Robust PET/CT Lesion Segmentation on Whole-Body Scans

Adrian Galdran[1,2] ADRIAN.GALDRAN@UPF.EDU
[1] *Universitat Pompeu Fabra, Barcelona, Spain*
[2] *Australian Institute for Machine Learning, Adelaide, Australia*

1. Introduction

The simultaneous acquisition of Computed Tomography (CT) and Positron Emission Tomography (PET) can enhance localization of malignant tumor across the whole body, resulting in early diagnosis and better disease management(Kurli et al., 2005; Froelich and Salavati, 2020). Of particular relevance is the availability of reliable lesion segmentations, which enables better tumor characterization. On the other hand, manual lesion delineation in whole body scans can be extremely costly in terms of expert time, and is not a realistic solution. In this context, machine learning has emerged as an ideal tool to speed up PET/CT scan processing and produce accurate lesion segmentations. However, this is not a straightforward problem, as lesions share hyper-intense appearance with healthy tissue on PET scans, potentially leading to the generation of False Positives that may render a computational system for this task useless.

The AutoPET competition was held in 2022 to deal with this challenging problem[1], with the second edition running in 2023 with an expanded dataset[2]. This document describes our approach to solve the AutoPET-II challenge, going over the main aspects of our proposed solution for robust PET/CT lesion segmentation. Accompanying code has been published [3]; both artifacts (code and report) might be better read in conjunction. It is important to stress that, for the sake of completeness, we report below all aspects of our solution. However, in our opinion, the most critical components in making our system work were 5) a careful data splitting and performance metric choice, which enabled meaningful both meaningful model ensembling and appropriate thresholding levels, and 8) a False Positive removal sub-model that has the potential for improving performance. In addition, it speeds up the prediction phase, as it can be useful for avoiding the costly segmentation of images that contain no lesion.

The remaining of this report is distributed as follows:

- Section 2), **Pre-processing and Data Augmentation**: where we describe the exact manner in which both CT and PET scans are prepared before being supplied to the segmentation model for learning.

- Section 3, **Architecture and Pre-Training**: where we detail the neural network architecture used in this work, and the starting weights from which learning is initialized.

- Section 4, **Loss Function, Model Output and Optimization**: where we motivate the choice of loss function, what is the structure of the output of our network, and how we optimize its parameters.

[1] https://autopet.grand-challenge.org/
[2] https://autopet-ii.grand-challenge.org/
[3] https://github.com/agaldran/autopet2

- Section 6, **Data Splitting and Ensembling**: where we analyze the often understated importance of a correct data split and how should multi-fold ensembling be handled in the particular scenario of binary lesion segmentation.
- Section 5, **From Probabilities to Predictions**: where we discuss the binarization mechanism necessary to turn voxel-wise probabilities into categorical segmentations.
- Section 8), **Performance metrics for model selection**: where we study what is the appropriate metric to evaluate intermediate model results, which determines important aspects of the training like, *e.g.*, early-stopping, and propose a (to the best of our knowledge) new performance metric for the evaluation of voxel-wise probabilities in binary segmentation problems.
- Section 7), **False Positive Removal**: where we describe the use of a secondary system, complementing the segmentation model, that may be beneficial for both reducing false positives and also accelerating inference times.

2. Pre-Processing and Data Augmentation

The data loading and augmentation step in our approach is quite straightforward. We use built-in data loading functionality from the Monai library (Cardoso and et al., 2022): we first crop the PET and CT scans to their foreground (based on the CT scan, discarding voxels with an intensity below the 5% percentile of values found in the training set). Then we adjust volume spacing to a target value close to the median spacing on the training set ($2 \times 2 \times 3$), and clamp intensity values to be between the 5% and 95% percentile of training values. Next, we normalize intensities to 0 mean and unit standard deviation, and for each scan we sample 12 volumetric patches of size $96 \times 96 \times 96$, with a 2/3 probability of sampling with a lesion voxel as center. Finally, we apply some random data augmentation operations on those patches, namely random scaling, zooming, and flipping to alter geometry, and random intensity scaling and shifting. More details can be found in our code repository [4]

3. Architecture and Pre-training

In many medical image analysis competitions the default approach is to opt for the nnUnet (Isensee et al., 2021), a segmentation framework that automatically adjusts its behavior depending on dataset characteristics. Despite its excellent default design choice decisions, we decided to start with another architecture, namely we selected the sliding-window transformer Swin-UNETR architecture (Hatamizadeh et al., 2022). Our choice was motivated by the facts that a) Swin-UNETR holds the state-of-the-art in the popular Medical Decathlon contest, and it is therefore a well-tested solution for volumetric image segmentation. and b) pretrained weights, optimized on a CT segmentation task, are publicly available for several instantiations of this architecture[5]. Specifically, three size variants exist: *tiny*, *small*, and *base*. We did not notice significant performance improvements when using larger variants, so we sticked to the *tiny* configuration, as shown in Fig. 1.

[4] https://github.com/agaldran/autopet2/utils/data_load.py

[5] See https://github.com/Project-MONAI/research-contributions/tree/main/SwinUNETR/BTCV

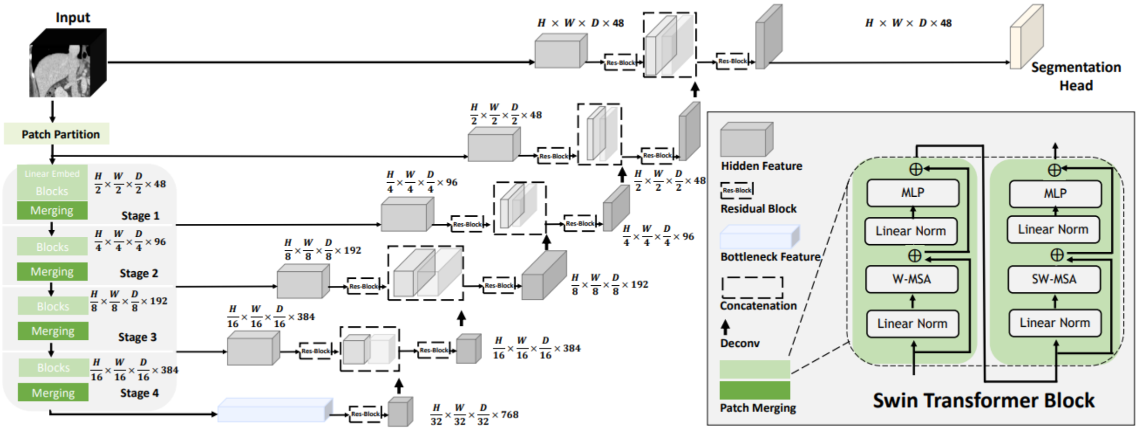
Input
Patch Partition
Linear Embed
Blocks
Merging
Stage 1
Blocks
Merging
Stage 2
Blocks
Merging
Stage 3
Blocks
Merging
Stage 4
$\frac{H}{2}\times\frac{W}{2}\times\frac{D}{2}\times 48$
$\frac{H}{4}\times\frac{W}{4}\times\frac{D}{4}\times 96$
$\frac{H}{8}\times\frac{W}{8}\times\frac{D}{8}\times 192$
$\frac{H}{16}\times\frac{W}{16}\times\frac{D}{16}\times 384$
$\frac{H}{32}\times\frac{W}{32}\times\frac{D}{32}\times 768$
$H\times W\times D\times 48$
Res-Block
Segmentation Head
Hidden Feature
Residual Block
Bottleneck Feature
Concatenation
Deconv
Patch Merging
MLP
Linear Norm
W-MSA
SW-MSA
Swin Transformer Block

Figure 1: Swin-UNETR model architecture, reproduced from the official repository.

4. Loss Function, Model Output, and Optimization

4.1. Loss Function, or the unreasonable robustness of Cross-Entropy

It is known that the Dice loss, despite being useful for maximizing segmentation and ground-truth overlap, tends to prefer more "binary" solutions, *i.e.* it results in overconfident segmentations (Mehrtash and et al., 2020). More recently, we benchmarked the Dice loss against the Cross-Entropy loss for a 2-D binary lesion segmentation task in which we trained models on images that always contained lesions, but then tested them on images that might not show one (Galdran et al., 2023). In that work, we showed that the Dice loss would result in catastrophic over-segmentations and false positives when no lesion was to be segmented, and lead to drastic under-performance. On the other hand, the Cross-Entropy loss would achieve slightly less Dice score on the part of the test set that contained lesions, but would generate much less false positives.

In this competition, we quickly noticed that the same phenomenon took place. Early in our model development we realized that training with Dice loss or variants of it (Dice+CE - default of the nnU-net framework (Isensee et al., 2021), Dice+Focal, etc.) had the effect of creating large volumes of false positives, even if we may sometimes achieve higher dice scores. In contrast, Cross-Entropy minimization led to slightly less overlap, but much better robustness. This is due to the increasing penalty that Cross-Entropy places on voxels that are wrongly predicted with high confidence, a situation that the Dice loss completely disregards. Since challenge metrics take into account the amount of false positives present in submitted solutions, we discarded the Dice loss, even if we could sometimes achieve less overlap on correctly segmented lesions.

4.2. Model Output

As a minor but relevant technical detail, when one trains a binary segmentation model, there is the option of having the network return two values per voxel, which are passed through a softmax layer to produce a "probability distribution" for belonging to the background and to a lesion respectively. This is the most common implementation choice because it is immediate to reuse it for multi-class problem, without any substantial modification. On the other hand, we can also have the network produce a single number, which is passed through a sigmoid operation, and returns the "probability" of belonging to a lesion. Even if performance differences will be small, this decision results in slightly different training dynamics. More importantly, it allows the user to run ROC analysis and select a binary threshold that may be optimal for the dataset at hand, instead of relying directly in the implicit $t = 0.5$ threshold provided by an arg-max operation. We provide further discussion on this aspect of our solution in Section 5.

4.3. Optimization

Non-standard choices in our approach, worth mentioning, are described next. Further details are left to be checked in our code repository.

- In order to find more generalizable optima, we used a Cyclical Learning Rate schedule (Smith, 2017) for training. We have typically found in our experience that this schedule benefits not only convergence but also tends to find more generalizable solutions. We trained our segmentation models during 50 cycles of 5 epochs each, and for classification we trained for 5 cycles of 3 epochs each.

- Nesterov-Adam (Dozat, 2016), a small modification of the Adam optimizer that adds momentum to it. At the start of the project, we found slightly better convergence when using N-Adam instead of conventional Adam, so all our models were trained with it.

5. From Probabilities to Predictions

When training binary segmentation models with a single output in practice, there is no reason why a model that perfectly sorts samples in the test set (AUC=1) needs to attain full accuracy or Dice Score (DSC) when thresholding its probabilities using $t = 0.5$. Even if the model is calibrated, so that its probabilities reflect real accuracy, an interesting theoretical fact is that for a particular scan with a DSC of F, then its optimal threshold is $F/2$ (Lipton et al., 2014). This points towards the need of selecting a meaningful threshold value to map our probabilistic predictions (a real value in the unit interval for each voxel) into categorical predictions (a binary {0,1} value per-voxel), *i.e.* final segmentations.

The standard approach in conventional machine learning is to use a validation set to decide which threshold to apply, and there is little reason not to re-use the same validation set we use during training. However, there is no clear answer regarding how to select a single threshold for an ensemble of several models, as discussed in the next section.

6. Careful Data Splitting and Ensembling

A common technique in many machine learning competitions is to train K-Fold ensembles of models, in which we divide training data into K subsets, and then use K-1 for training a model, the other for validation (*e.g.* early stopping), and then rotate the validation set until we eventually train K models. This is the default strategy in the popular nnU-Net framework, with K=5, and we also use here 5-fold cross-validation and ensembling.

Table 1: Optimal thresholding/performance for each model on its validation set. Given this, *How should we select an optimal thresholding value for their ensemble?*

	Fold 1	**Fold 2**	**Fold 3**	**Fold 4**	**Fold 5**
Opt. Thres./DSC	0.48/67.90	0.40/71.75	0.40/66.11	0.48/71.20	0.43/71.15

However, one needs to be careful when coupling this with a binary classification or segmentation model for which we wish to tune the thresholding hyper-parameter, as described in Section 5. In this case, we must ask ourselves: does each individual model require a different threshold to maximize DSC on their own validation sets? and if the answer is yes, how should we select an optimal thresholding strategy for the ensemble of all those models? Should we take the average threshold? The minimum over all individual thresholds?

Table 1 shows the threshold that maximized validation DSC for each model in our 5-fold training scheme. We can see that there is no clear way of choosing a single value for ensembling their ensemble. What is more important to notice is that, if we have spent all our training data on training the 5 models, there is no validation data left to run another threshold selection analysis for the ensemble! For this reason, we decided to first separate a common test set, that we would then use to find that value.

In addition, it should be noted that the challenge data contained just over a thousand volumes, of which roughly half contained cancerous lesions, and the other half were control cases. Since our model is already seeing negative examples each time we sample a volumetric patch from a non-lesion area, we decided to train our 5-fold segmentation model on the "positive" part of the dataset, which also allowed us to speed up training. With all this, our data split for segmentation is illustrated in Fig. 2. The reason for having a test set of negative examples will be made clear in next section.

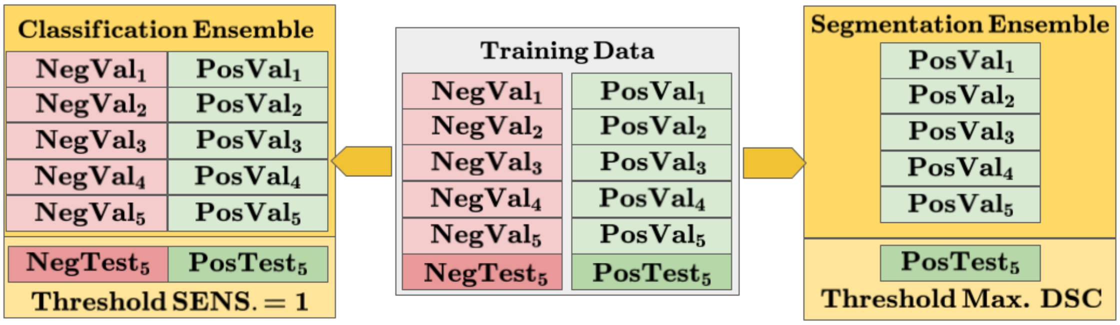

Figure 2: Data Splits of the available training data. Roughly half of the data is positive (contains lesions), and the other half is negative. Our segmentation model is trained on positive data only, and the False Positive Removal model (Section 7) is trained on positives and negatives. A test set is first separated so that we can fix a suitable thresholding value for the ensembles, and later merged into the training data for retraining. A 5-fold cross-validation ensembling strategy is implemented.

A last detail that is worth stressing is that, in order to avoid discarding valuable annotated positive data, once we have trained our 5-fold models and found a meaningful thresholding value for their ensembling on the test set (t=0.375), we merge the test set with the training data, and re-run the 5-fold training, this time without test set. The resulting five models are the core of our submitted solution.

7. False Positive Removal

The AutoPET v2 competition allowed for two submissions on the hidden test set. This set contains both positive and negative scans, and the submitted model is expected to segment nothing on lesion-free samples. As such, the first of our submissions was a 5-fold segmentation model processing every scan in the test set, and hoping that the selected threshold is enough to segment all lesions while avoiding false positives on control scans.

Our second submission is similar in the segmentation branch, but it adds a classification model that processes the scan first. This model attempts to find if the scan corresponds to a healthy individual, and if this is the case, the scan is never submitted for segmentation. Instead, an empty segmentation is returned by the system. This allows us to save on inference time (as the segmentation step is quite heavier than the classification one), and has the potential to improve our final performance.

Our segmentation model is trained to segment volumetric $96 \times 96 \times 96$ patches from a whole-body scan. At inference time, a sliding window approach is taken, but the model only know what a lesion “looks like” locally. In contrast, a classification model (cancer presence vs absence) needs to have the entire body view in order to make a decision, but fitting such a large volume of data into a GPU would be extremely computationally demanding. For this reason, inspired by (Heiliger et al., 2022), we train our classifier on two-dimensional images extracted from PET data, by means of in-plane projection operators. Specifically, we train a Swin transformer for classification (Liu et al., 2021), and use the Maximum Intensity Projection over the X and Y directions. We discard the Z direction due to worse classification performance, and keep the in-plane projections (see Table 2 for average validation performances of each projection). It is reasonable to expect less accuracy on Z, since the amount of slices we are adding up there is considerably larger (“head to feet”, as opposed to “horizontal” body slices). Another interesting aspect of this approach is that we perform data augmentation on the 3D volume before applying the projection, including small three-dimensional rotations and other geometric operations that are richer in this space than after projecting.

Table 2: Performance of 2D classifiers trained on MIP trained on X, Y, Z projections

	Projection X	**Projection Y**	**Projection Z**
Area Under the ROC Curve	92.86 ±1.84	93.01±1.83	83.82±1.89

The training procedure for this classifier are otherwise similar to the segmentation training process, with a five-fold cross validation ensemble trained on negative vs positive data, as illustrated on Fig. 2. The rest of the details can be checked out in our code repository.

It is important to stress that, just as in the segmentation mode, we again need to use an independent test set to find an optimal threshold for classifying a volume into the positive or negative class, but this time we do not seek to maximize F1, or accuracy. Instead we select the highest threshold on the test set[6] that attains **maximum sensitivity**, *i.e.* that picks up all positive samples without creating any False Negative. This is so because at inference time we prefer to avoid declaring a sample as cancer-free when it is not, and would rather send a scan without lesion into the segmentation leg of the system, which can still manage to output an empty segmentation. Restricting ourselves to such a conservative approach (a threshold of t=0.05 is used) results in producing approximately 1 False Positives for each scan that we identity as True Negative and refuse to segment. Due to lack of time, we do not have data on how many of those False Positives actually end up with an empty solution after going through the segmentation model.

8. Performance Metric for Model Selection

During neural network training, it is customary to track model performance in order to decide on hyper-parameters and carry out model selection. Usually we would be interested in monitoring metrics that are correlated to our final use case. In the case of this competition, that would be maximizing the Dice Score while reducing false positives. However, model predictions are not categorical but probabilistic, and attempting to track these metrics would require us so define a threshold during training, which we are unable to do in an optimal manner as discussed above.

We propose to track instead the Area Under the ROC curve (AUROC) for assessing if probabilistic predictions are discriminative enough, no matter of the scale in which they move, since we will be defining our threshold later in the post-processing stage. On the other hand, the AUROC could be easily overwhelmed by the extreme class imbalance typical of volumetric lesion segmentation. Inspired by the Top-K loss function definition (**?**), which mines hardest samples ("most misclassified", or lowest loss value) in order to derive a loss value, we apply a similar principle to performance measurement.

Consider a scan with a total amount of lesion voxels of n. We form an evaluation set $\mathcal{S}$ with all these n voxels, and also with the n background voxels that receive a worst prediction, *i.e.* that are closer to be considered as lesion by the model, discarding all the remaining background voxels. We then compute the AUROC restricted to $\mathcal{S}$, and utilize this metric to perform model selection. A model with a perfect AUROC on $\mathcal{S}$ has a perfect AUROC on the entire volume by definition, and we are able to bypass the class imbalance problem this way.

9. Conclusions

Taking part in the AutoPET v2 challenge was an interesting learning experience. It was one of my first attempts to learn the MONAI library: with all its ups and downs, it is a great tool to have under ones' belt. I would like to thank the organizers for keeping communication channels open and active, for providing a starting baseline solution, and a working docker container that could be submitted to GC out-of-the-box.

[6] hence the need mentioned above for a test set of negative samples also

Acknowledgments

This work was supported by a Marie Sklodowska-Curie Fellowship (No 892297).

References

M. Jorge Cardoso and et al. MONAI: An open-source framework for deep learning in healthcare, November 2022. URL http://arxiv.org/abs/2211.02701. arXiv:2211.02701 [cs].

Timothy Dozat. Incorporating Nesterov Momentum into Adam. In *Proceedings of the 4th International Conference on Learning Representations*, pages 1–4, 2016.

Jerry W. Froelich and Ali Salavati. Artificial Intelligence in PET/CT Is about to Make Whole-Body Tumor Burden Measurements a Clinical Reality. *Radiology*, 294(2):453–454, February 2020. ISSN 0033-8419. doi: 10.1148/radiol.2019192425. URL https://pubs.rsna.org/doi/full/10.1148/radiol.2019192425. Publisher: Radiological Society of North America.

Adrian Galdran, Gustavo Carneiro, and Miguel A. González Ballester. On the Optimal Combination of Cross-Entropy and Soft Dice Losses for Lesion Segmentation with Out-of-Distribution Robustness. In *Diabetic Foot Ulcers Grand Challenge*, Lecture Notes in Computer Science, pages 40–51, 2023. doi: 10.1007/978-3-031-26354-5_4.

Ali Hatamizadeh, Vishwesh Nath, Yucheng Tang, Dong Yang, Holger R. Roth, and Daguang Xu. Swin UNETR: Swin Transformers for Semantic Segmentation of Brain Tumors in MRI Images. In *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*, Lecture Notes in Computer Science, pages 272–284, 2022. doi: 10.1007/978-3-031-08999-2_22.

Lars Heiliger, Zdravko Marinov, Max Hasin, André Ferreira, Jana Fragemann, Kelsey Pomykala, Jacob Murray, David Kersting, Victor Alves, Rainer Stiefelhagen, Jan Egger, and Jens Kleesiek. AutoPET Challenge: Combining nn-Unet with Swin UNETR Augmented by Maximum Intensity Projection Classifier, October 2022. URL http://arxiv.org/abs/2209.01112. arXiv:2209.01112 [cs, eess].

Fabian Isensee, Paul F. Jaeger, Simon A. A. Kohl, Jens Petersen, and Klaus H. Maier-Hein. nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. *Nature Methods*, 18(2):203–211, February 2021. ISSN 1548-7105. doi: 10.1038/s41592-020-01008-z. URL https://www.nature.com/articles/s41592-020-01008-z. Number: 2 Publisher: Nature Publishing Group.

Madhavi Kurli, Shantan Reddy, Lawrence B. Tena, Anna C. Pavlick, and Paul T. Finger. Whole body positron emission tomography/computed tomography staging of metastatic choroidal melanoma. *American Journal of Ophthalmology*, 140(2):193–199, August 2005. ISSN 0002-9394. doi: 10.1016/j.ajo.2005.02.051.

Zachary C. Lipton, Charles Elkan, and Balakrishnan Naryanaswamy. Optimal Thresholding of Classifiers to Maximize F1 Measure. *Machine learning and knowledge discovery in databases : European Conference, ECML PKDD ... : proceedings. ECML*

PKDD (Conference), 8725:225–239, 2014. doi: 10.1007/978-3-662-44851-9_15. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4442797/.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. pages 10012–10022, 2021. URL https://openaccess.thecvf.com/content/ICCV2021/html/Liu_Swin_Transformer_Hierarchical_Vision_Transformer_Using_Shifted_Windows_ICCV_2021_paper.html.

Alireza Mehrtash and et al. Confidence Calibration and Predictive Uncertainty Estimation for Deep Medical Image Segmentation. *IEEE Transactions on Medical Imaging*, 39(12), December 2020. doi: 10.1109/TMI.2020.3006437.

Leslie N. Smith. Cyclical Learning Rates for Training Neural Networks. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 464–472, 2017. doi: 10.1109/WACV.2017.58.