - effective for deeper networks, and in combination with data augmentation
Batch dice (BD) - Dice loss over the entire training batch, instead of sample dice
Data augmentation (DA) - All kinds of generic data augmentations for images defined as insane DA2 within nnUNet - with parameterized rotation, scaling, elastic deformation, random crop, gamma transform, additive and multiplicative brightness, low-resolution transform, contrast, gaussian noise & blur, axes mirror**.**
More data augmentation (DA*) - increase variety & probabilities aggressively compared to main data augmentation **(DA)** for the BraTS 2020

- increase the probability of applying rotation and scaling from 0.2 to 0.3
- increase the scale range from (0.85, 1.25) to (0.65, 1.6)
- select a scaling factor for each axis individually
- use elastic deformation with a probability of 0.3
- use additive brightness augmentation with a probability of 0.3
- increase the aggressiveness of the Gamma augmentation

Large batch size (BL*) - large batch size, for this 2D case, was constrained to 64 for consistency and training.

Metrics

Region-based training mean Dice & HD95 metrics reported, where each 'trainer class' is a combination of one or more different improvement modules listed before. Note that BL* (fixed large batch size) is utilized for all trainer classes so not mentioned in trainer classes but assumed.

Table 1 - Training Set (80:20%, fold 0)

	Dice			**HD95**		
trainer_class	**Enh**	**Whole**	**Core**	**Enh**	**Whole**	**Core**
R	89.36	87.63	85.74	13.966	11.001	11.237
R+BN	90.51	89.02	87.27	11.685	10.841	11.057
R+DA	88.52	86.59	84.68	16.218	10.290	9.946
R+DA+BN	89.62	86.63	84.51	11.231	10.805	11.052
R+DA*+BN	89.61	87.51	85.43	13.408	7.450	7.663
R+DA+BN+BD	89.76	87.29	85.32	12.856	13.858	14.095
R+DA*+BN+BD	90.31	87.67	85.74	11.174	13.767	14.015

Table 2 - Test Set - Trained on entire training set (3360) and predict on test (380)

	Dice			**HD95**		
trainer_class	**Enh**	**Whole**	**Core**	**Enh**	**Whole**	**Core**
R	81.35	61.67	58.90	30.0017	**56.8589**	**57.9130**
R+BN	**83.11**	62.67	59.78	**27.8080**	74.626	74.8734
R+DA	80.85	58.98	56.12	33.8559	68.1004	68.2730
R+DA+BN	82.04	62.02	59.20	27.9922	71.2071	70.4204
R+DA*+BN	80.20	62.91	60.00	34.9345	63.7028	63.8604
R+DA+BN+BD	81.06	64.97	61.93	33.8226	80.7485	80.9358
R+DA*+BN+BD	82.77	64.61	**62.08**	28.8216	75.9308	74.1875
ensemble	82.18	**65.01**	62.06	29.9481	76.9002	77.1101

Table 3 - Ensemble of R+DA+BN+BD, R+DA*+BN, R+DA*+BN+BD (top 3 generally, based on the original study)

	Dice			**HD95**		
	Enh	**Whole**	**Core**	**Enh**	**Whole**	**Core**
Mean	82.18	65.01	62.06	29.948	76.900	77.110
StdDev	27.49	36.33	34.98	98.815	147.171	147.068
Median	89.50	83.33	77.85	1.000	2.828	3.162
25th percentile	79.43	47.37	45.34	0.000	1.000	1.000
75th percentile	100.00	90.11	86.06	2.000	15.493	15.493

Table 4 - Ensemble (same 3 trainer classes) but without pre-processing steps 1 & 2 (240x240 images)

	Dice			**HD95**		
	Enh	**Whole**	**Core**	**Enh**	**Whole**	**Core**
Mean	82.26	65.46	62.58	31.286	76.098	76.515
StdDev	27.30	36.20	34.98	100.281	144.086	143.899
Median	90.15	82.13	78.53	1.000	4.123	5.000
25th percentile	80.19	51.33	48.84	0.000	1.414	1.414
75th percentile	100.00	90.80	86.87	2.236	24.277	25.071

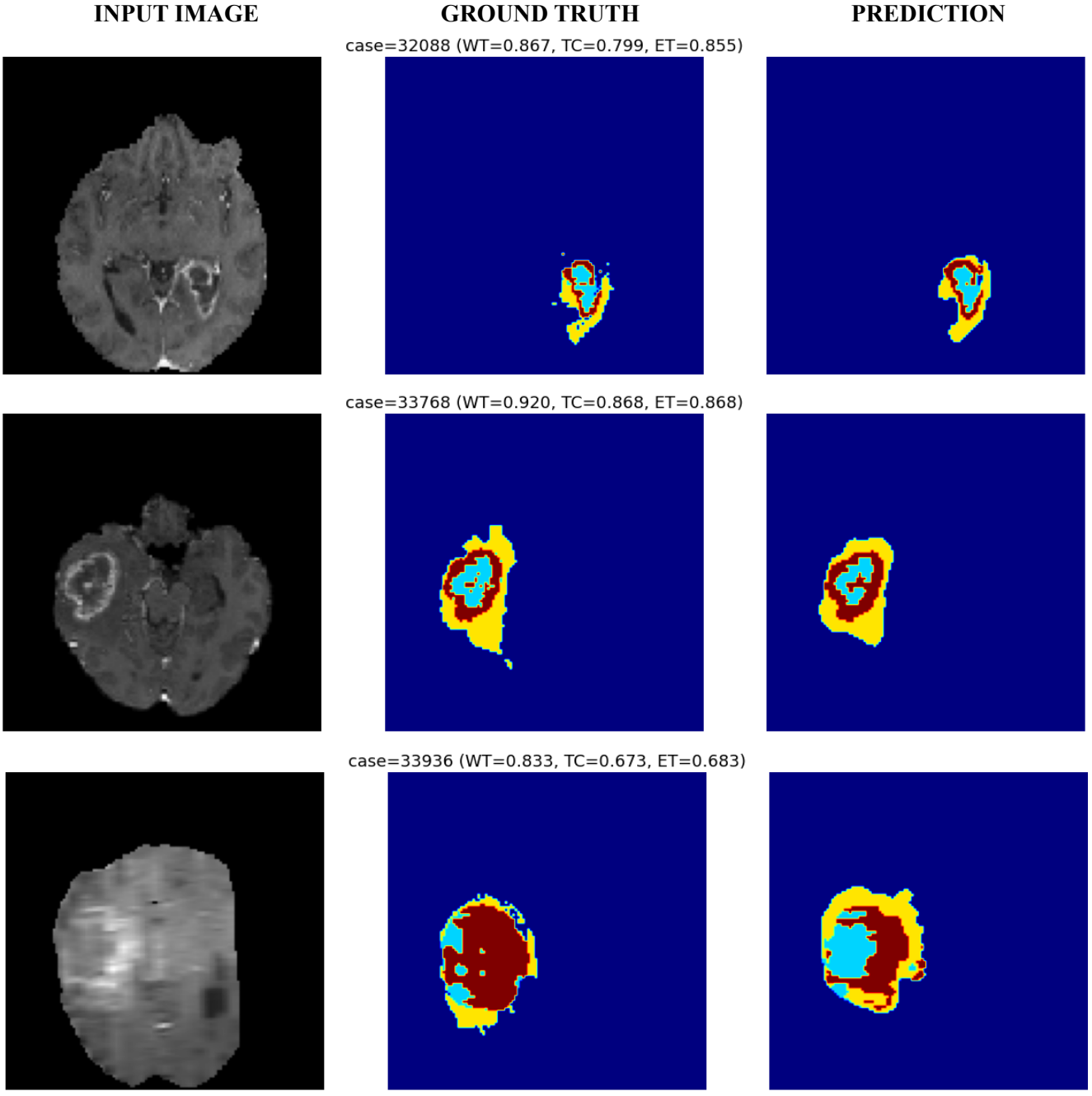
Table 5 - Results for the same ensemble in the original study on the 3D BraTS2020 for comparison

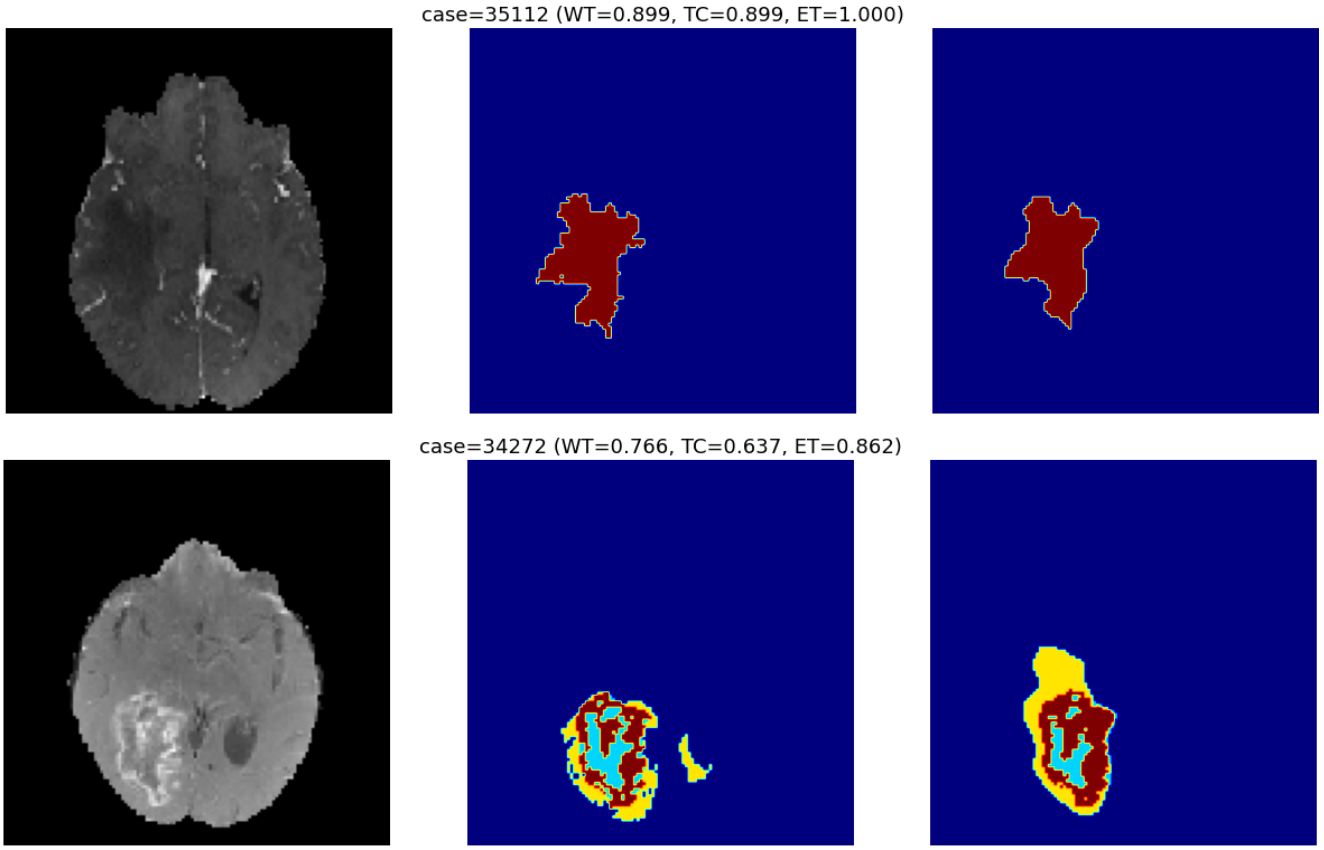
	Dice			HD95		
	Enh	Whole	Core	Enh	Whole	Core
Mean	82.03	88.95	85.06	17.805	8.498	17.337
StdDev	19.71	13.23	24.02	74.886	40.650	69.513
Median	86.27	92.73	92.98	1.414	2.639	2.000
25th percentile	79.30	87.76	88.33	1.000	1.414	1.104
75th percentile	92.25	95.17	96.19	2.236	4.663	3.606

Test Set Predictions

Using final ensembles (best), with case IDs and Dice scores.
('nan' Dice score treated as 1, as per original study evaluation)

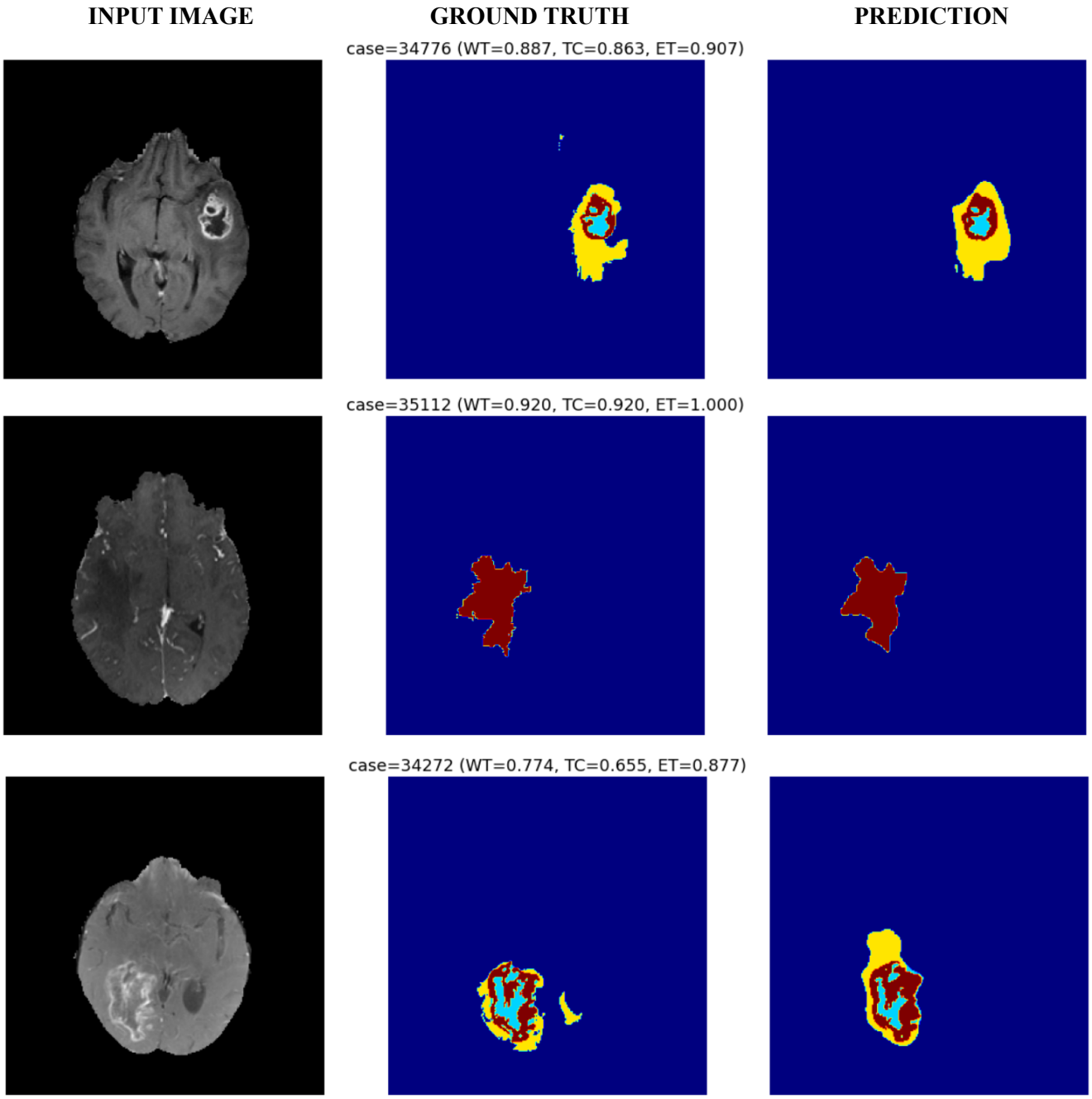
Figure 1 - Pre-processed 128x128 variant final ensemble predictions

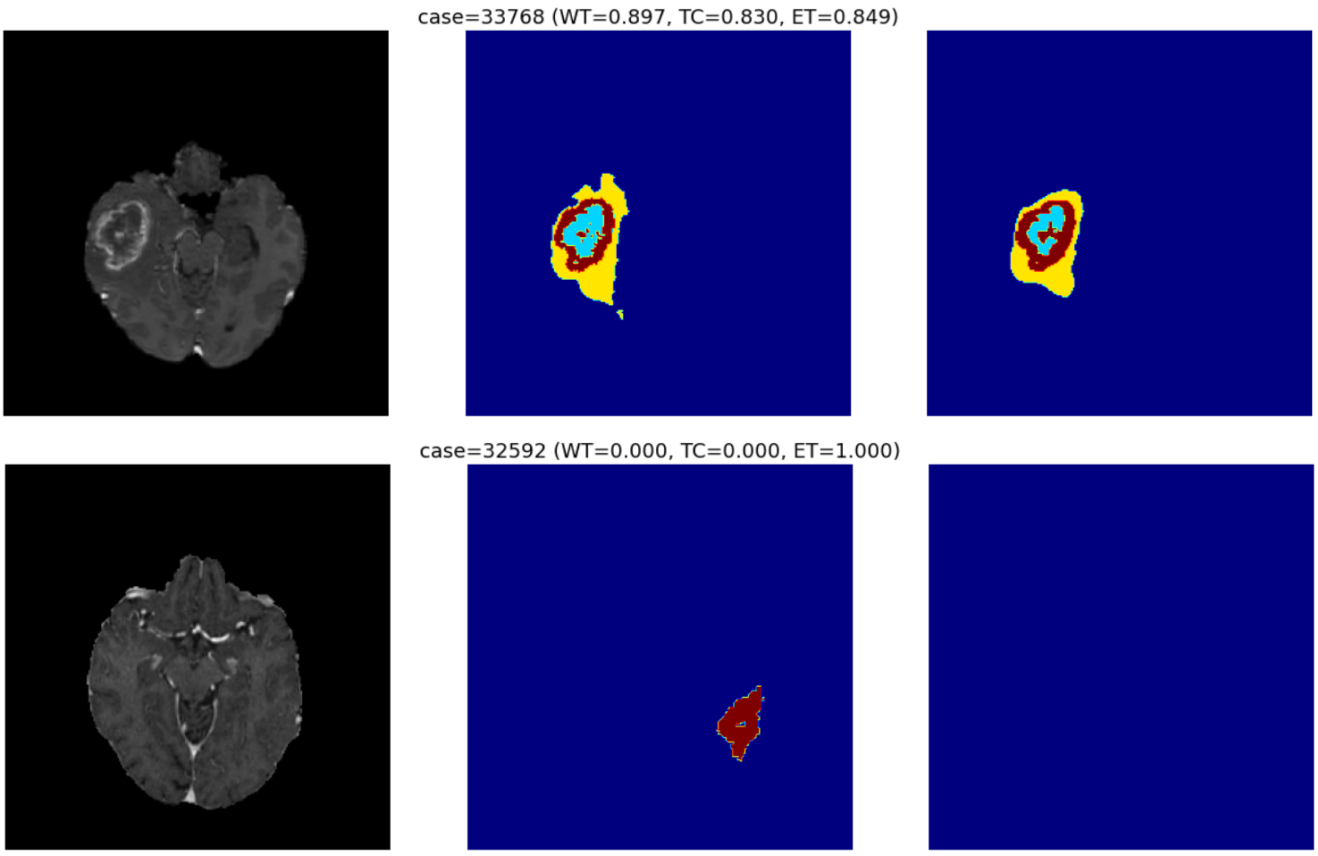



case=35112 (WT=0.899, TC=0.899, ET=1.000)
case=34272 (WT=0.766, TC=0.637, ET=0.862)

==========

Figure 2 - Non-preprocessed 240x240 final ensemble predictions




case=33768 (WT=0.897, TC=0.830, ET=0.849)
case=32592 (WT=0.000, TC=0.000, ET=1.000)

Discussion

We generally see increasing performance across all class labels overall, as more improvement modules are added and trainer classes grow advanced (batch dice, aggressive data augmentation, batch normalization etc.). Increased data augmentation improves model robustness and generalizability to the test set. We also observe that the enhanced tumor (ET) class performs much better compared to the WT (whole tumor) and TC (tumor core) regional class labels for this specific 2D case (Table 3, 4) unlike the 3D BraTS 2020 (Table 5) metrics, with the ET Dice score being almost similar to that of the BraTS2020 (82.18 vs 83.03). The number of samples, 2D vs. 3D context, and absence of multiple MRI modalities are some of the differences that could cause this result. Moreover, HD95 test metrics specifically are observed to be better without any additional improvement modules (R, R+BN best).

Based on our qualitative visual samples for different example slices given, we can understand why our computed metrics are better. The last example in Figure 2, shows why the WT and TC class has a lower score overall compared to other regions - at the borders of the tumor or in certain zones where the tumor is not visible on the provided MR modality, a ground truth label exists but the model does not find any abnormalities and predicts an empty slice yielding a WT and TC class Dice scores of 0 which severely affects the overall mean. However, the absence of the ET grants it a Dice score of 1 and also in other samples where predicting the ET becomes easy when the overall tumor is observed.

Both with and without pre-processing metrics based on the ensemble of the final 3 trainer class predictions (Table 3 vs. Table 4) and also comparing qualitative sample predictions, we observe almost the same results (all quantities mean/std/median). However, training time without preprocessing was almost 2x more (~90s vs. 40s per epoch) so the pre-processing was a great improvement in terms of computational efficiency that enabled us to evaluate all possible trainer class combinations within the current time constraints without losing out on performance.

Challenges:

- Brains and brain tumors are both very heterogeneous structures in that their sizes, shapes, and structures are different for every subject. [1]
- When analyzing solely 2D images, there is a lack of the 3D context required to determine a tumor's presence, type, and location. Also, a single scan can lack features from other modalities, and other parts of a tumor, which also presents inconsistencies in the dataset.[1]
- The brains featured in brain scans present only a small pool of total human brains. This lack of diversity in the dataset is due to the cost and volume limitation of MRI machines and scans. This affects the dataset as well due to not having enough data, and also not having a complete representative sample of the whole population. [1]
- Differences across MRI manufacturers, the varying magnetic field strength of scans, and image artifacts can all contribute to the heterogeneity of the dataset which reduces the required robustness for patient use.

Future work:

Plausible improvements to solve the current problems:

- Encode subject demographics (e.g. age, sex, and other clinical biomarkers) as inputs to better predict better case-by-case or divide the dataset based on these attributes.
- Using 3D data for each subject's individual brain, though computationally expensive, can better encode tumor context as they span across slices.

- More MRI and other structural modalities as additional features can better capture the clear visible shape and structural features of the tumor better to delineate it for segmentation.
- Domain-specific data augmentation (e.g. data augmentation by registration techniques or using intelligent generative models like GANs or stable diffusion).
- Techniques to encode global image feature context better - e.g. transformers [3] which can encode-decode image patches as token sequences capturing temporal constraints between these smaller image sub-sections as well to yield better performance and greater computational efficiency than CNNs.
- Better regularization (increased L1/L2 weight decay for convolutional kernels as well as the inclusion of general [2] or domain-specific regularization terms in the loss function based on the shape or structure of the tumor possibly)
- Fine-tuning to prevent overfitting, along with weight initialization and test different optimizers for faster convergence
- Balancing number of non-tumor images and the rest in the train-set for this specific 2D case could also yield better generalizability and metrics over the test set, as it would be able to better differentiate between the two types of images.

Code Execution & Trained Model Predictions

We provide a modified version of the source nnUNet GitHub repository [4] (some edits were to internal trainer classes), which also has some of our main custom bash and python scripts - for everything from pre-processing, downloading structured & pre-processed data, preparation, training, predictions, evaluation, and visualization.

You can also run predictions on the test set again directly, for saved best models trained for each improvement class on the training set. The README.md attached with instructions should help with the process.
GitHub Link

References

[1] Jungo, Alain, Fabian Balsiger, and Mauricio Reyes. "Analyzing the quality and challenges of uncertainty estimations for brain tumor segmentation." *Frontiers in neuroscience* (2020): 282.
[2] Myronenko, Andriy. "3D MRI brain tumor segmentation using autoencoder regularization." International MICCAI Brainlesion Workshop. Springer, Cham, 2018.
[3] Wang, Wenxuan, et al. "TransBTS: Multimodal brain tumor segmentation using transformer." International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer, Cham, 2021.
[4] Isensee, Fabian, et al. "nnU-Net for brain tumor segmentation." International MICCAI Brainlesion Workshop. Springer, Cham, 2020.