CS230

Post-stroke Lesion Detection using ATLAS Dataset

Andrew Zhang
Stanford University
azhang97@stanford.edu

Chenyao Yu
Stanford University
chenyaoy@stanford.edu

Helen Jiang
Stanford University
helennn@stanford.edu

Abstract

Hundreds of thousands of people in the United States suffer from a stroke every year. A vast majority of stroke survivors suffer long-term physical, emotional, and cognitive problems, highlighting the importance of early treatment for proper recovery. Accurate identification of stroke lesions from a victim's head MRI slices can help doctors in treatment and research for post-stroke recovery, but currently this requires manual segmentation by medical experts. Recently, deep learning has helped automate lesion identification and a well-trained system could provide quick lesion identification and produce datasets for further stroke research. In this paper, we explore the 2D U-Net architecture with a variety of improvements on the new ATLAS dataset in the domain of post-stroke lesion detection.

1 Introduction

We hope to tackle the problem of automatic identification of lesions from MRI slices of the brains of stroke victims. The current gold standard for identifying the location of the post-stroke lesions is through manual segmentation, which takes trained experts an hour per MRI scan. As such, existing stroke neuroimaging datasets are prohibitively small; automation of lesion identification would allow researchers to feasibly compile large neuroimaging datasets, and advance research to improve post-stroke recovery. We are looking to improve the efficacy of automated identification of the lesions using a larger dataset and deep learning techniques.

The input to our algorithm is a single MRI scan: a 232×196 grayscale image. We use a neural network to output a prediction for lesion segmentation: a 232×196 binary image. While we used a conv-deconv network as our baseline model, our research focused on exploring the effects of modifications to a 2D U-Net model architecture.

2 Related work

[1], [2], and [3] gives the context and details for brain MRI segmentation using deep learning, where they all use Convolutional Neural Networks for segmentation.

[4] provides a short background of many approaches to medical image segmentation, including many methods that do not involve any neural networks, even though CNN's are shown to perform better than those methods that do not involve deep learning on its own.

In [7], we are given the benefits of BatchNorm, which we incorporate into our model to achieve better results. Batchnorm allows higher learning rates, reduces the dependency on initialization, and provides some sort of regularization.

[5] introduces 3D U-Net which learns from sparsely annotated volumetric images for volumetric segmentation, which works for both semi-automated and automated segmentation cases.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

3 Dataset and Features

For our task, we are using the provided ATLAS (Anatomical Tracings of Lesions After Stroke) Dataset [9]. ATLAS contains 229 MRI scans from 220 distinct patients, collected from 9 sites. Each scan includes slices of the head MRI, ground truth segmentation masks, as well as meta-labels for each scan. Some meta-labels include the number of lesions, the type of stroke, and the primary stroke location. There are a total of 43281 MRI slices. A sample slice is seen in 1.

The dataset was split 70/30 by slices into our train/dev set. Each scan is preprocessed by cropping to size 232×196. We used data augmentation like random flips and distortions to increase the amount of training data and reduce overfitting.

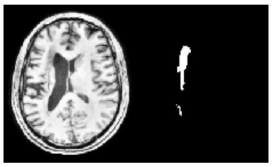


Figure 1: An example slice from the ATLAS Dataset

4 Methods

4.1 Baseline: Conv-Deconv Network

The baseline is a Conv-Deconv which consists of a convolutional encoder part comprising two stacks of 2D convolution with ReLU layer, max pooling, and dropout. This is followed by a fully-connected layer, another dropout layer, and a final fully-connected layer. This part of the network allows the network to learn higher level features from the input image.

The second deconvolution decoder part essentially rescales the output back to the original input size to produce the high-resolution mask predictions. This part is comprised of a fully-connected layer followed by a dropout layer and another fully-connected layer with dropout, and then by two upsampling and 2D deconvolution layers.

4.2 2D U-Net

The 2D U-Net model is a network that combines a contracting path that learns higher level features with an expansive path that allows for the network to output a high-resolution segmentation map. In these successive layers, the pooling operators are replaced by upsampling operators (as seen in fig. 1), which increases the resolution of the output. It is essentially a more complex Conv-Deconv model.

We chose to explore the 2D U-Net model for the domain of post-stroke lesion detection because it can capture information from multiple levels of the input image of the MRI scans directly into the output. We also wanted to see if this proven model would generalize well to this new dataset and problem space of predicting post-stroke lesions.

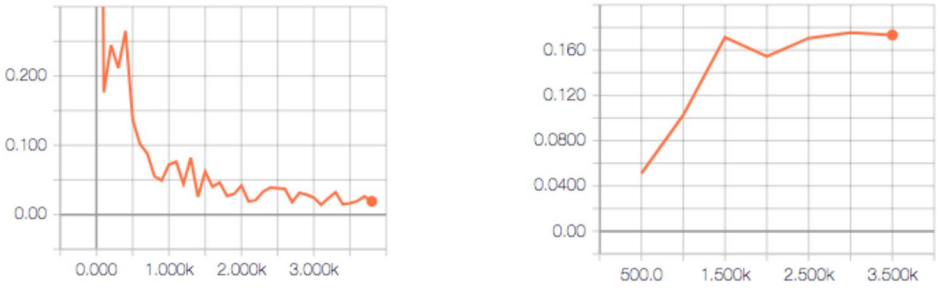
We also experimented with adding BatchNorm [7] to our models, a technique popular today for accelerating convergence and improving results. The BatchNorm paper was published very shortly before [6] and so although it was used in the 3D U-Net architecture, it was not used in the former 2D architecture. We added BatchNorm before the ReLU activation in our 2D convolution layers.

Since we were given patient metadata for each MRI scan, we wanted to test the performance of integrating metadata into our input. Our input included the number of left and right, cortical and subcortical strokes that the patient suffered. The metadata values were appended to the input scan as 4 additional channels, so as to make the information available to all pixels.

Each slice contains varying numbers of lesion-positive pixels, with many having few-to-none. As such, the network has greater difficulty increasing its recall. We use weighted cross-entropy between the logits for our prediction and the ground truth segmentation, and experimented with the weight that should be used to calculate our loss. Theoretically, a larger positive weight encourages a decrease in the false negative count and increases recall.

–	Vanilla	w/ BatchNorm	w/ Metadata + BatchNorm
Baseline	0.182	0.288	—
2D U-net	0.147	0.173	0.202

Table 1: Dice coefficients of various models



(a) Training Loss (b) Dice Coefficient

Figure 2: Baseline Conv-Deconv Model (batch size 100)

5 Experiments/Results/Discussion

5.1 Evaluation

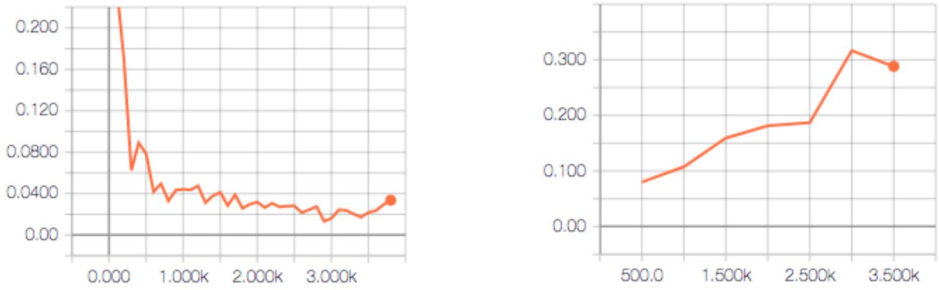
We evaluate our segmentation performance by comparing the dice coefficient, the most common metric for evaluating segmentations of MRI imaging data.

$$DICE = \frac{2TP}{2TP + FP + FN}$$

where TP, FP, FN represent the true positive, false positive, and false negative pixel counts of the lesion mask predicted by the algorithm compared to the ground-truth lesion mask.

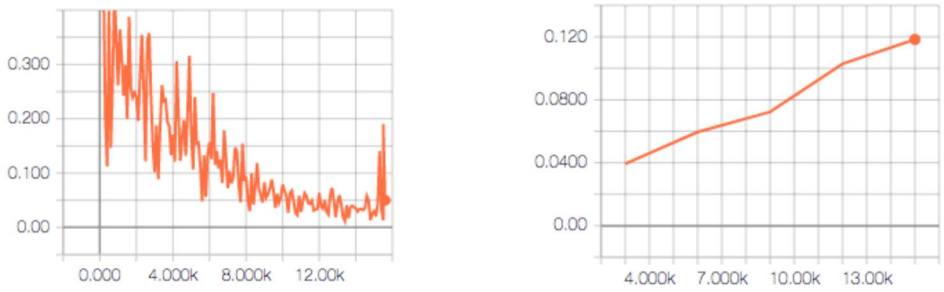
5.2 Results & Discussion

The baseline conv-deconv model performs decently well with a dice coefficient of 0.182 after 30 epochs. The addition of batch normalization makes a noticeable increase in performance, resulting in a dice coefficient of 0.288 after 30 epochs. This makes sense due to the benefits that batch normalization gives us, as described in [7]. Batch normalization is a technique applicable to a wide variety of problems and domains and we have shown that incorporating it in this relatively simple model can be very beneficial for working with segmenting imaging data. From the dice coefficient



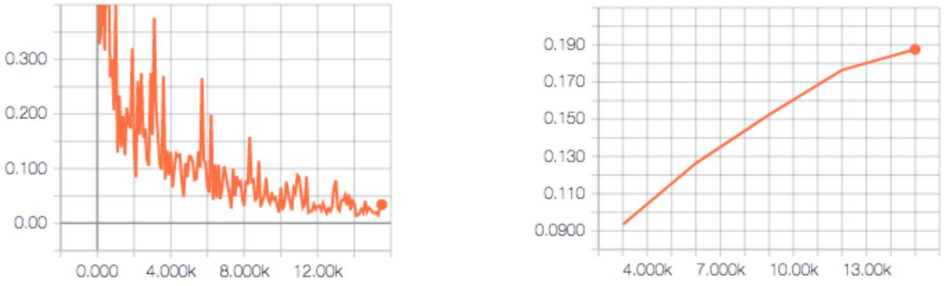
(a) Training Loss (b) Dice Coefficient

Figure 3: Baseline Conv-Deconv Model w/ BatchNorm (batch size 100)



(a) Training Loss (b) Dice Coefficient

Figure 4: 2D UNet Model (batch size 25)



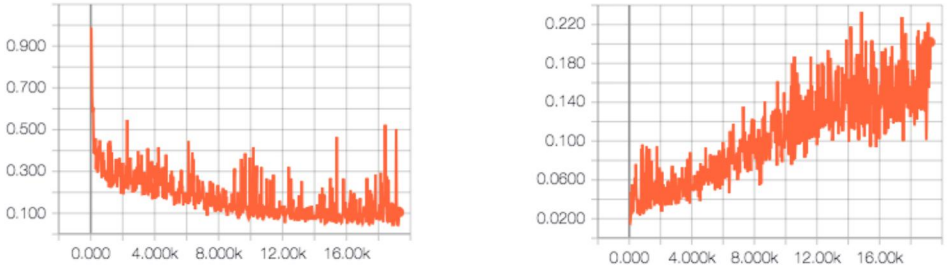
(a) Training Loss (b) Dice Coefficient

Figure 5: 2D UNet Model w/ BatchNorm (batch size 25)

curves, we also see that early stopping could be beneficial with this model, as the dice coefficient peaks or plateaus before the end of 30 epochs.

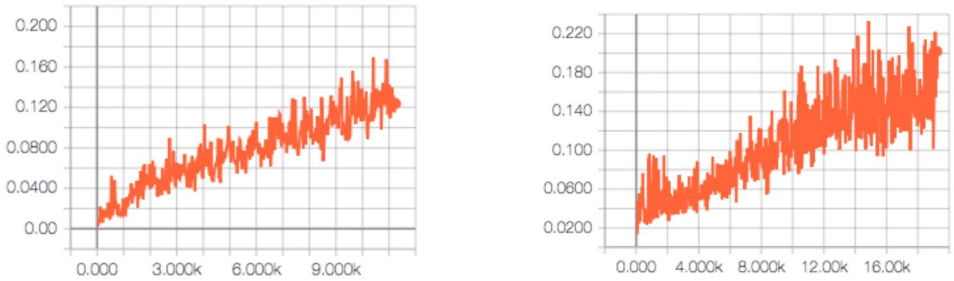
The 2D U-Net seems to under-perform the conv-deconv model, with a dice coefficient of 0.147. Adding batch normalization only increased this to 0.173, which is still lower than the baseline model without batchnorm. There are a few reasons for the disappointing results. First, from Figures 3 & 4, we see that after 30 epochs, the performance has not plateaued and we would expect better results from training the model with more epochs, which we did not have time to do. The U-Net model took around 10 hours to train 30 epochs.

Another consequence of such long training times is the difficulty of iterating and tuning hyperparameters. As a result, we did not have the time to extensively explore the effects of many hyperparameters on the performance and instead focused on changing parts of the model. Because it is a fairly complex model, we would expect the U-Net to be sensitive to hyperparameters that were carefully tuned in the original paper, which is the second reason for the poor performance.



(a) Training Loss (b) Dice Coefficient

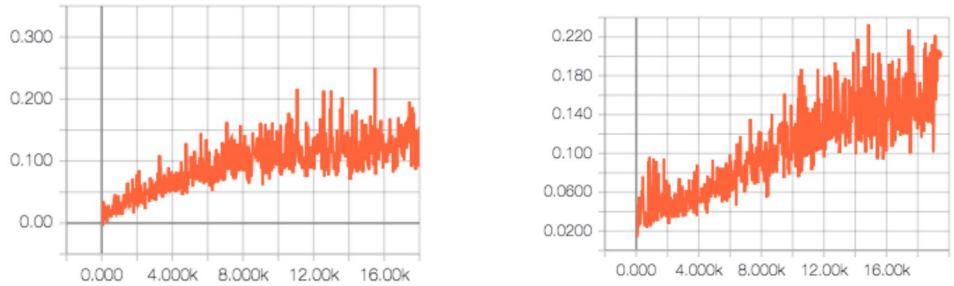
Figure 6: 2D UNet Model w/ Metadata & BatchNorm (batch size 25)



(a) Original Dice

(b) Thinned Dice

Figure 7: Thinned vs Original 2D UNet Model w/ Metadata & Batchnorm



(a) Dice, weight = 50

(b) Dice, weight = 100

Figure 8: Effect of weight in weighted cross-entropy loss

Thinning our network by halving the number of filters at each stage of our U-Net architecture helped speed up epochs by more than double. The thinned network even outperformed the larger architecture at equal amounts of training, suggesting that the model likely had difficulty learning the large number of parameters in our original network.

Finally, as expected, using a larger positive weight in our weighted cross-entropy helped increase our recall, which helped the network learn a higher DICE score as well.

We were also limited by GPU memory, which accounts for the much noisier loss curves of the U-Net model because we had to reduce the batch size from 100 to 25.

6 Conclusion & Future Work

The baseline conv-deconv model performed much better than the 2D U-Net model on our task. While this is surprising, it shows how much the 2D U-Net model, a much more complex model, depends on computation resources and hyperparameter tuning. We also showed that incorporating BatchNorm can yield a big performance boost for our task and including metadata features can be promising as well.

Because we have volumetric medical imaging data, we hope to implement the 3D U-Net from [5] in the future and apply it to our problem. 3D U-Net has been shown to work better than the 2D architecture for volumetric data in related domains, so it would be an interesting model to try.

Another issue is the amount of time it takes to train the U-Net models. The 2D U-Net model took us 10 hours to train 30 epochs, and as mentioned in the discussion section, we would have benefited from training it on even more epochs. We expect the 3D U-Net to take even longer to train. For future work, we would want to train the U-Net models longer, until performance plateaus. We would also want to perform a hyperparameter search, which we did not have time to do.

Later on, we would also try to improve performance by adding more features, layers, and dimensions. We can experiment with using different clustering methods and Random Markov Field models, which are also popular methods for segmentation [4].

7 Contributions

Andrew focused on the metadata and cross-entropy weighting additions to the conv/deconv and U-Net model. Chenyao focused on the batchnorm and data augmentation. Helen focused on the drafts of the papers and poster. All team members worked together on brainstorming and the final draft.

References

[1] Z. Akkus, A. Galimzianova, A. Hoogi, D. L. Rubin, and B. J. Erickson. (2017) Deep Learning for Brain MRI Segmentation: State of the Art and Future Directions.

[2] C. Bermudez, A. J. Plassardb, L. T. Davisc, A. T. Newtonc, S. M. Resnickb, B. A. Landman. (2018) Learning Implicit Brain MRI Manifolds with Deep Learning.

[3] N. Bien. (2018) Don't Just Scan This: Deep Learning Techniques for MRI.

[4] D. L. Pham, C. Xu, and J. L. Prince. (2007) Current Methods in Medical Image Segmentation.

[5] O. Cicek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and Olaf Ronneberger. (2016) 3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation.

[6] Olaf Ronneberger and Philipp Fischer and Thomas Brox. (2017) U-Net: Convolutional Networks for Biomedical Image Segmentation.

[7] Sergey Ioffe and Christian Szegedy. (2015) Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.

[8] Sook-Lei Liew. (2017) The Anatomical Tracings of Lesions After Stroke (ATLAS) Dataset