3D Capsule Networks for Brain MRI Segmentation

Authors

Arman Avesta, MD,[1,2,3] Yongfeng Hui, BS, MPH,[2,3] Mariam Aboian, MD, PhD,[1] James Duncan, PhD,[1,4,5] Harlan M. Krumholz, MD, MS,[3,6] Sanjay Aneja, MD.[2,3,4]

[1] Department of Radiology and Biomedical Imaging, Yale School of Medicine, New Haven, CT 06510
[2] Department of Therapeutic Radiology, Yale School of Medicine, New Haven, CT 06510
[3] Center for Outcomes Research and Evaluation, Yale School of Medicine, New Haven, CT 06510
[4] Department of Statistics and Data Science, Yale University, New Haven, CT 06510
[5] Department of Biomedical Engineering, Yale University, New Haven, CT 06510
[6] Division of Cardiovascular Medicine, Yale School of Medicine, New Haven, CT 06510

Abstract

Background: In patients undergoing radiotherapy or neurosurgery, neuroanatomical segmentation is a critical aid that improves outcomes. Current auto-segmentation methods are limited because they cannot generalize to brain images with features not represented in the training data.

OBJECTIVE: To develop and validate 3D capsule networks (CapsNets) to segment brain images with features not represented in the training data.

METHODS: We used 3430 brain MRIs acquired in a multi-institutional study. We compared our CapsNets with U-Nets, the current standard, based on the accuracy in segmenting various brain structures and those with spatial features not represented in the training data. We also assessed performance when the models are trained using limited data, memory requirements, and computation times.

RESULTS: 3D CapsNets segmented the third ventricle, thalamus, and hippocampus with Dice scores of 94%, 94%, and 91%, respectively. 3D CapsNets outperformed 3D U-Nets in segmenting brain images with features that were not represented in the training data, with Dice scores more than 30% higher (P-values < 0.01). 3D CapsNets had 93% fewer trainable parameters than 3D U-Nets. The 3D CapsNets were 25% faster to train compared with U-Nets. The two models were equally fast during testing.

CONCLUSION: 3D CapsNets segment brain structures with high accuracy, outperform U-Nets in segmenting brain images with features that were not represented during training, and are more efficient compared with U-Nets, achieving similar results while their size is an order of magnitude smaller.

Introduction

In patients undergoing radiotherapy or neurosurgery, neuroanatomical segmentation is a critical component of the clinical workflow.[1–3] Manual segmentation on diagnostic images is difficult because it requires radiologist-level expertise and often cannot be completed in a timely fashion for clinical use. Additionally, manual segmentation is prone to inter- and intra-operator variability, limiting generalizability.[4] Although machine learning based auto-segmentation methods have been developed, current methods fail to properly segment anatomy that is not well represented in the training data. Given the variety of ways in which brain anatomy can change with various pathologic conditions, having training data that captures all potential anatomical variations is near impossible.[4] There is an increasing need for auto-segmentation methods which can generalize beyond the spatial features that are present in the training data.

Auto-segmentation using capsule networks (CapsNets) is a potential solution to this problem.[6–8] CapsNets, initially developed by Sabour et al,[6] have properties that enable them to generalize beyond the narrow features present in the training data.[7] In addition to learning representative features of an image, CapsNets also encode spatial information (such as rotation, size, and thickness), suggesting that they can generalize beyond the images represented in the training data. Presumably, if an anatomical structure rotates, changes in size, or undergoes other spatial changes, the capsule encoding that structure can still recognize it while encoding the changed spatial features.[7]

Two-dimensional (2D) CapsNets have shown success in segmenting lungs on two-dimensional (2D) computed tomography (CT) slices, and muscle and fat tissues on 2D magnetic resonance imaging (MRI) slices,[8] but have shown less impressive results in segmenting brain MRIs.[9] We hypothesized that a 3D CapNet method would improve neuroanatomical segmentation by using the information in the entire 3D image volume as opposed to using the information in one 2D slice only.

Accordingly, we developed and validated 3D CapsNets for volumetric neuroanatomical segmentation using a multi-institutional dataset of more than 3,000 brain MRIs . We compared the utility of 3D CapsNets with other standard deep-learning-based segmentation methods, across different neuroanatomic structures with varying levels of segmentation difficulty. We also evaluated the performance of the models when the test data had features that were not represented in the training data, the performance of the models with limited training data, computations times, and memory requirements to train and deploy the models.

Methods

Dataset

This study was approved by the Institutional Review Board of Yale School of Medicine (IRB number 2000027592). The dataset used for this study included 3,430 T1-weighted brain MRI images, belonging to 841 patients from 19 institutions enrolled in Alzheimer's Disease Neuroimaging Initiative (ADNI) study.[10] The inclusion criteria of ADNI are already published.[11] The participants in this multi-institutional study range from normal to mild cognitive impairment to Alzheimer's dementia. On average, each patient underwent four MRI acquisitions. Details of MRI acquisition parameters are provided in Appendix 1. We

acquired the anonymized MRIs of the patients enrolled in ADNI through Image and Data Archive, which is a data-sharing platform.[10] We randomly split the patients into training (3,199 MRI volumes), validation (117 MRI volumes), and test (114 MRI volumes) sets. Patient demographics are provided in Table 1.

Anatomic Segmentations

Three neuroanatomic structures were chosen for our analysis including third ventricle, thalamus, and hippocampus. These structures were chosen to represent neuroanatomic structures with varying degrees of segmentation difficulty. Segmentations for training and testing were obtained using FreeSurfer, which is a segmentation software with expert-level performance for non-distorted brain images (including in patients with mild cognitive impairment or Alzheimer's dementia).[12–14] To ensure that segmentations were free from error, 120 randomly-selected MRIs from the training set as well as all 114 MRIs in the test set were evaluated by a board-eligible radiologist for accuracy. The procedures used to ensure the accuracy of ground-truth segmentations are detailed in Appendix 2.

Image Pre-Processing

To make data loading faster, we converted the DICOMs of each brain MRI into a 3D NIfTI file.[15] MRI volumes were then corrected for intensity inhomogeneities, including B1-field variations.[16,17] Then, the skull, face, and neck tissues were removed, leaving only the brain.[18] The resultant 3D images were cropped around the extracted brain. To overcome memory limitations, we cropped 64×64×64-voxel boxes of the MRI volume that contained each segmentation target. Details of pre-processing are provided in Appendix 3.

3D CapsNet

We built on the 2D CapsNets, as introduced by LaLonde et al[8], to develop 3D CapsNets for volumetric segmentation. CapsNets are composed of three main components: 1) capsules that each encode a structure together with the *pose* of that structure: the pose is an n-dimensional vector that learns to encode orientation, size, curvature, location, and other spatial information about the structure; 2) a supervised learning paradigm that learns the transforms between the poses of the parts (e.g. head and tail of hippocampus) and the pose of the whole (e.g. the entire hippocampus); and 3) a clustering paradigm that detects a whole if the poses of all parts (after getting transformed) vote for matching poses of the whole. Therefore, any CapsNet architecture requires procedures for: 1) creation of the first capsules from the input; 2) learning transforms between the poses of parts and wholes; and 3) clustering the votes of the parts to detect wholes. Details about the fundamental difference between CapsNets and other deep learning methods are provided in Appendix 4.

Figure 1.A shows the architecture of our 3D CapsNet. The first layer, Conv1, performs 16 convolutions (5×5×5) on the input volume to generate 16 feature volumes, which are reshaped into 16D vectors at each voxel. The 16D vector at each voxel provides the first pose that can learn to encode spatial information at that voxel. The next layer, PrimaryCaps2, has two capsule channels that learn two 16D-to-16D convolutional transforms (5×5×5) from the poses of the previous layer to the poses of the next layer. Likewise, the next *convolutional* capsule layers (green layers in Figure 1.A) learn m-to-n-dimensional

transforms between the poses of the previous layer and the poses of the next layer. The number of transforms at each layer matches the number of capsule channels (shown by stacks of capsules in Figure 1.A). Our CapsNet has downsampling and upsampling limbs. The downsampling limb learns *what* structure is present at each voxel, and the skip connections from downsampling to upsampling limbs preserve *where* each structure is on the image. Downsampling is done using 5×5×5 convolutional transforms with stride = 2. The poses in the deeper parts of the downsampling limb have more pose components (up to 64) to be able to encode more complex spatial information. Additionally, layers in the deeper parts of the model contain more capsule channels (up to 8) to be able to encode more structures at each voxel, since each voxel in these layers corresponds to multiple voxels in the input that can each represent a separate structure. Upsampling is done using 4×4×4 transposed convolutional transforms with stride = 2 (turquoise layers in Figure 1A). The final layer, FinalCaps13, contains one capsule channel that learns to activate capsules within the segmentation target and deactivate them outside the target. Details about how the final layer activations were converted into segmentations are provided in Appendix 5.

To find clusters of the agreeing votes of the parts, we used the inner products between the poses of the parts and the aggregate pose of the whole.[6] The details are provided in Appendix 6. We used Dice loss to train our models and to evaluate segmentation accuracy.[19]

3D U-Net

The 3D U-Net was used as a benchmark to compare the performance of the 3D CapsNets. The U-Net is considered among the highest-performing segmentation algorithms in diagnostic imaging.[20–23] The U-Net has shown strong auto-segmentation accuracy across a variety of different imaging modalities and anatomic structures.[21–26] Figure 1.B shows the architecture of our 3D U-Net. The input image undergoes 64 convolutions (3×3×3) to generate 64 feature maps. These volumes then undergo batch normalization and ReLU activation. Similar operations are carried out again, followed by downsampling using max-pooling (2×2×2). The downsampling and upsampling limbs each include four units. Upsampling is done using 2×2×2 transposed convolutions with stride = 2. The final layer carries out a 1×1×1 convolution to aggregate all 64 channels, followed by soft thresholding using the sigmoid function. The model learns to output a number close to 1 for each voxel inside the segmentation target, and a number close to 0 for each voxel outside the target.

Model Training

We used Dice loss for training our models. Adam optimizer was used with the following hyperparameters: $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. Training was done using 50 epochs, each consisting of all 3,199 brain MRIs in the training set, and with the batch size of four. Because of the large epoch size, we split each epoch into mini-epochs that each comprised 30 batches (120 MRIs). After each mini-epoch during training, the Dice loss was computed for the validation set (117 MRIs).

We used dynamic paradigms for learning rate scheduling and for selecting the best models. The initial learning rate was set at 0.002. The validation set Dice loss was monitored after each mini-epoch, and if it did not decrease over 10 consecutive mini-epochs, the learning rate was decreased by half. The minimum

learning rate was set at 0.0001. The model with the lowest Dice loss *over the validation set* was selected as the best model and was used for testing. The training hyperparameters are detailed in Appendix 7.

Model Performance

Segmentation accuracy was compared, using Dice scores, for the third ventricle, thalamus, and hippocampus. These structures were chosen to represent neuroanatomic structures with varying degrees of segmentation difficulty. Third ventricle is an easy structure because it is a cerebrospinal fluid (CSF) filled cavity with clear boundaries. Thalamus is a medium-difficulty structure because it is abutted by CSF on one side and brain parenchyma on the other side. Hippocampus is a difficult structure because it has a complex shape and is abutted by multiple brain structures with indistinct borders. To evaluate performance across different dataset sizes, segmentation performance was tested using the full data set as well as random subsets of (600, 240, and 120) MRI volumes.

To evaluate the performance of CapsNets on the images that were not represented during training, we trained the models using only data from the right thalamus and right hippocampus. We subsequently evaluated the segmentation accuracy of the models on test sets that only included the images of the left thalamus and left hippocampus. Since the left brain structures in the test set were not represented in the distribution of the right brain structures in the training set, this experiment evaluated the out-of-distribution performance of the models.

Relative efficiency of the segmentation models was measured by the number of trainable parameters, the memory required to run the model (in megabytes), the computational times required for training, and the computational times required for segmenting each MRI.

For all experiments, the mean segmentation accuracies over the test set were compared between CapsNets and U-Nets using paired-samples t-tests. The mean Dice scores together with their 95% confidence intervals were also tabulated for the two models and the three brain structures that were segmented.

Implementation

Image pre-processing was done using FreeSurfer and Python. PyTorch was used for model development and testing. The SciPy package was used for statistical analyses. Training and testing of the models were run on AWS instances (4 vCPUs, 61 GB RAM, 12 GB NVIDIA GK210 GPU with Tesla K80 Accelerators). The code used to train and test our models is publicly available at:
github.com/Aneja-Lab-Yale/Aneja-Lab-Public-CapsNet.

Results

The accuracy of 3D CapsNets in segmenting various brain structures is above 90% and is within 1.5% of the accuracy of U-Nets. Figure 2 shows the segmentation of various brain structures by both models in a patient. Table 2 compares the segmentation accuracy of the two models, measured by Dice scores.

The 3D CapsNets achieved better out-of-distribution segmentation accuracy compared to 3D U-Nets. When both models were trained to segment right brain structures and tested on segmenting contralateral left brain structures, 3D CapsNets significantly outperformed 3D U-Nets with Dice scores more than 30% higher. Figure 3 illustrates segmentation of the contralateral left thalamus and hippocampus by both models in a patient. Table 3 compares out-of-distribution segmentation accuracy between the two models.

The 3D CapsNets and 3D U-Nets achieved similar segmentation accuracies (within 3% of each other) when trained on smaller datasets. When the size of the training set was decreased from 3199 to 600 brain MRIs, both CapsNet and U-Net were minimally affected. Further decrease in the size of the training set down to 120 brain MRIs caused a decrease in the accuracy of both models down to 85. Figure 4 shows the performance of both models when trained on smaller datasets.

The 3D CapsNets are over 10 times smaller compared to 3D U-Nets. The 3D CapsNet has 7.4 million trainable parameters, while the corresponding 3D U-Net has 90.3 million trainable parameters. In addition, the 3D CapsNet has fewer layers and fewer steps of image propagation in forward and backward passes, leading to a smaller cumulative size of the feature volumes in the entire model. The 3D CapsNet and 3D U-Net respectively hold 228 and 1,364 megabytes of cumulative feature volumes in the entire model. Figure 5.A compares the size of 3D CapsNet and 3D U-Net models.

The 3D CapsNets train slightly faster compared with U-Nets by approximately 0.5 seconds per sample. When we compared the training time between the two models (on an AWS instance with NVIDIA GK210 GPU providing 12 GB of GPU memory), 3D CapsNets and 3D U-Nets respectively took about 1.5 and 2 seconds per example per epoch to train. The two models are equally fast during testing, taking 0.9 seconds to segment the MRI volume. Figure 5.B compares the training and testing times between the two models.

Discussion

In this study, we introduce 3-D CapsNets as a superior approach to autosegmentation because of its flexibility in characterizing anatomy that is not represented in the training set. This advantage is critically important because no training set can comprehensively capture every way that the anatomy might be distorted by cancer. Our results show that 3D CapsNets have high segmentation accuracy for segmenting various brain structures with Dice scores above 90%. While our CapsNets are one order of magnitude smaller than traditional U-Nets, their segmentation accuracy is within 1.5%. In out-of-distribution segmentation, our CapsNets outperformed U-Nets with Dice scores more than 30% higher. These results suggest that 3D CapsNets may generalize better than traditional deep learning auto-segmentation methods on data not well represented in training.

This study extends the literature in key ways. This is the first study to develop 3D CapsNets, and to use them for volumetric image segmentation. To overcome memory limitations, we added a pre-processing step in which we placed a 3D box around the segmentation target. We subsequently segmented the volume within this box using our models. We also overcame the problem of unstable loss optimization during the training of CapsNets, which is a known problem in CapsNet training,[8] by converting the outputs of the final layer of CapsNets using a forgiving paradigm (details are provided in Appendix 5). We also showed that

3D CapsNets can segment images that are different from the images in the training data. Given this capability of CapsNets to generalize to images with new spatial features that were not represented in the training data, CapsNets may be the solution to the problem of segmenting brain images with changed spatial features caused by space-occupying lesions.

Our results corroborate previous studies that deep learning is effective in medical image segmentation.[20–22,26,27] Multiple prior studies have shown the success of U-Nets in biomedical image segmentation.[20,22,27] The 3D U-Net that we coded in this study also showed strong performance in segmenting brain MRIs. The only previous study that used 2D CapsNet to segment brain MRIs did not achieve impressive results.[8] In this study, we developed 3D CapsNets that rival U-Nets in performance and efficiency. Moreover, we found 3D CapsNets to have improved out-of-distribution generalizability compared to U-Nets.

Our results corroborate previous studies that CapsNets have superior out-of-distribution generalizability compared to more traditional deep learning methods, including U-Nets.[6,7] In 2D object recognition, 2D CapsNets outperformed other deep learning methods when the objects were imaged from viewpoints that were not represented during training.[6] In 2D image segmentation, 2D CapsNets outperformed 2D U-Nets in segmenting rotated and flipped images.[7] Our study extends the literature by showing that 3D CapsNets outperform 3D U-Nets in segmenting mirror-image 3D image volumes that were not represented during training.

This study also corroborates previous studies showing that CapsNets can model spatial features more efficiently, achieving better or similar performance compared to other deep learning methods while their model size is significantly smaller.[5–7] Our 3D CapsNet is one order of magnitude smaller than 3D U-Net while achieving similar segmentation results. Our results also corroborate with previous studies that show faster convergence of CapsNets during training, as compared to other deep learning methods.[6,7] Our 3D CapsNets are slightly faster to train compared to 3D U-Nets, because they converge faster. Although clustering of pose vectors between capsule layers slows down CapsNets, the significantly fewer trainable parameters lead to faster convergence and, as a result, faster training of CapsNets. During testing, however, the two models are equally fast. Given that the forward pass through the fixated, trained parameters during testing is faster compared to the forward *and* backward passes during training, the larger size of the 3D U-Net does not slow it down as much during testing as it does during training. At the same time, clustering between the capsule layers slows down the 3D CapsNet during testing to the same degree as during training. The overall effect of these opposing factors, fewer layers of CapsNet (making it faster) but more complex computations between CapsNet layers (making it slower), make CapsNet and U-Net equally fast during testing.

To develop 3D CapsNets and make them work for volumetric brain MRI segmentation, we explored multiple design options, hyperparameters, loss functions, and implementation details to find optimal solutions. We used the validation set to explore these questions, and tested our final model on the test set only once. While our model performs well for volumetric segmentation of T1-weighted brain MRIs, we did not evaluate its performance for segmentation of other organs or other imaging modalities. We assume that our model would need modifications to perform well on other segmentation tasks or on brain MRIs that are pre-processed differently. We have described the experiments that helped us find optimal solutions for our

design questions in the supplemental material, and we welcome further research to generalize our 3D CapsNet to the segmentation of other organs and to other imaging modalities.

There are limitations of this study that should be noted. First, the ground truth segmentations were elicited from FreeSurfer software package, which is shown to have segmentation accuracy similar to human experts.[11–13,28] To ensure that possible inaccurate ground-truth segmentations would not negatively affect our study, a radiologist confirmed and approved the segmentations of all MRIs in the test set as well as 120 randomly-selected MRIs in the training set. Second, we only validated our model for the segmentation of three brain structures, having varying levels of difficulty. Our model may not generalize to other anatomic structures or other imaging modalities. Third, computation times were measured using the same computing resources, including GPU memory. While we showed faster training of CapsNets compared to U-Nets, our results may not translate to different computational settings. Last, we did not compare 3D CapsNets against all available deep learning segmentation methods. However, we compared 3D CapsNets with U-Nets, because U-Nets have become the standard deep learning method for biomedical image segmentation.

Conclusion

In this study, we show 3D CapsNet as a superior method of brain image segmentation because of its ability to segment neuro-anatomy that was not represented in the training data. Moreover, our 3D CapsNet is one order of magnitude smaller than the equivalent U-Net, but still achieves comparable performance in segmenting various brain structures.

Acknowledgements:

Arman Avesta is a PhD Student in the Investigative Medicine Program at Yale which is supported by CTSA Grant Number UL1 TR001863 from the National Center for Advancing Translational Science, a component of the National Institutes of Health (NIH). The contents of this article are solely the responsibility of the authors and do not necessarily represent the official view of NIH. The data used in this article were obtained from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database (adni.loni.usc.edu). The investigators within the ADNI contributed to the design and implementation of ADNI but did not participate in the analysis or writing of this article.

Disclosures:

Arman Avesta:
Arman Avesta is on the trainee editorial board of *Radiology: Artificial Intelligence*. Journal policy recused the author from having any role in the peer review of this manuscript.
Research Funding: CTSA UL1 TR001863 from the National Center for Advancing Translational Science.

The MedNet, Inc, American Cancer Society, National Science Foundation, Agency for Healthcare Research and Quality, National Cancer Institute, ASCO
Potential Conflict of Interest: Arman Avesta holds securities at Hyperfine Inc.

Yongfeng Hui:
The publication was written prior to Yongfeng Hui joining Amazon.

Mariam Aboian:
Research funding: KL2 TR001862 from the National Center for Advancing Translational Science and NIH roadmap for Medical Research. The contents of this article are solely the responsibility of the authors and do not necessarily represent the official view of NIH.

James S. Duncan
Patents, Royalties, Other Intellectual Property: Systems, Methods and Apparatuses for Generating Regions of Interest from Voxel Mode Based Thresholds, Publication No: US20190347788A1, application No. 15/978,904. Filed on May 14, 2018, Publication Date: November 14, 2019. Inventors: Van Breugel J, Abajian A, Treli- hard J, Smolka S, Chapiro J, Duncan JS and Lin M. Joint application from Philips, N.V. and Yale University. US Patent 10,832,403 (2020)

Harlan M. Krumholz
Employment: Hugo Health (I), FPrime
Stock and Other Ownership Interests: Element Science, Refactor Health, Hugo Health
Consulting or Advisory Role: UnitedHealthcare, Aetna
Research Funding: Johnson and Johnson
Expert Testimony: Siegfried and Jensen Law Firm, Arnold and Porter Law Firm, Martin/Baughman Law Firm

Sanjay Aneja:
Sanjay Aneja is an Associate Editor for *JCO Clinical Cancer Informatics*. Journal policy recused the author from having any role in the peer review of this manuscript.
Consulting or Advisory Role: Prophet Consulting (I)
Research Funding: The MedNet, Inc, American Cancer Society, National Science Foundation, Agency for Healthcare Research and Quality, National Cancer Institute, ASCO
Patents, Royalties, Other Intellectual Property: Provisional patent of deep learning optimization algorithm
Travel, Accommodations, Expenses: Prophet Consulting (I), Hope Foundation
Other Relationship: NRG Oncology Digital Health Working Group, SWOG Digital Engagement Committee, ASCO mCODE Technical Review Group

References

1. Feng CH, Cornell M, Moore KL, et al. Automated contouring and planning pipeline for hippocampal-avoidant whole-brain radiotherapy. *Radiat Oncol Lond Engl* 2020;15:251.

2. Dasenbrock HH, See AP, Smalley RJ, et al. Frameless Stereotactic Navigation during Insular Glioma Resection using Fusion of Three-Dimensional Rotational Angiography and Magnetic Resonance Imaging. *World Neurosurg* 2019;126:322–30.

3. Dolati P, Gokoglu A, Eichberg D, et al. Multimodal navigated skull base tumor resection using image-based vascular and cranial nerve segmentation: A prospective pilot study. *Surg Neurol Int* 2015;6:172.

4. Despotović I, Goossens B, Philips W. MRI Segmentation of the Human Brain: Challenges, Methods, and Applications. *Comput Math Methods Med* 2015;2015:e450341.

5. Seo H, Khuzani MB, Vasudevan V, et al. Machine learning techniques for biomedical image segmentation: An overview of technical aspects and introduction to state-of-art applications. *Med Phys* 2020;47:e148–67.

6. Sabour S, Frosst N, Hinton GE. Dynamic routing between capsules. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Red Hook, NY, USA: Curran Associates Inc.; 2017:3859–69.

7. Hinton GE, Sabour S, Frosst N. Matrix capsules with EM routing. In: *International Conference on Learning Representations 2018*.

8. LaLonde R, Xu Z, Irmakci I, et al. Capsules for biomedical image segmentation. *Med Image Anal* 2021;68:101889.

9. Survarachakan S, Johansen JS, Aarseth M, et al. Capsule Nets for Complex Medical Image Segmentation Tasks. In: Gjovik, Norway; 2020:15.

10. Crawford KL, Neu SC, Toga AW. The Image and Data Archive at the Laboratory of Neuro Imaging. *NeuroImage* 2016;124:1080–3.

11. Weiner M, Petersen R, Aisen P. *Alzheimer's Disease Neuroimaging Initiative*. URL: https://clinicaltrials.gov/ct2/show/NCT00106899. Accessed on: 03/21/2022.; 2014.

12. Clerx L, Gronenschild EHBM, Echavarri C, et al. Can FreeSurfer Compete with Manual Volumetric Measurements in Alzheimer's Disease? *Curr Alzheimer Res* 2015;12:358–67.

13. Ochs AL, Ross DE, Zannoni MD, et al. Comparison of Automated Brain Volume Measures obtained with NeuroQuant and FreeSurfer. *J Neuroimaging Off J Am Soc Neuroimaging* 2015;25:721–7.

14. Fischl B. FreeSurfer. *NeuroImage* 2012;62:774–81.

15. Li X, Morgan PS, Ashburner J, et al. The first step for neuroimaging data analysis: DICOM to NIfTI conversion. *J Neurosci Methods* 2016;264:47–56.

16. Fischl B, Salat DH, Busa E, et al. Whole brain segmentation: automated labeling of neuroanatomical structures in the human brain. *Neuron* 2002;33:341–55.

17. Ganzetti M, Wenderoth N, Mantini D. Quantitative Evaluation of Intensity Inhomogeneity Correction Methods for Structural MR Brain Images. *Neuroinformatics* 2016;14:5–21.

18. Somasundaram K, Kalaiselvi T. Automatic brain extraction methods for T1 magnetic resonance images using region labeling and morphological operations. *Comput Biol Med* 2011;41:716–25.

19. Taha AA, Hanbury A. Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool. *BMC Med Imaging* 2015;15:29.

20. Cardenas CE, Yang J, Anderson BM, et al. Advances in Auto-Segmentation. *Semin Radiat Oncol* 2019;29:185–97.

21. Rudie JD, Weiss DA, Colby JB, et al. Three-dimensional U-Net Convolutional Neural Network for Detection and Segmentation of Intracranial Metastases. *Radiol Artif Intell* 2021;3:e200204.

22. Rauschecker AM, Gleason TJ, Nedelec P, et al. Interinstitutional Portability of a Deep Learning Brain MRI Lesion Segmentation Algorithm. *Radiol Artif Intell* 2022;4:e200152.

23. Weiss DA, Saluja R, Xie L, et al. Automated multiclass tissue segmentation of clinical brain MRIs with lesions. *NeuroImage Clin* 2021;31:102769.

24. Punn NS, Agarwal S. Modality specific U-Net variants for biomedical image segmentation: a survey. *Artif Intell Rev* https://doi.org/10.1007/s10462-022-10152-1.

25. Elguindi S, Zelefsky MJ, Jiang J, et al. Deep learning-based auto-segmentation of targets and organs-at-risk for magnetic resonance imaging only planning of prostate radiotherapy. *Phys Imaging Radiat Oncol* 2019;12:80–6.

26. Francis S, Jayaraj PB, Pournami PN, et al. ThoraxNet: a 3D U-Net based two-stage framework for OAR segmentation on thoracic CT images. *Phys Eng Sci Med* 2022;45:189–203.

27. Rudie JD, Weiss DA, Saluja R, et al. Multi-Disease Segmentation of Gliomas and White Matter Hyperintensities in the BraTS Data Using a 3D Convolutional Neural Network. *Front Comput Neurosci* 2019;13.

28. Duong MT, Rudie JD, Wang J, et al. Convolutional Neural Network for Automated FLAIR Lesion Segmentation on Clinical Brain MR Imaging. *Am J Neuroradiol* https://doi.org/10.3174/ajnr.A6138.

29. Yaakub SN, Heckemann RA, Keller SS, et al. On brain atlas choice and automatic segmentation methods: a comparison of MAPER & FreeSurfer using three atlas databases. *Sci Rep* 2020;10:2837.

Figure 1: CapsNet (A) and U-Net (B) architectures. Both models process 3D volumes in all layers, with dimensions shown on the left side. *D*, *H*, and *W* respectively represent the depth, height, and width of the image in each layer. In (A), the number over the Conv1 layer represents the number of channels. The numbers over the capsule layers (ConvCaps, DeconvCaps, and FinalCaps) represent the number of pose components. The stacked layers represent capsule channels. In (B), the numbers over each layer represent the number of channels. In the 3D U-Net, the convolutions have stride=1 and the transposed convolutions have stride = 2. Please note that the numbers over capsule layers show the number of pose components, while the numbers over non-capsule layers show the number of channels.

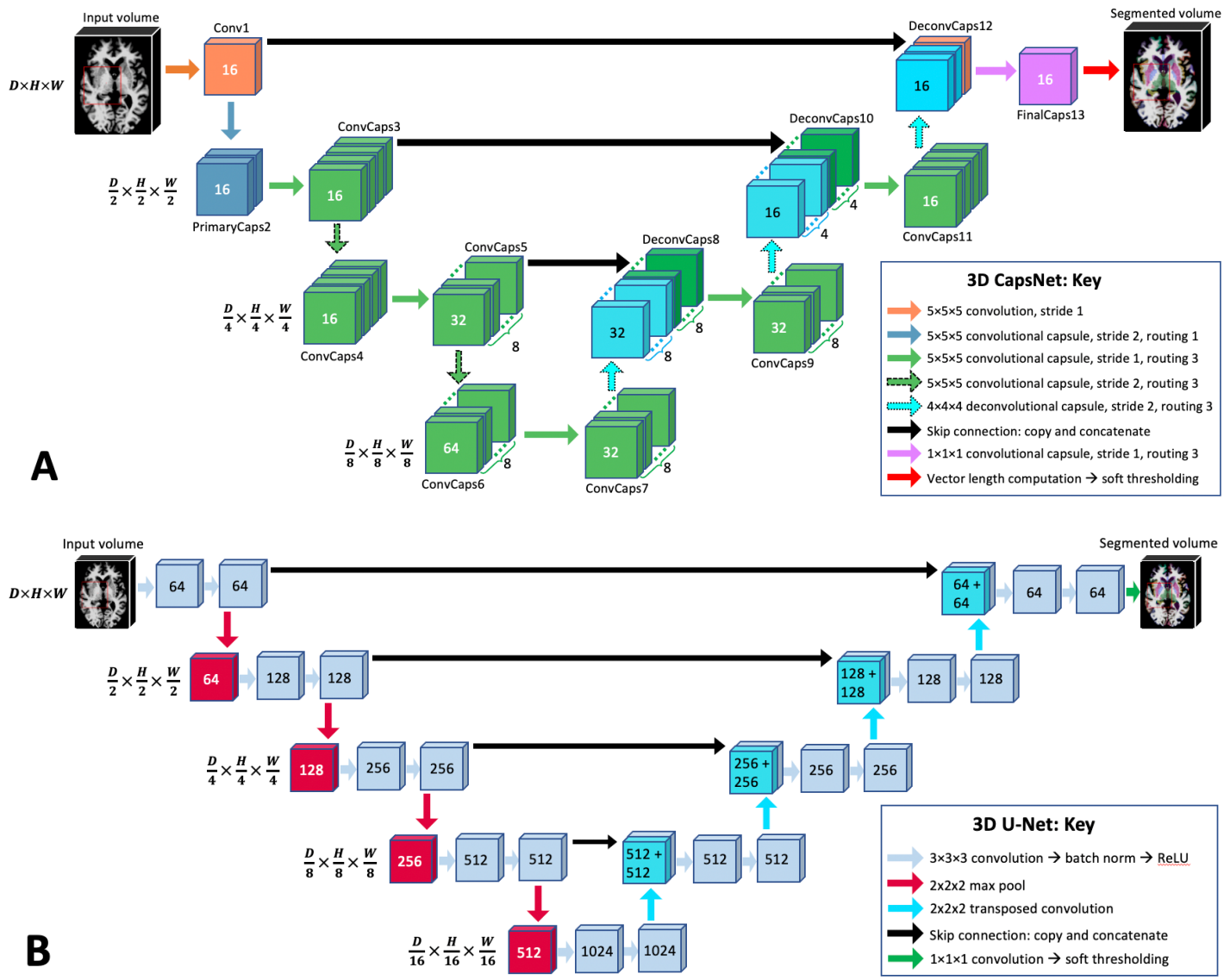


Table 1: Study participants tabulated by the training, validation, and test sets.

Data Partitions	Number of MRI volumes	Number of patients	Age mean ± SD	Gender†	Diagnosis††
Training set	3199	841	76 ± 7	42% F, 58% M	29% CN, 54% MCI, 17% AD
Validation set	117	30	75 ± 6	30% F, 70% M	21% CN, 59% MCI, 20% AD
Test set	114	30	77 ± 7	33% F, 67% M	27% CN, 47% MCI, 26% AD

† F: female; M: male.

†† CN: cognitively normal; MCI: mild cognitive impairment; AD: Alzheimer's disease.

Figure 2: CapsNet vs U-Net in segmenting brain structures that were represented in the training data. Segmentations for three structures are shown: 3rd ventricle, thalamus, and hippocampus. Target segmentations and model predictions are respectively shown in white and red. Dice scores are provided for the entire volume of the segmented structure *in this patient* (who was randomly chosen from the test set).

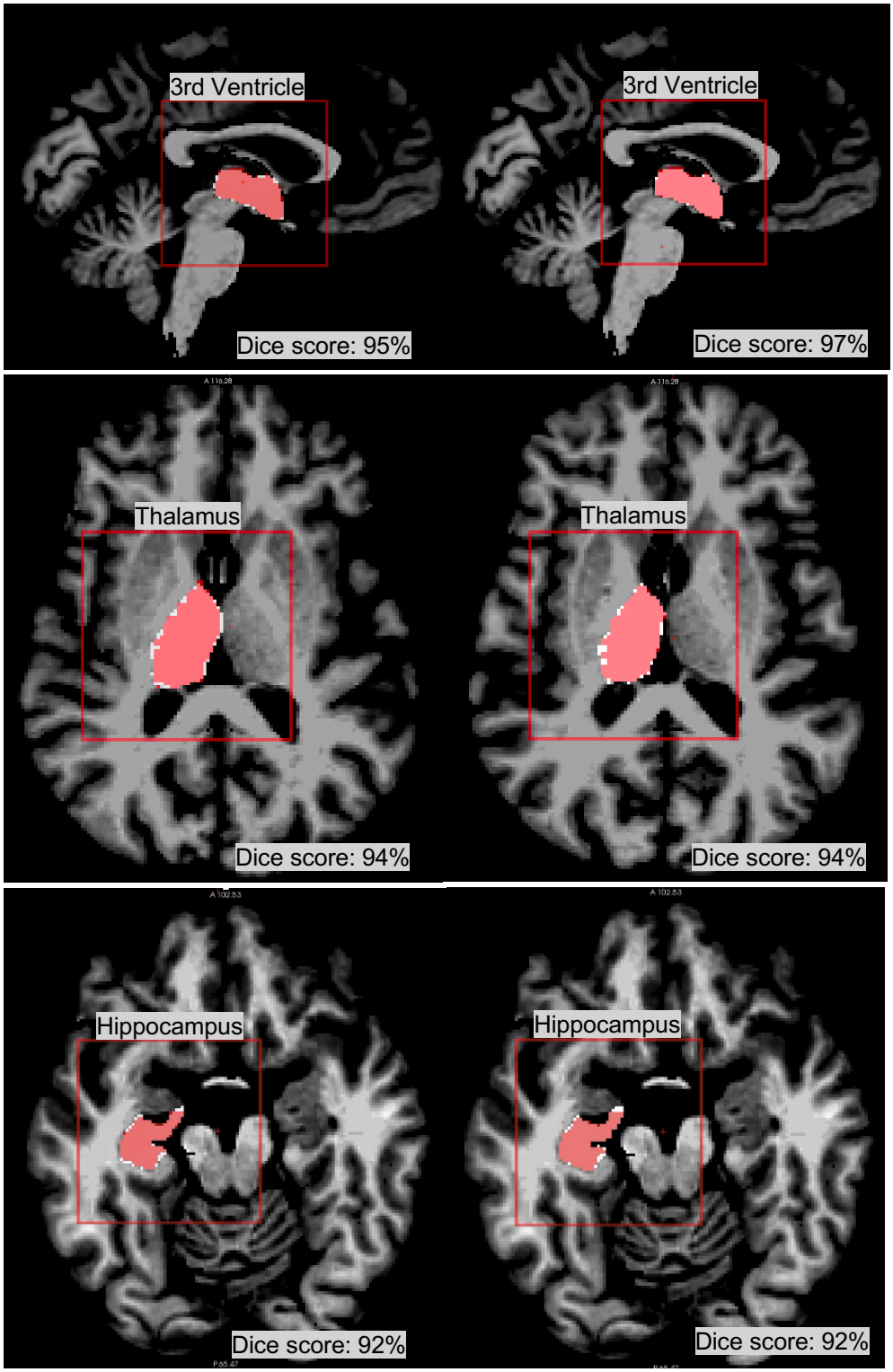


Table 2: CapsNet vs U-Net in segmenting brain structures that were represented in the training data. The segmentation accuracy was quantified using Dice scores on the test (114 brain MRIs). The 3rd ventricle, thalamus, and hippocampus respectively represent easy, medium, and difficult structures to segment.

Brain structure	CapsNet Dice score (95% CI)	U-Net Dice score (95% CI)	P-value†
3rd ventricle	93.6 (93.2 to 94.0) %	95.3 (95.0 to 95.6) %	< 0.01
Thalamus	93.6 (93.4 to 93.8) %	94.4 (94.3 to 94.6) %	< 0.01
Hippocampus	91.0 (90.7 to 91.3) %	92.5 (92.1 to 92.9) %	< 0.01

† Paired-samples t-test, degrees of freedom = 114 - 1 = 113

Figure 3: CapsNet outperforms U-Net in out-of-distribution segmentation. Both models were trained to segment right brain structures, and were tested to segment contralateral left brain structures. Target segmentations and model predictions are respectively shown in white and red. Dice scores are provided for the entire volume of the segmented structure *in this patient*. While CapsNet partially segmented the contralateral thalamus and hippocampus, U-Net poorly segmented thalamus and entirely missed the hippocampus.

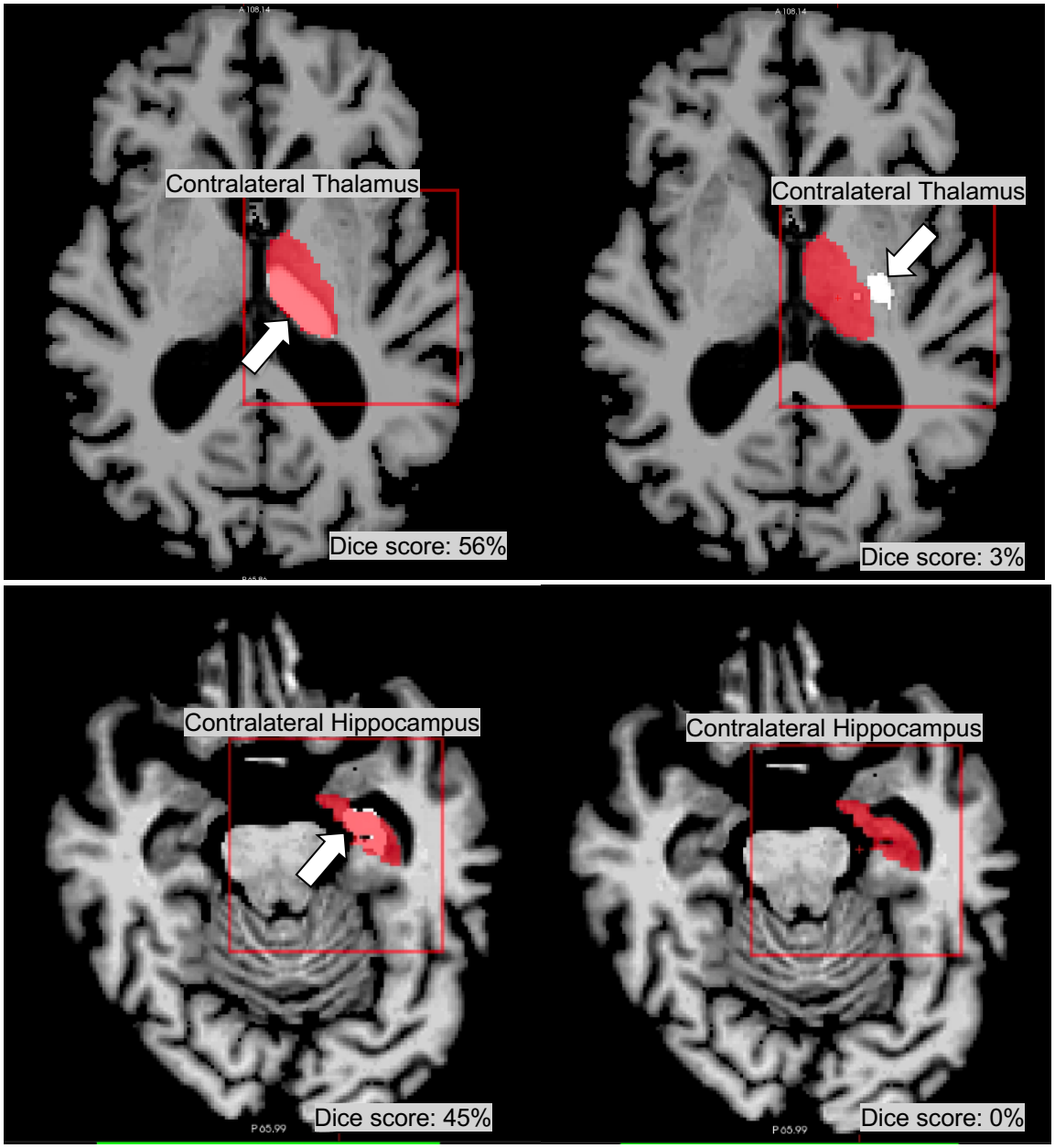


Table 3: CapsNet vs U-Net out-of-distribution segmentation accuracy. Both models were trained to segment the right thalamus and hippocampus. Then, they were tested on segmenting the contralateral left thalamus and hippocampus.

Brain structure	CapsNet Dice score (95% CI)	U-Net Dice score (95% CI)	P-value†
Thalamus	52 (46 to 58) %	16 (11 to 21) %	< 0.01
Hippocampus	43 (38 to 48) %	10 (6 to 14) %	< 0.01

† Paired-samples t-test, degrees of freedom = 114 - 1 = 113

Figure 4: CapsNet vs U-Net segmentation accuracy as a measure of training set size. When the size of the training set was decreased from 3199 to 600 brain MRIs, both models maintained their segmentation accuracy above 90%. Further decrease in the size of the training set down to 120 MRIs led to worsening of their segmentation accuracy down to 85% (measured by Dice scores).

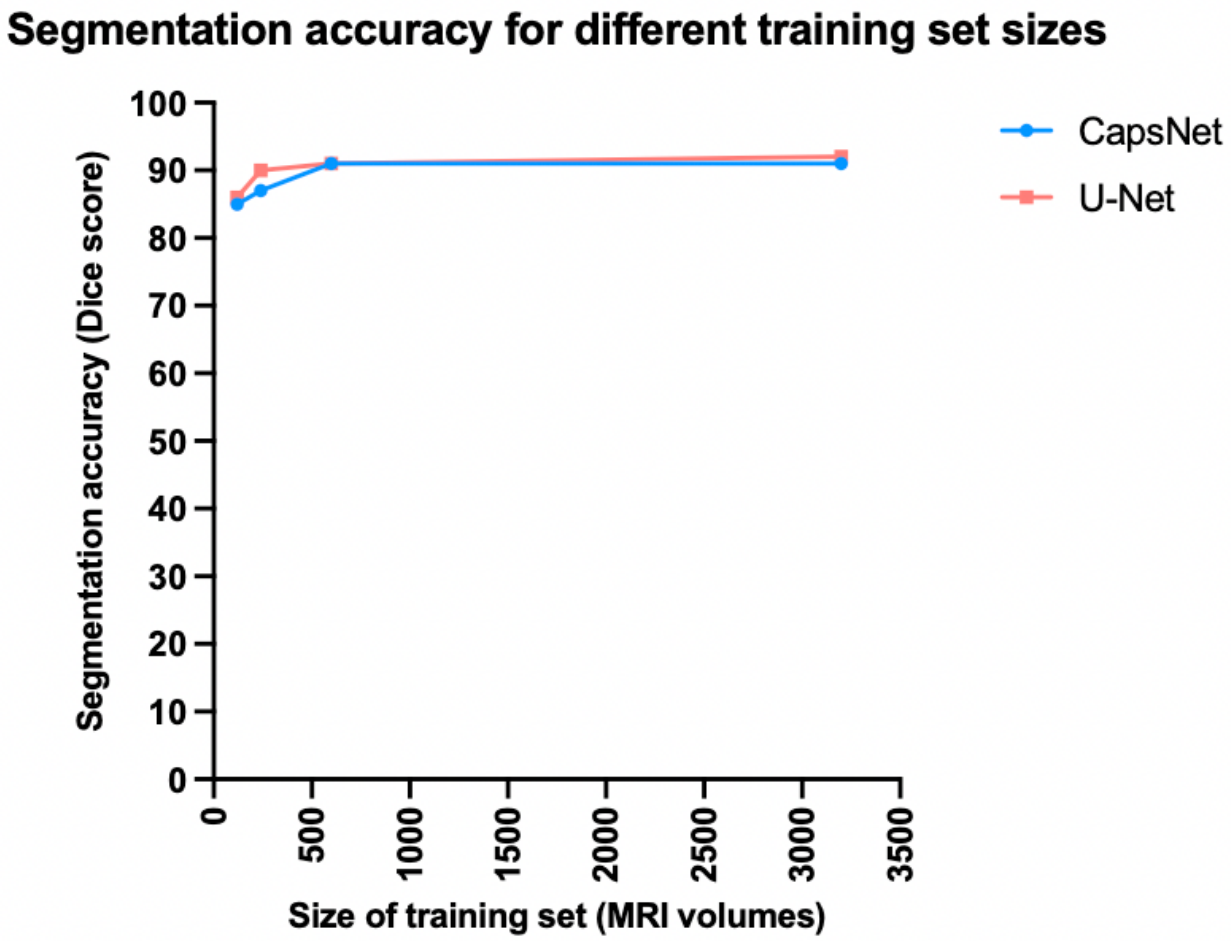
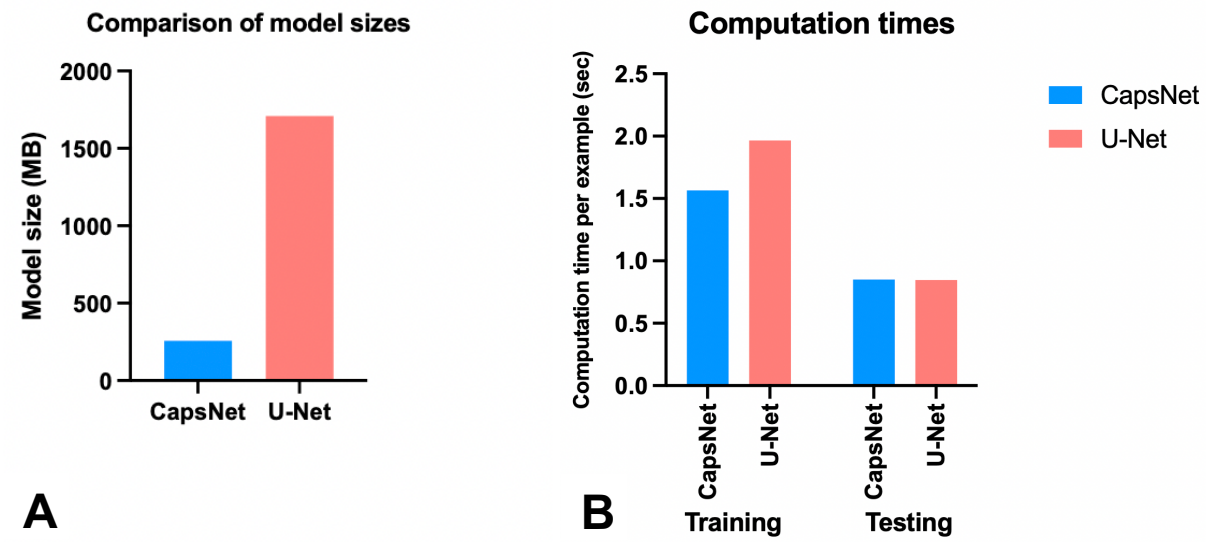


Figure 5: Model size (A) and computation times (B) compared between CapsNet and U-Net. The model size bars in (A) represent parameter size (28 and 345 MB for CapsNet and U-Net, respectively) plus the cumulative size of the forward and backward pass feature volumes (228 and 1364 MB for CapsNet and U-Net, respectively). The CapsNets train slightly faster (B), given that they have 93% fewer trainable parameters. However, clustering between the capsule layers slows down CapsNets, making them only slightly faster than U-Nets during training. The two models are equally fast during testing.



SUPPLEMENTAL MATERIAL

Appendix 1: MRI acquisition parameters

Field strength = 3.0 tesla
Coil = 8HR Brain

Weighting = T1
Flip angle=8.0 degree
TR = 6.6 ms
TE = 2.8 ms
TI = 900.0 ms

Acquisition type = 3D
Acquisition plane = Sagittal
Matrix size = 256×256×166 pixels (X×Y×Z)
Pixel size = 1×1×1.2 mm (X×Y×Z)
Pixel spacing: along X direction = 1 mm; along Y direction= 1 mm

Appendix 2: Ground-Truth Segmentations

In this study, the ground truth segmentations were established by the FreeSurfer software package.[1] Several previous studies have shown that FreeSurfer can segment brain images with accuracy similar to human experts, if the brain image is not distorted by space-occupying lesions.[1–4] This holds true for the brain images of normal individuals as well as the brain images of patients with mild cognitive impairment or Alzheimer's disease.[2] Since we used the brain images of normal individuals and patients with mild cognitive impairment or Alzheimer's disease in our study, FreeSurfer segmentations can be regarded as ground-truth.

Still, to ensure that FreeSurfer segmentations were free from error, 120 randomly-selected MRIs from the training set as well as all 114 MRIs in the test set were evaluated by a board-eligible radiologist for accuracy. To evaluate the segmentation accuracy for each brain MRI, color-coded segmentations were overlaid on T1-weighted brain images. To streamline visualizations, a BASH script was developed to automatically overlay the segmentations on T1-weighted images in FreeView.[1] The color-coded segmentations were made 50% transparent so that the underlying anatomy on T1-weighted image could be visualized. The radiologist then scrolled through the images in axial, coronal, and sagittal planes and visually inspected the segmentations. Segmentation of each brain structure was deemed acceptable if:

1) the borders of color-coded segmentation did not deviate more than two voxels from the borders of the corresponding brain structure seen on the T1-weighted image;
2) the segmentation included all clinically-important portions of the structure (for instance, the entire tail of the hippocampus should be included in the hippocampus segmentation); and
3) the segmentation excluded all clinically-important portions of neighboring structures (for instance, the optic nerves and chiasm should be excluded from the amyglada segmentation).

We planned to manually correct the segmentation of any brain structure that did not meet any of the above criteria. However, all visualized segmentations met all criteria detailed above.

One might question the purpose of this study if FreeSurfer can already segment brain images with expert-level accuracy. The aim of this study was to develop 3D capsule networks for volumetric neuroanatomical segmentation. While we used non-distorted brain images to train and test the model in this study, our ultimate goal is to segment neuroanatomy in brain images distorted by space-occupying lesions. Our central hypothesis is that capsule networks have the potential to segment distorted brain images because they can generalize to novel spatial features. FreeSurfer, on the other hand, does not have such a potential (segmenting distorted brain images) because it works by constructing distributions for the shape and location of each brain structure. When a space-occupying lesion distorts the brain anatomy, it changes the shape and location of brain structures to the extent that they fall out of their expected distributions. As a result, FreeSurfer often fails to accurately segment brain images distorted by space-occupying lesions because it does not provide out-of-distribution generalizability.[5]

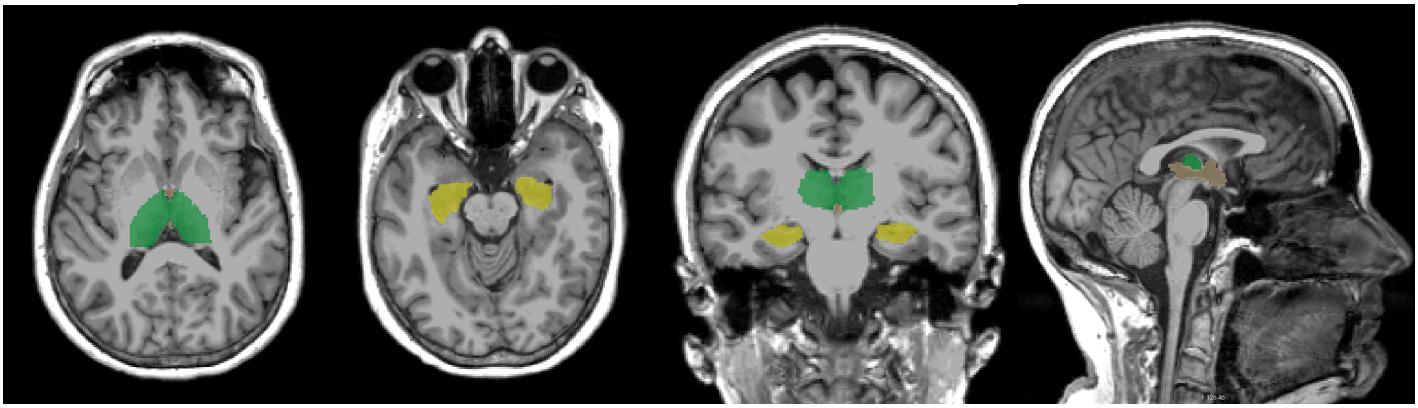


Figure S2: evaluating the quality of ground-truth segmentations. The thalami, hippocampi, and third ventricle are respectively shown in green, yellow, and brown. The segmentations were overlaid on T1-weighted images and were visualized in axial (A and B), coronal (C), and sagittal (D) planes.

Appendix 3: Pre-Processing

We used FreeSurfer to correct for intensity inhomogeneities including B1-field variations. FreeSurfer first registers the brain to MNI305 atlas. Then, pixel intensities are used to roughly segment the white matter. The variations in the pixel intensities in the white matter are then used to estimate the B1 field map Finally, B1 bias field correction is done by dividing the pixel intensities by the estimated bias field.[6]

We used FreeSurfer[1] for skull stripping. Skull stripping includes removal of the skull, face, and neck, only leaving the brain. FreeSurfer uses a hybrid method of skull stripping that combines a watershed algorithm and a deformable surface model.[7] This method first roughly segments the white-matter based on pixel intensities. Then, watershed algorithms are used to find the gray-white matter junction and the brain surface. Next, a deformable surface model is used to model the brain surface. The curvature of the brain surface at each point is computed, and these curvatures are used to register the brain surface onto an atlas. The atlas is formed by computing the curvatures of the brain sulci and gyri in several subjects. Notably, the sulci and gyri of the brain surface are constant among all humans, constituting positive and negative curvatures that are present in any brain image (unless the brain surface is markedly distorted by space-occupying lesions). The reconstructed brain surface, registered to the atlas, is then corrected in case the curvatures in a particular region of the surface do not make sense. The resulting corrected brain surface model is used for skull stripping.[7]

To overcome memory limitations, we cropped 64×64×64-voxel boxes of the MRI volume that contained each segmentation target. The position of each box (e.g. for segmenting the right hippocampus) was determined by visually inspecting 20 brain MRI volumes, randomly selected from the training set. The visual inspection included moving the "crop" box (64×64×64 voxels) over the MRI volumes to find the optimal position of the crop box. The position of this box was then fixated (with regard to the center of the skull-stripped brain volume) for each brain structure and for all subjects in the training, validation, and test sets. This task was done by the first author (board-eligible radiologist with 9 years of experience in neuroimaging research).

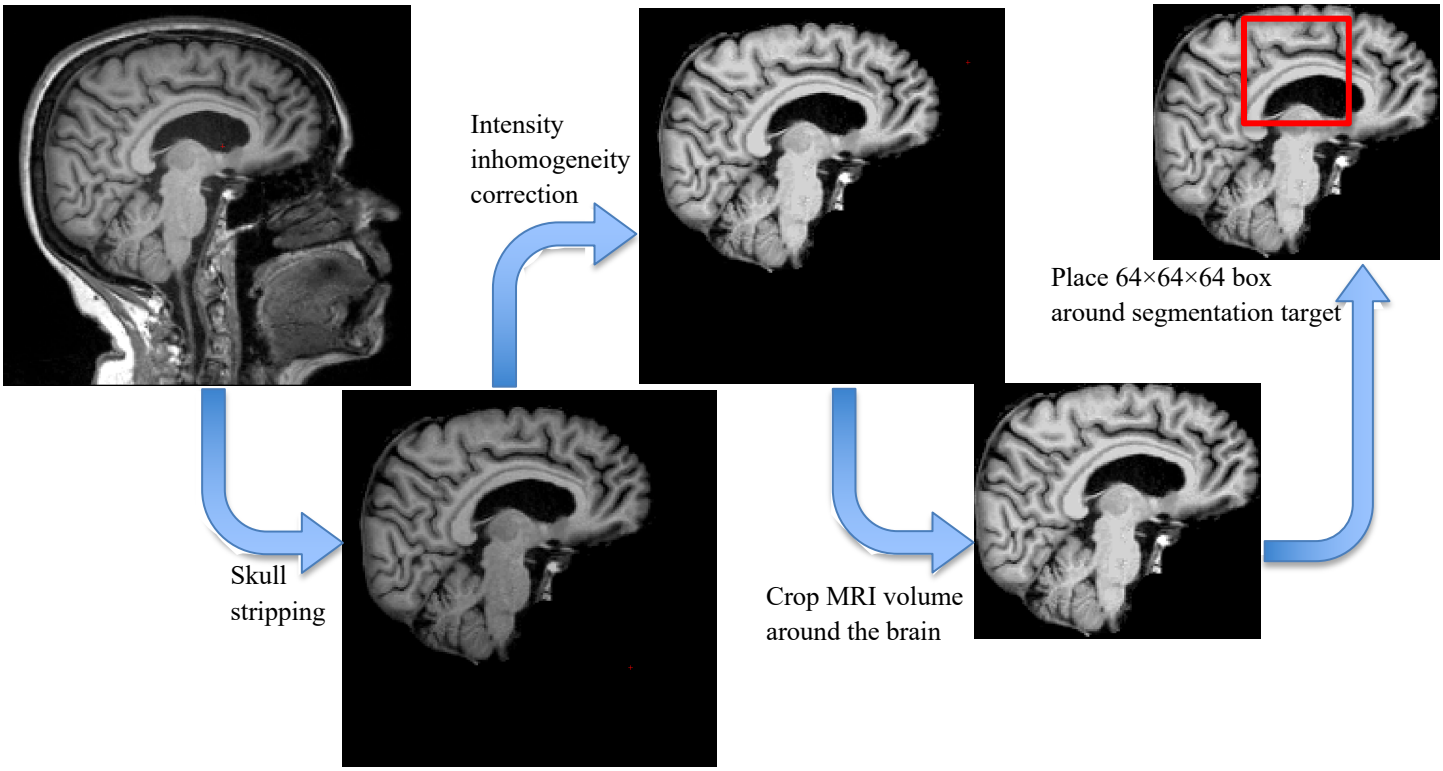
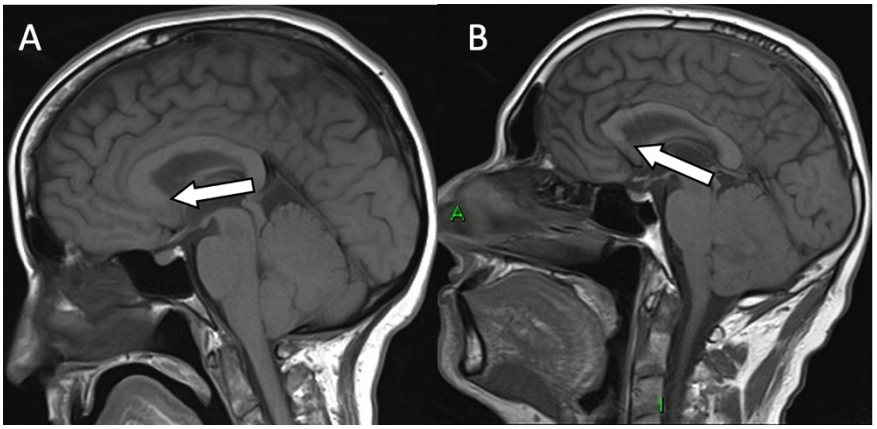
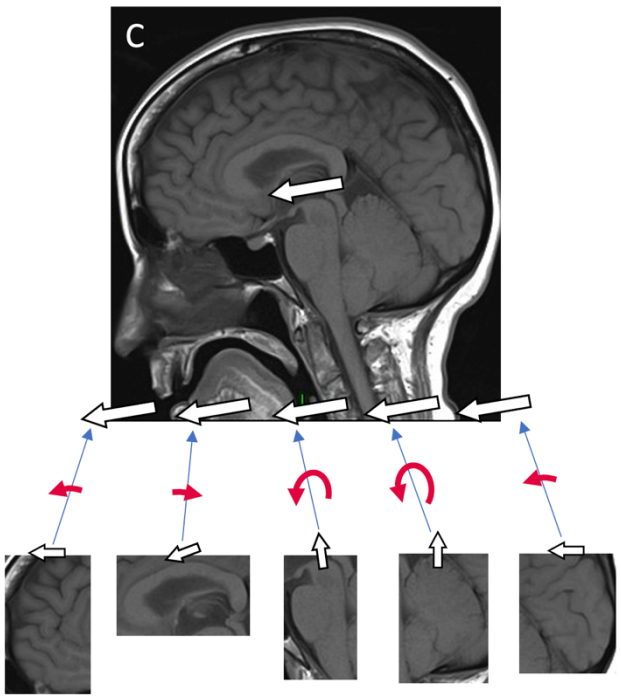


Figure S3: Pre-processing steps.

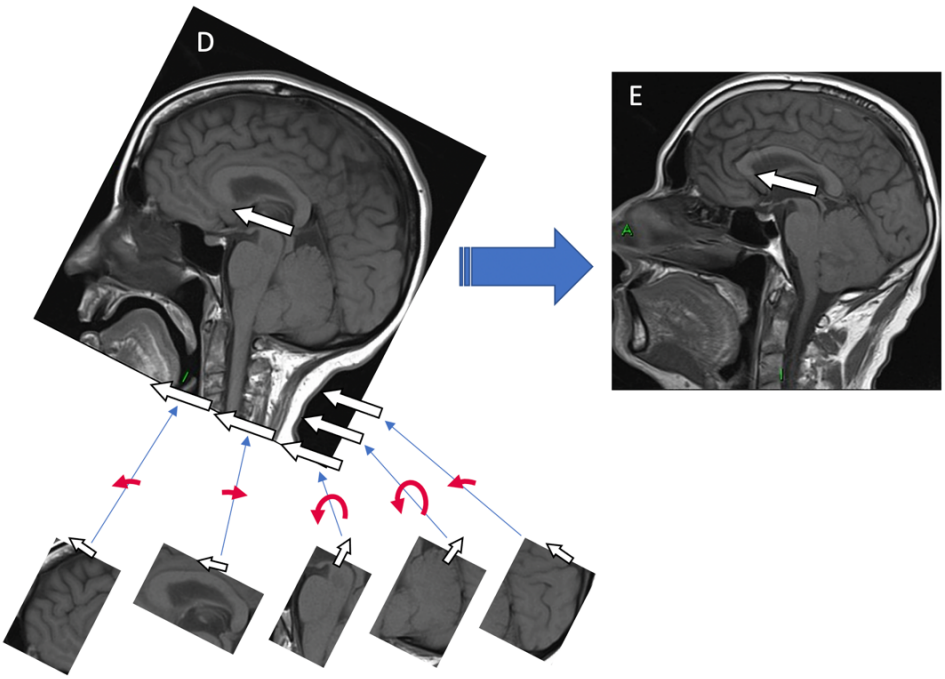
Appendix 4: Capsule Networks



Capsule networks (CapsNets) can detect objects when their spatial features change.[8] This is a fundamental property of CapsNets that enables them to perform well when a test example is not represented in the training data. (A) shows the sagittal T1-weighted brain MRI of a patient with a forward head tilt, and (B) shows the MRI of another patient with a backward head tilt. White arrows (connecting the posterior commissure to the anterior commissure) demonstrate the orientation of the brain. Let's assume that we have a CapsNet that is trained to segment the entire brain. Let's also assume that the training set only contains patients with forward head tilt (like in A). An ideal CapsNet should generalize to segment the brain in patients with a backward tilt (like in B). To achieve this goal, CapsNets encode the spatial features of each structure that they detect. The spatial features of the brain are encoded in a *pose* vector. The pose contains spatial features such as orientation, position, size, curvature, etc. Here, the orientation of the brain (one of the spatial features) is shown by the white arrow. Our goal is to illustrate how CapsNets detect a whole (the brain) when parts (frontal pole, corpus callosum, brainstem, cerebellum, occipital pole, etc.) all vote for the same spatial features of the whole.

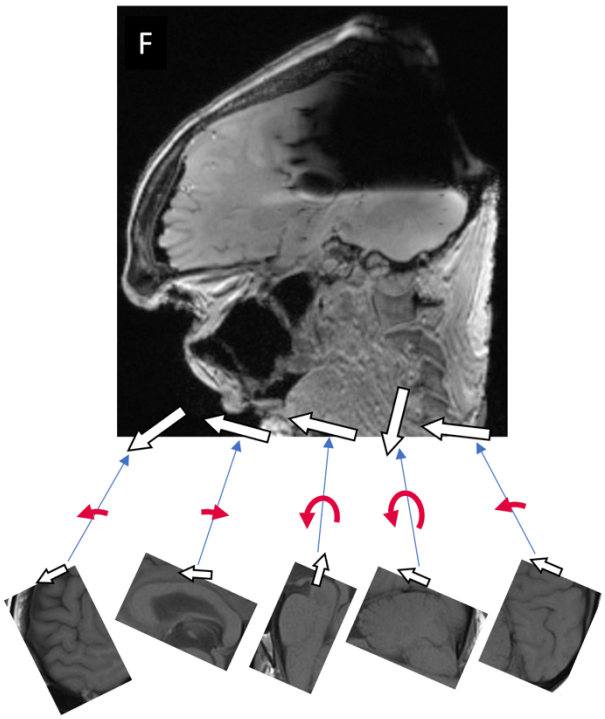


CapsNets are composed of three main ingredients: 1) capsules that each encode a structure together with the *pose* of that structure; 2) a supervised learning paradigm that learns the transforms between the poses of the parts (e.g. corpus callosum, brainstem) and the pose of the whole (e.g. the entire brain); and 3) a clustering paradigm that detects a whole if the poses of all parts (after getting transformed) vote for matching poses of the whole. Therefore, any CapsNet needs to: 1) learn the transforms between the poses of parts and wholes; and 2) cluster the votes of the parts to detect wholes.



(C) shows a CapsNet that has already detected parts of the brain and has encoded their spatial features (demonstrated by the smaller white arrow over each part). The red curved arrows demonstrate the transforms between the poses of the part and the pose of the whole. After transformation, each part votes for a candidate pose of the whole. If all these votes match, the

whole is present. Please note that we are only showing the orientations here for simplicity, but the pose vectors encode more complex spatial features.



In (E), We want the CapsNet to detect the backward-tilted brain while the model is only trained on forward-tilted brain images (such as in C). We can imagine that (E) is just the rotated version of (C), as demonstrated in (D). The parts are all rotated clockwise (compared to the poses of the parts in C). However, the same transforms (red curved arrows) can still transform the poses of the parts into the candidate poses of the whole. The candidate poses of the whole still match, and therefore the whole is detected. This process does not need any data augmentation: an ideal CapsNet can detect objects when they are rotated or have undergone other spatial changes, without the need for any data augmentation. This is because the CapsNet can still use the same transforms between the parts and the wholes (red curved arrows) even though the input image has rotated. Therefore, a change in the poses of the parts will cause an equivalent change in the pose of the whole, while the relationship between the poses of the parts and the whole remains the same. This is a powerful capability that makes CapsNets *equivariant* to the changes in the inputs: spatial change in the inputs will cause an equivalent spatial change in the pose of the detected objects.[8] Such CapsNets can still detect the changed objects and will encode these changes in the pose of the detected objects. As a result, a CapsNet that is trained on forward-tilted brains (such as in C) can detect backward-tilted brains (such as in E) without the need for any data augmentation.

This approach is fundamentally different from other machine learning methods such as U-Nets, which do not have equivariance capabilities. Instead, the max-pooling layers in U-Nets try to kill information about the changes in the inputs to make the model *invariant* to the changed inputs. In essence, CapsNets use equivariance to *encode and model the spatial changes* in the inputs, making CapsNets more efficient in handling variations of the same object.[8] On the other hand, U-Nets use information killing (in max-pooling layers) to make the model *invariant to the spatial changes* in the inputs. Therefore, U-Nets cannot efficiently detect variations of the same object.

(F) demonstrates why CapsNets are less susceptible to adversarial attacks compared to U-Nets. Here, this adversarial image contains all parts of the brain but with orientations that do not make sense, not making a whole. When the poses of the parts are transformed into the candidate poses of the whole (using the same transforms as in C), the candidate poses of the whole do not match. Therefore, the CapsNet would not detect a brain because of the mismatch between the candidate poses of the brain. On the other hand, a U-Net that is trained using augmented data may detect the parts. Such a U-Net has no mechanism to encode the orientation and other spatial features of each part. In the U-Net feature space, each part is either present or absent. Since all parts are present on this adversarial image, the U-Net can be fooled to detect the entire brain.

We can indeed use data augmentation to train U-Nets to detect objects with changed spatial features. We can also use adversarial training to prevent U-Nets from detecting adversarial images. But these inefficiencies lead to the need for a larger U-Net model. On the other hand, CapsNets handle the changed spatial features in a smarter way. This allows CapsNets, which are one order of magnitude smaller compared to U-Nets, to achieve similar results.[9]

Appendix 5: Converting Final Layer Activations into Segmentations

The final layer of the 3D CapsNet is composed of one capsule channel that learns to activate capsules within the segmentation target and deactivate them outside the target. Activation of a capsule is determined by the length of its pose vector, which is a number between 0 and 1. The ground truth segmentations are coded similarly: pixels outside and inside the segmentation target are respectively coded by 0 and 1.

During testing, the length of the final layer's pose vectors is thresholded at T:

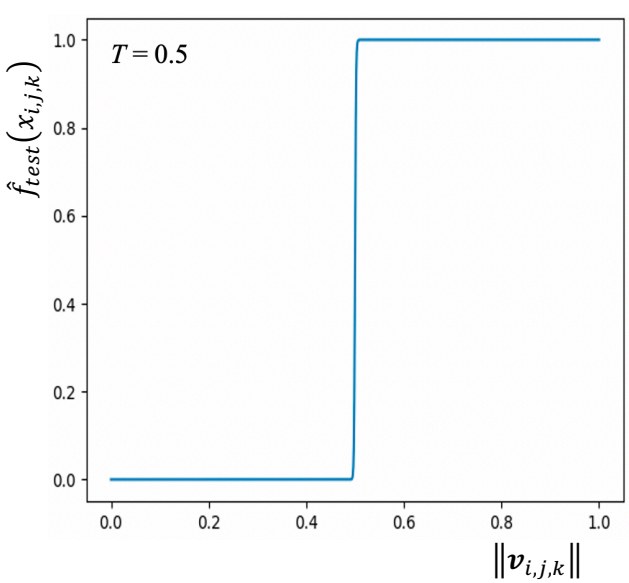
$$\hat{f}_{test}(x_{i,j,k}) = \begin{cases} 0, & \|\boldsymbol{v}_{i,j,k}\| < T \\ 1, & \|\boldsymbol{v}_{i,j,k}\| \geq T \end{cases} \tag{1}$$

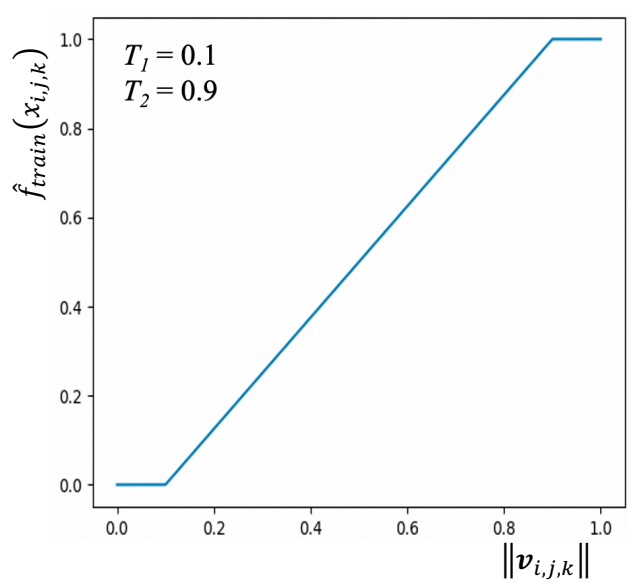
where $\hat{f}(x_{i,j,k})$ is the prediction of the CapsNet for the input voxel $x_{i,j,k}$ and $\|\boldsymbol{v}_{i,j,k}\|$ is the length of the final layer's pose vector $\boldsymbol{v}_{i,j,k}$ at the location (*i,j,k*) of the MRI volume (please note that $\boldsymbol{v}_{i,j,k}$ is itself a function of $x_{i,j,k}$, the function being the entire CapsNet that takes $x_{i,j,k}$ as the input and gives $\boldsymbol{v}_{i,j,k}$ as the output).

During training, the length of the final layer's pose vector and each location (*i,j,k*) undergo a piecewise linear transform as follows:

$$\hat{f}_{train}(x_{i,j,k}) = \begin{cases} 0 & , \quad \|\boldsymbol{v}_{i,j,k}\| < T_1 \\ \dfrac{\|\boldsymbol{v}_{i,j,k}\| - T_1}{T_2 - T_1} & , \quad T_1 \leq \|\boldsymbol{v}_{i,j,k}\| < T_2 \\ 1 & , \quad \|\boldsymbol{v}_{i,j,k}\| \geq T2 \end{cases} \tag{2}$$

If we set $T = 0.5$ $T_1 = 0.1$ and $T_2 = 0.9$, we get the following diagrams for $\hat{f}_{test}(x_{i,j,k})$ and $\hat{f}_{train}(x_{i,j,k})$ as functions of $\|\boldsymbol{v}_{i,j,k}\|$:





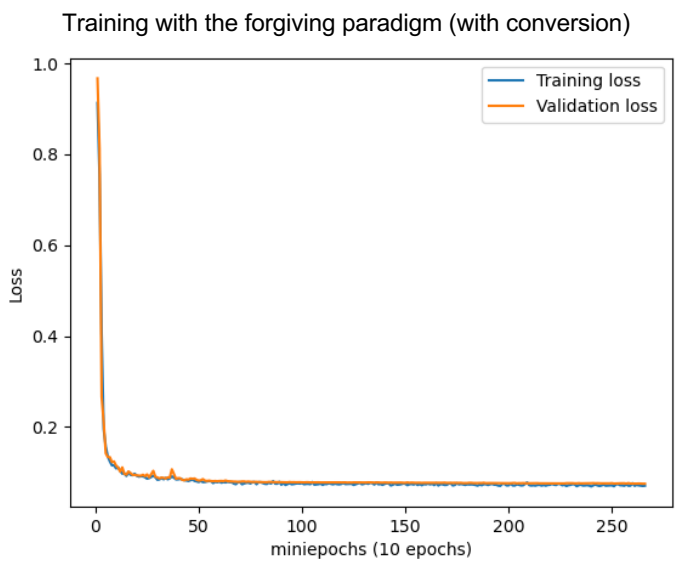
During training, the piecewise conversion (formula 2) enables a *forgiving paradigm* for the length of the final layer's pose vectors: if the length of the vector is more than 0.9 for a voxel inside the segmentation target, the loss for that voxel would be zero. Intuitively, a pose vector with a length more than 0.9 for a voxel inside the segmentation target is considered "good enough", so the training algorithm should not try to perfect the length of this vector to 1. Similarly, a pose vector with a

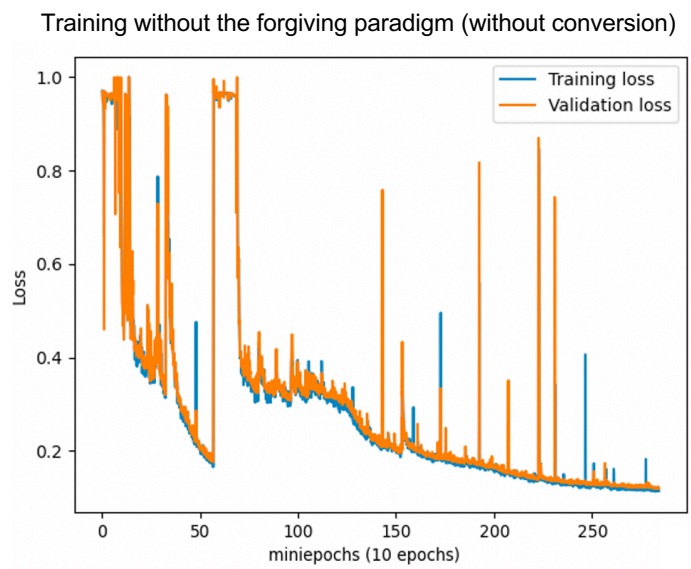
length less than 0.1 is considered good enough for a voxel outside the segmentation target, so the training algorithm should not try to perfect the length of this vector to 0. This forgiving training paradigm makes the training of CapsNet stable because this paradigm does not try to perfect the length of the pose vectors of the final layer to 0's and 1's. In contrast, if a training paradigm tries to perfect the length of the pose vectors to 0's and 1's, that training paradigm becomes unstable because the pose vectors can assume a length close to 0 or 1, but not exactly 0 or 1. Remember that the pose vectors are generated by the *squash function*,[10] which cannot generate vectors with a length equal to 0 or 1:

$$\boldsymbol{v}_{i,j,k} = squash(\boldsymbol{s}_{i,j,k}) = \frac{\boldsymbol{s}_{i,j,k}}{\|\boldsymbol{s}_{i,j,k}\|} \cdot \frac{\|\boldsymbol{s}_{i,j,k}\|^2}{1 + \|\boldsymbol{s}_{i,j,k}\|^2} \tag{3}$$

where $\boldsymbol{s}_{i,j,k}$ is the total input to the final layer capsule at the location (i,j,k), and $\boldsymbol{v}_{i,j,k}$ is the pose vector of the final layer capsule at that location.

Our experiments show that training with the forgiving paradigm is more stable and leads to faster convergence. When we did not convert the length of the pose vector $\|\boldsymbol{v}_{i,j,k}\|$ using the conversion function (formula 2), CapsNet training became unstable. Here we show the evolution of the training set and the validation set losses during 10 epochs of training, with and without the forgiving paradigm:

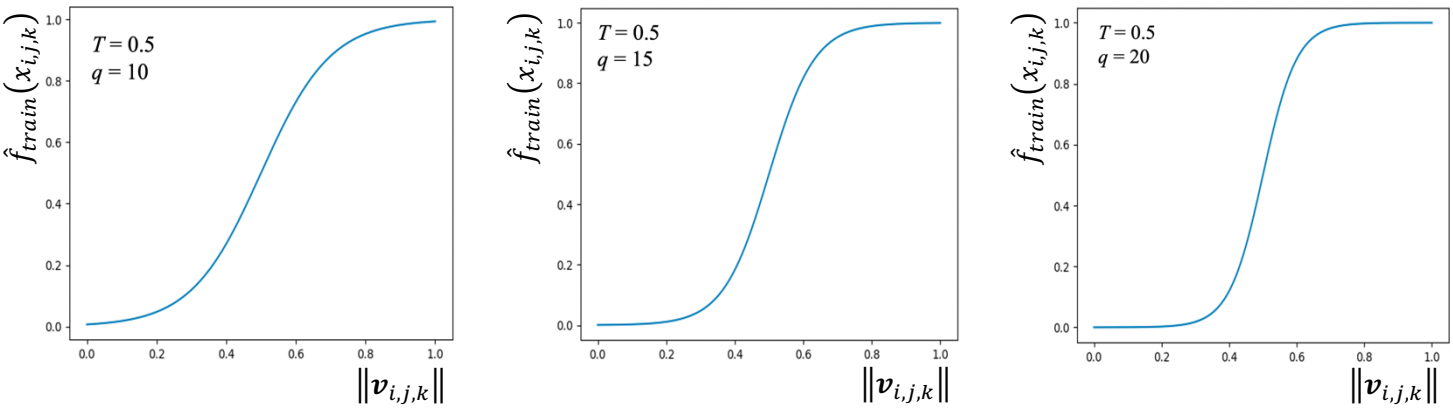




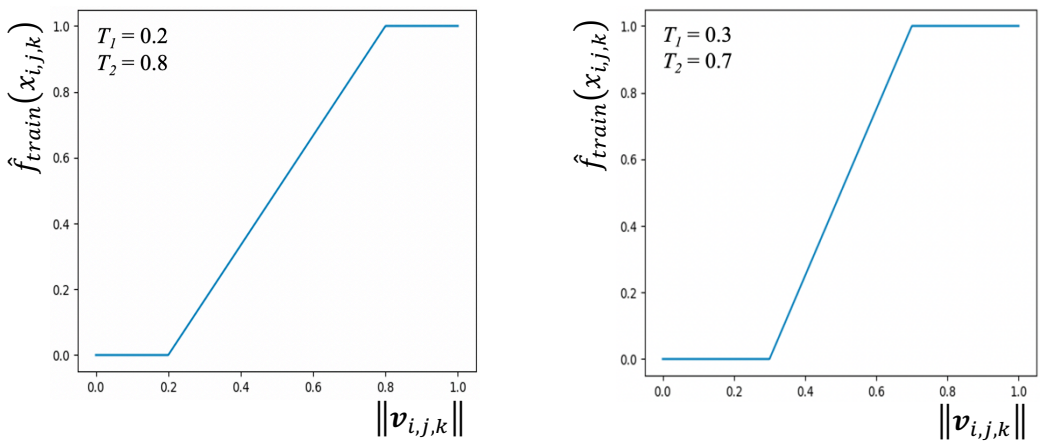
We additionally searched for the optimal conversion functions. The piecewise linear function led to the most stable training and fastest convergence. Here we describe other functions that we studied (together with their plots) so that other groups would be aware of these conversion functions that we think are suboptimal for this task:

$$\hat{f}_{train}(x_{i,j,k}) = sigmoid\left(q.\left(\|\boldsymbol{v}_{i,j,k}\| - T\right)\right) = \frac{1}{1 + e^{-q.(\|\boldsymbol{v}_{i,j,k}\| - T)}} \tag{4}$$

We set $T = 0.5$ and tried different values for q (10, 15, and 20):



We also examined the piecewise conversion function (formula 2) with values for T_1 and T_2 other than 0.1 and 0.9:



None of these conversion functions was as effective as the piecewise function with $T_1 = 0.1$ and $T_2 = 0.9$ in improving the stability and convergence of CapsNet training.

Appendix 6: Findings Agreeing Pose Vectors

Let's assume the previous capsule layers has six capsule channels, each outputting the vote vector of a part (v_1 to v_6). To find the vote vectors that agree, we first compute the vector summation of all vote vectors (v):

$$v = \sum_i v_i$$

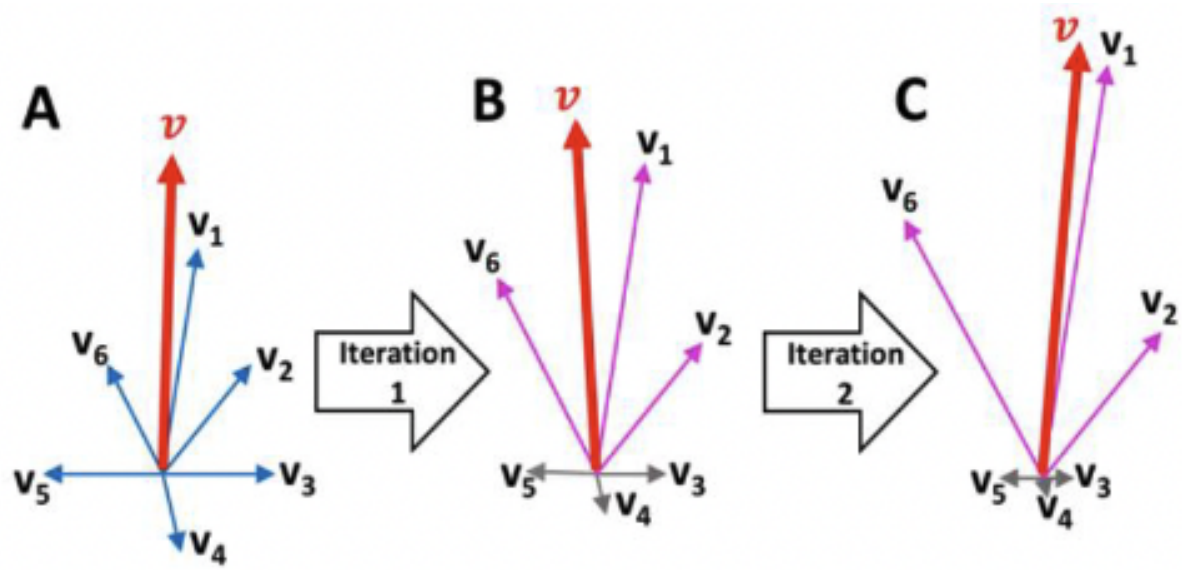
Then, we compute the inner products between each vote vector v_i and the sum v, yielding weights for each vote vector w_i:

$$w_i = v_i . v$$

Please note that each w_i is a scalar. Next, we re-compute the vector sum v using the weighted average of the vote vectors using weights w_i computed in the previous step :

$$v = \sum_i w_i \, v_i$$

This process is often repeated for three iterations. The number of iterations is a hyperparameter that should be set between capsule layers. This whole process increases the weights of the vectors that align with the sum (v_1, v_2, and v_6 in this example) and decreases the weights of the vectors that do not align with the sum (v_3, v_4, and v_5 in this example).



Appendix 7: Training hyperparameters

Training set size (MRI volumes):	3199
Validation set size (MRI volumes):	117
Test set size (MRI volumes):	114
Training batch size (MRI volumes):	4
Training mini-epoch size:	30 batches: during training, the validation set loss was computed after each mini-epoch
Training epochs:	50
Optimizer:	Adam
Optimizer hyperparameters:	$\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$
Initial learning rate:	0.002
Minimal learning rate:	0.0001
Learning rate scheduling:	Dynamic (via monitoring the validation set loss during training): Learning rate was decreased by half if the validation set loss did not improve over 10 mini-epochs

References:

1. Fischl B. FreeSurfer. *NeuroImage* 2012;62:774–81.

2. Clerx L, Gronenschild EHBM, Echavarri C, et al. Can FreeSurfer Compete with Manual Volumetric Measurements in Alzheimer's Disease? *Curr Alzheimer Res* 2015;12:358–67.

3. Ochs AL, Ross DE, Zannoni MD, et al. Comparison of Automated Brain Volume Measures obtained with NeuroQuant and FreeSurfer. *J Neuroimaging Off J Am Soc Neuroimaging* 2015;25:721–7.

4. Yaakub SN, Heckemann RA, Keller SS, et al. On brain atlas choice and automatic segmentation methods: a comparison of MAPER & FreeSurfer using three atlas databases. *Sci Rep* 2020;10:2837.

5. Despotović I, Goossens B, Philips W. MRI Segmentation of the Human Brain: Challenges, Methods, and Applications. *Comput Math Methods Med* 2015;2015:e450341.

6. Dale AM, Fischl B, Sereno MI. Cortical surface-based analysis: segmentation and surface reconstruction.

7. Ségonne F, Dale AM, Busa E, et al. A Hybrid Approach to the Skull Stripping Problem in MRI.

8. Hinton GE, Sabour S, Frosst N. Matrix capsules with EM routing. In: *International Conference on Learning Representations 2018*.

9. LaLonde R, Xu Z, Irmakci I, et al. Capsules for biomedical image segmentation. *Med Image Anal* 2021;68:101889.

10. Sabour S, Frosst N, Hinton GE. Dynamic routing between capsules. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Red Hook, NY, USA: Curran Associates Inc.; 2017:3859–69.