Label merge-and-split: A graph-colouring approach for memory-efficient brain parcellation

Aaron Kujawa[1][0000−0002−1445−6207], Reuben Dorent[2][0000−0002−7530−0644], Sebastien Ourselin[1][0000−0002−5694−5340], and Tom Vercauteren[1][0000−0003−1794−0456]

[1] King's College London, UK
[2] Harvard University, USA

Abstract. Whole brain parcellation requires inferring hundreds of segmentation labels in large image volumes and thus presents significant practical challenges for deep learning approaches. We introduce label merge-and-split, a method that first greatly reduces the effective number of labels required for learning-based whole brain parcellation and then recovers original labels. Using a greedy graph colouring algorithm, our method automatically groups and merges multiple spatially separate labels prior to model training and inference. The merged labels may be semantically unrelated. A deep learning model is trained to predict merged labels. At inference time, original labels are restored using atlas-based influence regions. In our experiments, the proposed approach reduces the number of labels by up to 68% while achieving segmentation accuracy comparable to the baseline method without label merging and splitting. Moreover, model training and inference times as well as GPU memory requirements were reduced significantly. The proposed method can be applied to all semantic segmentation tasks with a large number of spatially separate classes within an atlas-based prior.

Keywords: Image segmentation · Whole brain parcellation · Graph-colouring.

1 Introduction

Whole brain parcellation (WBP) is a multi-class semantic segmentation task that requires the assignment of labels to all brain regions based on distinct anatomical and/or functional features. Although the level of granularity may vary, WBP typically requires dealing with hundreds of distinct classes. For example, the parcellation tools Freesurfer [7] and GIF (Geodesic Information Flows) [3,22] predict up to 105 classes and 158 classes, respectively. While classical tools such as Freesurfer and GIF are robust and accurate, processing times vary between multiple hours to a full day for a single input image.

Deep learning models can greatly reduce inference time, with some methods achieving WBP within 1 minute [11,24,23] or as little as 11 seconds [25]. An important limitation of existing approaches is the high demand for GPU and CPU

memory and computational power. Bottleneck operations are typically those which involve continuous values/probabilities for each label such as multi-label resampling (for image scaling and spatial data augmentation) and label-wise loss calculations. Resources required for these operations scale linearly with the number of labels.

While smaller network architectures and techniques like network compression and model quantization have been employed to partially address this problem [16], the fixed output size of the softmax layer at the end of 3D networks remains a driver of GPU requirements and thus a bottleneck for all deep neural networks.

This has led to mitigation strategies such as patch-based methods which extract smaller sub-volumes from the full image during training and inference, at the cost of losing spatial context [20,19,16,6,28,12,4,25,19]. Alternatively, 2.5D methods reduce memory requirements by producing separate predictions for all 2D slices in axial, coronal and sagittal orientation [11,24,23].

Other existing approaches further reduce memory and computational requirements by merging semantically related label pairs during model training and inference [11,25]. These methods exploit the symmetry of pairs of structures that are present in both brain hemispheres, for example the right and left Pallidum, and merge the corresponding labels into a single label. At inference time, as a post-processing step, the predicted merged label is separated along the mid-sagittal plane to recover the original labels. However, many brain structures do not have left/right pairs (e.g., brain stem). In other cases, the left/right components may be spatially too close to the mid-sagittal plane (e.g., left/right cerebellum white matter), which makes their separation during post-processing error-prone. This limits the reduction in the number of labels to a relatively small fraction. For example, in [11] and [25], a reduction from 95 to 78 labels was achieved by merging 17 label pairs. The process also requires manual specification of symmetric and separate label pairs.

In this work, we propose an automated and scalable method to significantly reduce the number of labels while allowing for accurate label splitting during post-processing. The method includes the following steps: Firstly, we construct an undirected graph where vertices correspond to brain structures and edges to discretised spatial distance priors. Using a graph colouring algorithm [18], our approach automatically groups and merges multiple spatially separate labels of similar volume. The spatial separation is required to allow for label splitting during inference, while the constraint for similar volumes prevents high class imbalance. Secondly, a memory-efficient 3D U-Net approach is trained for automated parcellation and prediction of merged labels. Thirdly, we employ a strategy to split the predicted merged segmentation labels using atlas-based influence regions. Finally, experiments on three publicly available datasets (Mindboggle101 [15], AOMIC [26], and IXI [13]) demonstrate the effectiveness of the proposed merge-and-split approach. The number of labels is reduced by up to 68%, resulting in a great reduction of training and inference time (by 43-49%) and GPU requirements during training (by 50%) and inference (by 21%) while achieving

a segmentation quality comparable to the memory-heavy model trained without label merging and splitting.

2   Methods

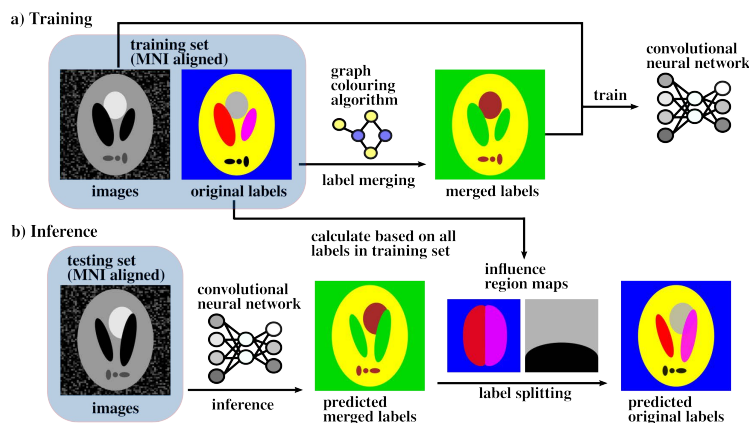An overview of the proposed method is shown in Fig. 1.

Fig. 1. Method overview illustrated on a toy example. a) Training. Original labels in MNI space are automatically grouped using a greedy graph colouring algorithm and merged. Toy example: {blue, red, purple} → green; {gray, black} → brown; and {yellow} → yellow. The CNN is trained on merged labels. b) Inference. The CNN predicts merged labels. In a post-processing step, original labels are recovered using one influence map for every set of merged labels.

Preprocessing. Brain images are aligned to the MNI152 T1 template (ICBM 2009a Nonlinear Asymmetric) [8] with affine registration using Advanced Normalization Tools (ANTs) [1] and resampled to a voxel spacing of $1 \times 1 \times 1$ mm and matrix dimension $193 \times 229 \times 193$.

Merging of labels. Label merging is performed under two constraints. **1)** The distance between to-be-merged labels in the training set must not be smaller than an empirically selected threshold to facilitate label-splitting at inference time. **2)** To-be-merged labels must not be too different in volume to avoid reduced weighting of small labels in the loss function. This is achieved by setting an empirical threshold on the ratio of average label volumes in the training set. Our method illustrated in Fig. 2 is comprised of the steps described below.

1. To account for variability in brain morphology, we consider the support of labels in the MNI space by computing a simple prior map S similar to a
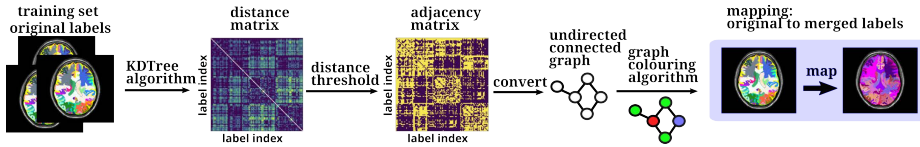
Fig. 2. Label merging. The distance between original labels is calculated from (MNI aligned) training set label volumes. A distance threshold and is applied to obtain an adjacency matrix (yellow: adjacent, blue: non-adjacent). Similarly, an average label volume ratio threshold is applied (not shown here). The adjacency matrix corresponds to an undirected connected graph, to which a greedy graph colouring algorithm is applied. Labels of the same colour are grouped, forming a new merged label.

probabilistic atlas. For this, we sum the one-hot encoded training label volumes in the MNI space. Formally, let $X^i : \Omega \subset \mathbb{R}^3 \to \mathbb{R}$ denote the ith image of the set of N_{tr} training images, with Ω being the spatial domain, and let $Y^i : \Omega \subset \mathbb{R}^3 \to \{0, \ldots, N_{\mathrm{ol}} - 1\}$ be its corresponding ground-truth segmentation with N_{ol} original labels. Denoting spatial coordinates as $\mathbf{v} = (i, j, k)$ and label indices as $l \in \{0, \ldots, N_{\mathrm{ol}} - 1\}$, the original label support map S is given by:

$$S_{\mathbf{v},l} = \sum_{i=0}^{N_{\mathrm{tr}}-1} \tilde{Y}_{\mathbf{v},l}^i \qquad \text{where } \tilde{Y}_{\mathbf{v},l} = \begin{cases} 1 & \text{if } Y_{\mathbf{v}} = l \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

2. We calculate the minimum distances between original labels as observed in the label support map S. This information is encoded in a distance matrix D whose entries are the minimum Euclidean distance between any two voxels of original label l_1 and l_2 taken from any two training volumes in MNI space:

$$D_{l_1 l_2} = \min\{d(\mathbf{v}_1, \mathbf{v}_2) | \mathbf{v}_1 \in L_1, \mathbf{v}_2 \in L_2\} \tag{2}$$

where L_1 and L_2 are the sets of spatial coordinates which have non-zero support for l_1 and l_2 in the label support matrix: $L_1 = \{\mathbf{v} | S_{\mathbf{v},l_1} > 0\}$, $L_2 = \{\mathbf{v} | S_{\mathbf{v},l_1} > 0\}$. To reduce processing time for (2), it is sufficient to consider only the border voxels of L_1 and L_2. These were extracted using the `find_boundaries` function of the Python library scikit-image [29]. Then, a KDTree algorithm [17] is used on each pair of original labels and the respective boundary voxels to find the minimum distance between any two voxels of different label. Note that the matrix is symmetric, therefore calculation of the upper triangular part is sufficient.

3. The average volume ratio matrix $V_{l_1 l_2}$ between two labels l_1 and l_2 is calculated by dividing the larger average label volume by the smaller average label volume.

4. We convert the distance matrix and volume ratio matrix into an adjacency matrix A, using the manually selected hyperparameters δ_{D} (distance threshold) and δ_{V} (volume ratio threshold):

$$A_{l_1 l_2} = \begin{cases} 0 & \text{if } D_{l_1 l_2} > \delta_{\mathrm{D}} \text{ and } V_{l_1 l_2} < \delta_{\mathrm{V}} \\ 1 & \text{otherwise} \end{cases} \tag{3}$$

From the adjacency matrix A, an undirected graph is constructed using Python's NetworkX library [10]. Each vertex of the graph represents an original label and only adjacent vertices are connected. Lower δ_D and higher δ_V thresholds will result in a smaller number of labels and thus greater memory savings. Conversely, higher δ_D facilitates the downstream splitting of merged labels and a smaller δ_V prevents excessive class imbalance.

5. Next, we cast the identification of merged label groups as a graph colouring problem. Graph colouring assigns colours (in our case corresponding to the merged labels) to the vertices of a graph such that no two adjacent vertices are assigned the same colour. This property, in combination with (3), ensures that only original labels with a minimum distance above δ_D and an average volume ratio below δ_V are grouped within a given merged label. Minimising the number of merged labels, i.e. colours, will lead to minimising memory requirements. Finding the smallest number of colours to achieve this is an NP-hard problem [9] but good practical solutions exist. We use a greedy graph colouring method which considers the vertices of the graph in sequence and assigns each vertex its first available colour in linear time. Specifically, NetworkX's greedy graph colouring algorithm `greedy_color` is applied to the graph, using the "`smallest_last`" strategy [18]. Finally, each merged group is assigned a unique label, $\mathrm{ml} \in \{0..N_{\mathrm{ml}} - 1\}$, where N_{ml} is the number of merged labels.

Convolutional neural network. Our method is compatible with any neural network backbone. Here, we rely on a patch-based 3D U-Net which acts as an established baseline. Our implementation is based on the `DynUNet` class in the MONAI library [2], itself built on top of PyTorch [21]. The network encoder and decoder paths contain 5 levels in addition to the bottleneck level. Training policies were adapted from nnU-Net [14], using random patches of size $192 \times 192 \times 128$ and a batch size of 2. All models were trained for 1000 epochs where 1 epoch was defined as iteration over 250 mini-batches. SGD with Nesterov momentum ($\mu = 0.99$) and an initial learning rate of 0.01 were used. The poly learning rate policy was employed to decrease the learning rate $(1 - \frac{\mathrm{epoch}}{\mathrm{epoch}_{\mathrm{max}}})^{0.9}$. The loss function is the sum of cross-entropy and Dice loss. In addition, deep supervision was applied in the decoder to all but the two lowest resolutions.

Splitting of labels. To split the merged labels predicted by the CNN, we use influence region maps created from the training set label volumes. Our splitting method illustrated in Fig. 3 is comprised of the steps described hereafter.

1. From the original training samples, a *fuzzy prior* $\hat{S}$ is created by normalizing the label support map S from (1): $\hat{S} = \frac{1}{N_{\mathrm{tr}}} S$

2. Next, the fuzzy prior is "fudged" with an Euclidian Distance Transform (EDT) to replace entries of zero likelihood with a likelihood that reflects the spatial distance of each voxel from voxels of non-zero likelihood in $\hat{S}$. The fudged

training set — fuzzy prior — fudged fuzzy prior — influence regions

label counts per voxel — Euclidian Distance Transform — for each merged label find original label with maximum likelihood — one map for each set of merged labelsl

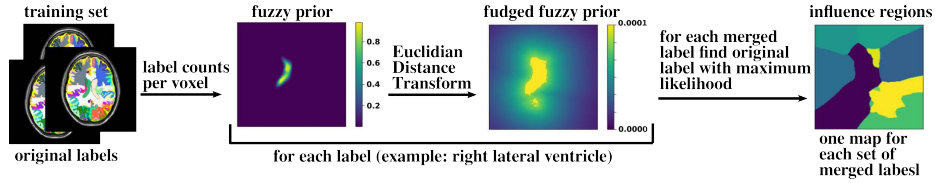original labels — for each label (example: right lateral ventricle)

Fig. 3. Label splitting. A fuzzy prior is created from all (MNI aligned) training set label volumes. Subsequently, a Euclidean Distance Transform is applied for each label. One influence region map is created for each merged label. In this figure, the image on the right is the influence region map for the merged label that was merged from the right lateral ventricle label and 6 other original labels.

fuzzy prior is defined as:

$$\tilde{S}_{\mathbf{v},l} = \begin{cases} \hat{S}_{\mathbf{v},l} & \text{if } \hat{S}_{\mathbf{v},l} > 0 \\ \frac{1}{N_{tr}} \exp(-E_{\mathbf{v},l}) & \text{otherwise} \end{cases} \tag{4}$$

where $E_l = \text{edt}(S_l)$ is the output of the Euclidean Distance Transform. We use SciPy's function `distance_transform_edt` to calculate the EDT [27].

3. Using the fudged prior, N_{ml} influence region maps I_m can be created. An influence region map of a merged label $m \in \{0..N_{\text{ml}} - 1\}$ is created by assigning to each voxel the original label with the highest likelihood:

$$I_{\mathbf{v},m} = \underset{l \in M_m}{\operatorname{argmax}} \tilde{S}_{\mathbf{v},l} \tag{5}$$

where M_m is the set of original labels contained in merged label m.

4. Finally, the merged labels predicted by the CNN can be split based on the influence maps. If the network predicts a merged label volume $\tilde{N}$, each voxel's label is replaced with the original label found in the corresponding influence region map. Thus, the prediction after label splitting is given by:

$$\tilde{Y} = \sum_{m=0}^{N_{\text{ml}}-1} \hat{N} I_m \qquad \text{where } \hat{N}_{\mathbf{v},m} = \begin{cases} 1 & \text{if } \tilde{N}_{\mathbf{v}} = m \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

where $\hat{N}$ corresponds to $\tilde{N}$ in one-hot encoded format.

3   Evaluation methodology

Data sets. We evaluate the proposed method on T1-weighted images of three publicly available datasets. **1)** Mindboggle101 dataset which contains 101 manually labelled images from various publicly available sub-datasets [15]. Two of these sub-datasets (NKI-RS-22 and NKI-TRT-20) were combined into a test set of 42 images (referred to as Mindboggle42). The remaining sub-datasets with 59 annotated images (referred to as Mindboggle59) were used as a multi-atlas for the

classical GIF algorithm [3] to create pseudo-ground-truth segmentations for the AOMIC and IXI datasets. **2)** AOMIC PIOP2 dataset which contains 226 images from healthy participants (age: 18-25 years) [26] with GIF pseudo-ground-truth labels. **3)** IXI dataset which contains 581 images from healthy participants (age: 20-86 years) [13] with GIF pseudo-ground-truth labels.

Model training and testing. The AOMIC and IXI datasets were randomly split into three sets for model training ($N = 144$, $N = 371$), validation ($N = 36$, $N = 93$), and testing ($N = 46$, $N = 117$). Mindboggle42 was used for testing only. Mindboggle59 was not directly used for training so as to keep Mindboggle42 as a more independent test set. To perform two independent merging and training assessments, we used the AOMIC and IXI training datasets separately. Following the steps in Section 2, we obtained models referred to as $AOMIC_{merged}$ and IXI_{merged}. Both models are evaluated on the AOMIC, IXI, and Mindboggle42 testing sets. Preprocessing, label merging, and label splitting were performed on an Intel Core i9 CPU (8 cores at 2.3GHz). CNN training and inference were performed on 2 NVIDIA A100 GPUs with 40GB VRAM each.

Baseline and evaluation metric. As comparison, two baseline models were trained with the original labels, i.e. without label merging and splitting. These models are referred to as $AOMIC_{orig}$ and IXI_{orig}. The training images and the other stages of the training and inference pipeline were identical. For evaluation, the Dice Similarity Coefficient (DSC) is applied for each label [5]. Relative volume error and Hausdorff distance are reported in the supplementary material.

4   Results

The effect of hyperparameters δ_D and δ_V on the effective number of labels determined by the graph coloring algorithm is shown in Fig. A of the supplementary material. Values of $\delta_D = 10$ and $\delta_V = 3.5$ were chosen for all experiments to achieve a substantial reduction of labels while ensuring spatial separation between merged labels and avoiding high class imbalance. The latter threshold was introduced after our initial experiments showed that merging labels with a large average volume ratio resulted in reduced segmentation accuracy of small structures (supplementary material Fig. B). Table 1 compares the resources needed for model training and inference with and without label merge-and-split. The number of labels was reduced by a factor of at least three for both training sets. GPU memory required during training was halved and reduced by 21% during inference. Epoch times during training were reduced by more than 43% and inference time was halved. For label merging prior to model training additional pre-processing times need to be considered: calculation of the fuzzy label prior ($<$1s per training image), average volumes ($\approx$1s per training image), adjacency matrix (12min in total), and label merging before training ($\approx$2s per image) leads to a pre-processing time of less than 1h for both training sets. At inference, label

splitting takes less than 1s per image (without focus on code optimization). Note that these additional processing times are small compared to the time required for network training/inference. The effect of the merge-and-split approach on

Table 1. GPU memory requirements and processing times for training and inference and mean Dice Similarity Coefficients (DSC) according to Fig. 4

model	num. of labels	mem. train [GiB]	mem. infer [GiB]	epoch time train [s]	infer time [s]	mean DSC [%] test set		
						AOMIC	IXI	MB42
$AOMIC_{orig}$	108	32.1	3.53	345	140	87.2	82.7	73.1
$AOMIC_{merged}$	34	15.7	2.77	185	70	87.4	82.4	74.0
IXI_{orig}	108	31.9	3.53	308	101	83.8	87.2	73.4
IXI_{merged}	36	16.1	2.79	175	51	85.8	87.5	73.5

the accuracy of predictions is shown in Fig. 4. The average DSC of the label
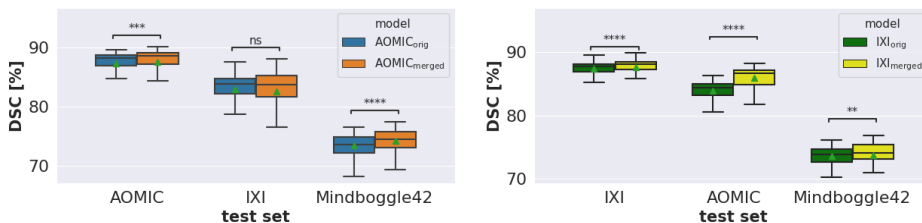

Fig. 4. Effect of the label merge-and-split approach on the quality of predictions. Dice Similarity Coefficients (DSC) are calculated with respect to the GIF pseudo-groundtruth (AOMIC and IXI testing sets) or manual groundtruth (Mindboggle42). Each data point is the average DSC of all labels. Boxplots with DSCs of all individual labels are included in the supplementary material. Green triangles represent the mean. Asterisks indicate statistical significance according to a Wilcoxon test (paired samples) with Bonferroni correction where *, **, ***, and **** indicate p-values below 0.05, 0.01, 0.001, 0.0001, respectively, and ns (not significant) p-values above 0.05.

merge-and-split models is comparable to the baseline models. In 5 of 6 comparisons the merge-and-split models even achieve statistically higher DSCs than the corresponding baseline models.

5   Discussion and conclusion

This study shows that label merge-and-split maintains WBP segmentation performance while greatly reducing memory requirements and processing times. In contrast to previous label merging strategies that focus on merging symmetric

label pairs we show that larger numbers of semantically unrelated labels can be merged and split without reducing segmentation quality. While the results presented here were obtained with the popular U-Net architecture, the method can in principle be combined with any segmentation approach to reduce the effective number of labels.

A limitation of the proposed approach is the reliance on affine registration to a common space. Future work could focus on adapting label merge-and-split to work without affine registration. For example, the distance matrix in (2) for label merging can be constructed from individual distance matrices for each training case. Label splitting could be achieved by assigning original labels to connected components based on their volume and orientation with respect to each other.

Furthermore, for applications other than WBP, the hyperparameters δ_D and δ_V might have to be optimized depending on the required number of labels. Additional constraints can be introduced in the adjacency matrix/graph to control which labels can be merged. For example, for combined WBP and lesion segmentation an additional lesion label may be introduced whose graph node is connected to all other nodes and therefore not merged with any other label.

References

1. Avants, B.B., Tustison, N., Song, G., et al.: Advanced normalization tools (ANTs). Insight Journal **2**(365), 1–35 (2009)
2. Cardoso, M.J., Li, W., Brown, R., Ma, N., Kerfoot, E., Wang, Y., Murrey, B., Myronenko, A., Zhao, C., Yang, D., et al.: Monai: An open-source framework for deep learning in healthcare. arXiv preprint arXiv:2211.02701 (2022)
3. Cardoso, M.J., Modat, M., Wolz, R., Melbourne, A., Cash, D., Rueckert, D., Ourselin, S.: Geodesic information flows: spatially-variant graphs and their application to segmentation and fusion. IEEE transactions on medical imaging **34**(9), 1976–1988 (2015)
4. Coupé, P., Mansencal, B., Clément, M., Giraud, R., de Senneville, B.D., Ta, V.T., Lepetit, V., Manjon, J.V.: AssemblyNet: a novel deep decision-making process for whole brain MRI segmentation. In: Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, October 13–17, 2019, Proceedings, Part III 22. pp. 466–474. Springer (2019)
5. Dice, L.R.: Measures of the amount of ecologic association between species. Ecology **26**(3), 297–302 (1945)
6. Dolz, J., Desrosiers, C., Ayed, I.B.: 3D fully convolutional networks for subcortical segmentation in MRI: A large-scale study. NeuroImage **170**, 456–470 (2018)
7. Fischl, B., Salat, D.H., Busa, E., Albert, M., Dieterich, M., Haselgrove, C., Van Der Kouwe, A., Killiany, R., Kennedy, D., Klaveness, S., et al.: Whole brain segmentation: automated labeling of neuroanatomical structures in the human brain. Neuron **33**(3), 341–355 (2002)

8. Fonov, V., Evans, A.C., Botteron, K., Almli, C.R., McKinstry, R.C., Collins, D.L., Group, B.D.C., et al.: Unbiased average age-appropriate atlases for pediatric studies. Neuroimage **54**(1), 313–327 (2011)
9. Garey, M.R., Johnson, D.S., Stockmeyer, L.: Some simplified NP-complete problems. In: Proceedings of the sixth annual ACM symposium on Theory of computing. pp. 47–63 (1974)
10. Hagberg, A., Swart, P., S Chult, D.: Exploring network structure, dynamics, and function using NetworkX. Tech. rep., Los Alamos National Lab.(LANL), Los Alamos, NM (United States) (2008)
11. Henschel, L., Conjeti, S., Estrada, S., Diers, K., Fischl, B., Reuter, M.: Fastsurfer-a fast and accurate deep learning based neuroimaging pipeline. NeuroImage **219**, 117012 (2020)
12. Huo, Y., Xu, Z., Xiong, Y., Aboud, K., Parvathaneni, P., Bao, S., Bermudez, C., Resnick, S.M., Cutting, L.E., Landman, B.A.: 3D whole brain segmentation using spatially localized atlas network tiles. NeuroImage **194**, 105–119 (2019)
13. Imperial College London: IXI Dataset. https://brain-development.org/ixi-dataset, accessed: 2023-03-05
14. Isensee, F., Jaeger, P.F., Kohl, S.A., Petersen, J., Maier-Hein, K.H.: nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. Nature methods **18**(2), 203–211 (2021)
15. Klein, A., Tourville, J.: 101 labeled brain images and a consistent human cortical labeling protocol. Frontiers in neuroscience **6**, 171 (2012)
16. Li, W., Wang, G., Fidon, L., Ourselin, S., Cardoso, M.J., Vercauteren, T.: On the compactness, efficiency, and representation of 3D convolutional networks: brain parcellation as a pretext task. In: Information Processing in Medical Imaging: 25th International Conference, IPMI 2017, Boone, NC, USA, June 25-30, 2017, Proceedings 25. pp. 348–360. Springer (2017)
17. Maneewongvatana, S., Mount, D.M.: Analysis of approximate nearest neighbor searching with clustered point sets. arXiv preprint cs/9901013 (1999)
18. Matula, D.W., Beck, L.L.: Smallest-last ordering and clustering and graph coloring algorithms. Journal of the ACM (JACM) **30**(3), 417–427 (1983)
19. Mehta, R., Majumdar, A., Sivaswamy, J.: BrainSegNet: a convolutional neural network architecture for automated segmentation of human brain structures. Journal of Medical Imaging **4**(2), 024003–024003 (2017)
20. Moeskops, P., Viergever, M.A., Mendrik, A.M., De Vries, L.S., Benders, M.J., Išgum, I.: Automatic segmentation of MR brain images with a convolutional neural network. IEEE transactions on medical imaging **35**(5), 1252–1261 (2016)
21. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems **32** (2019)
22. Prados, F., Cardoso, M.J., Burgos, N., Wheeler-Kingshott, C., Ourselin, S., Angela, C., Gandini, M., Ourselin, S.: NiftyWeb: web based platform for image processing on the cloud. In: 24th scientific meeting and exhibition of the international society for magnetic resonance in medicine (ISMRM). pp. 7–13 (2016)
23. Roy, A.G., Conjeti, S., Navab, N., Wachinger, C., Initiative, A.D.N., et al.: Quicknat: A fully convolutional network for quick and accurate segmentation of neuroanatomy. NeuroImage **186**, 713–727 (2019)
24. Roy, A.G., Conjeti, S., Sheet, D., Katouzian, A., Navab, N., Wachinger, C.: Error corrective boosting for learning fully convolutional networks with limited data. In:

Medical Image Computing and Computer Assisted Intervention- MICCAI 2017: 20th International Conference, Quebec City, QC, Canada, September 11-13, 2017, Proceedings, Part III 20. pp. 231–239. Springer (2017)

25. Roy, S., Kügler, D., Reuter, M.: Are 2.5 D approaches superior to 3D deep networks in whole brain segmentation? In: International Conference on Medical Imaging with Deep Learning. pp. 988–1004. PMLR (2022)

26. Snoek, L., van der Miesen, M.M., Beemsterboer, T., Van Der Leij, A., Eigenhuis, A., Steven Scholte, H.: The Amsterdam Open MRI Collection, a set of multimodal MRI datasets for individual difference analyses. Scientific data **8**(1), 85 (2021)

27. Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Courna-peau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., et al.: Scipy 1.0: fundamental algorithms for scientific computing in python. Nature methods **17**(3), 261–272 (2020)

28. Wachinger, C., Reuter, M., Klein, T.: DeepNAT: Deep convolutional neural net-work for segmenting neuroanatomy. NeuroImage **170**, 434–445 (2018)

29. Van der Walt, S., Schönberger, J.L., Nunez-Iglesias, J., Boulogne, F., Warner, J.D., Yager, N., Gouillart, E., Yu, T.: scikit-image: image processing in python. PeerJ **2**, e453 (2014)