Spine Segmentation using U-Net

Andrew Selvia (014547273)
Department of Software Engineering
San José State University
San José, California
andrew.selvia@sjsu.edu

***Abstract*—Image segmentation is an important first step of many research projects which apply computer vision techniques to medical imagery. This project demonstrates how U-Net [1], a popular convolutional neural network (CNN) architecture, can be used to isolate the spine from a posteroanterior X-ray. Inspired by a similar paper by Horng, Kuok, Fu, Lin, and Sun which performs spine segmentation in service of learning Cobb angles for scoliosis detection [2], this project replicates the image segmentation task in a novel way while also setting the stage for future exploration of the advanced methods described in that paper.**

***Index Terms*—machine learning, computer vision, image segmentation, neural networks**

I. Introduction

Scoliosis is a common spinal deformity affecting around 3 percent of the U.S. population [3]. For decades, advocates have promoted testing children starting in grade school to detect spinal curvature as they grow. These efforts are justified since no cure exists but early, accurate, and frequent testing can prevent unnecessary suffering later in life.

Obviously, innovation which might accelerate, standardize, and reduce the cost of scoliosis detection would have a large impact. For exactly these reasons, computer vision is a compelling research area which could help millions of people.

The ultimate goal of my research is to detect Cobb angles from X-rays for scoliosis detection. Though that remains out-of-reach for now, I have proven spine segmentation is possible through the use of TensorFlow and the U-Net CNN architecture.

II. Implementation

A. Code

Similarly to the original paper [2], U-Net was chosen as the neural network architecture for this project. A quick literature study demonstrated its wide adoption for medical imaging tasks, lending confidence that this approach is well-founded.

The code for this project is hosted on GitHub [4]. The code is inspired heavily by the U-Net implementation created for the official Keras code examples by François Chollet [5] but has been adapted to train on the X-ray dataset rather than the original Oxford-IIIT Pet Dataset [6]. Importantly, the original implementation is in the form of a Jupyter notebook; this had to be converted into a standalone Python application which could be run via a SLURM batch job on the SJSU HPC. The ReLU activation function, RMSprop optimizer, and Sparse Categorical Crossentropy loss function remain in use exactly as in the original implementation.

B. Data

The authors of the paper that inspired this project [2] kindly shared their data when contacted via email. It consists of 481 training images and 128 testing images. These raw X-rays vary in size but all provide a posteroanterior view of spines. The degree of the curves represented varies from mild to severe.

Some images exhibit characteristics which present challenges to learning. Specifically, the images contain visual artifice in the form of necklaces, organs, and surgically-implanted Harrington rods used to stabilize the spine from further curvature. In addition, there is the complication of blurriness which can at times render the spine imperceptibly blended behind visual noise.

C. Pre-processing

Unfortunately, the provided masks used for image segmentation are in MATLAB format, thereby proving incompatible with this project's Python training implementation. Thus, each of the 609 images was hand-labeled into a trimap [7] using the Mask AI program created by Topaz Labs. It took 6 hours to label them all.

The masks were created with red, green, and blue (RGB) regions where green represents the high-confidence segment of the image to learn, red represents the segment to crop, and blue represents the border between the two. You can see an example trimap in Figure 2.

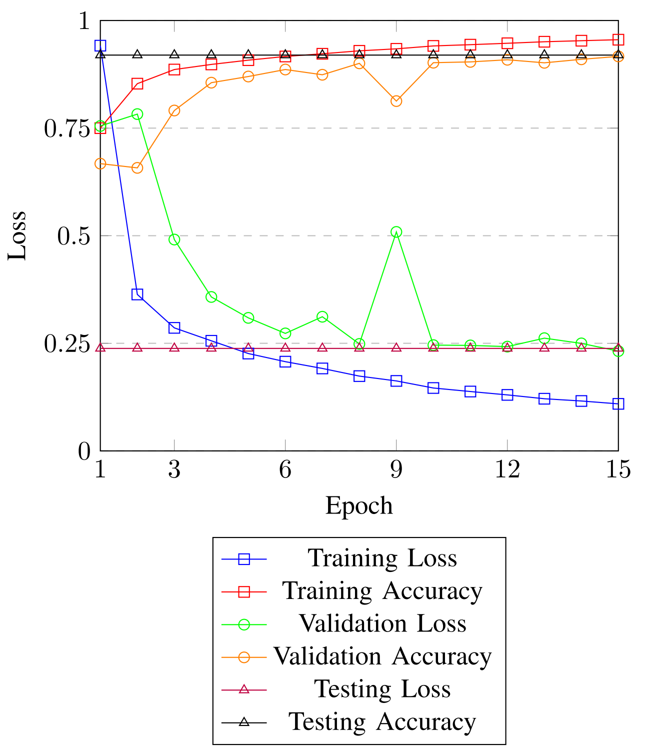
The hand-labeled RGB trimaps required further processing to be used in the U-Net implementation leveraged for this project. It expects the trimaps to be constrained to only the three darkest shades of black (1, 2, and 3 in RGB parlance). Therefore, the pixels of each RGB trimap were mapped into that colorspace.

D. Training

Training occurred over 15 epochs. Higher numbers of epochs proved only to overfit the training data.

A critical discovery was the impact the batch size played in the accuracy of the model's predictions. When left at its initial value of 32, the model struggled to learn how to segment spines due to the drastic reduction in total training data between the pets dataset containing 7,500 images and the X-ray dataset containing just 481. When batch size was reduced to 3, the model's accuracy rose dramatically.

Fig. 1: Loss and Accuracy



Performing training on the HPC became a top priority upon discovery of the time required to train the model locally with just a CPU. Training the model for 15 epochs took 25 minutes locally without a GPU. Training it on a GPU node in the SJSU HPC yielded a much more reasonable training time of 2.5 minutes; a 10x improvement.

E. Testing

The trained model was evaluated over all 128 testing images plus one extra that will be explained in section III. Loss and accuracy are reported in Figure 1 for each data subset. Notably, the validation set clearly tracked the out-of-sample testing performance during training.

F. Post-processing

Once the model was trained and evaluated against the testing data, it was used to predict segmentation trimaps for each of the testing images. Of course, the predicted trimaps were normalized to the network's output size of 150x150 pixels. These square outputs look distorted when compared to the original X-rays, so each was resized to the original dimensions of its respective testing image.

While input X-rays could now be compared with their predicted trimaps, it became obvious that interlacing the two would yield a superior visualization. By leveraging ImageMagick techniques [8], each of the testing image trimaps was interlaced with its input image. In between, each trimap had to be converted from the predicted representation using three distinguishable colors (black, gray, and white) to an intermediate trimap representation using the three darkest shades of black (1, 2, and 3 in RGB coordinates). This conversion produces a trimap that is compatible with ImageMagick, though it is impossible to distinguish between the shades of black visually. Each of the final interlaced images was saved to disk.

III. Results

The two metrics used to evaluate the model's ability to learn are loss and accuracy. The predictions it made yielded a loss rate of 24% and an accuracy of 92% when compared against the hand-labeled trimaps. These metrics are likely insufficient for real-world application when compared against human performance. While a metric tracking human performance at spine segmentation could not be identified, I expect it to be higher than the model's performance. Keep in mind, the ultimate goal of doing Cobb angle prediction has the potential for more competition as inter- and intra-person performance varies between 3–10% [9] due to factors including image quality, professional experience, and time pressure. That problem retains promise for further research.

The following figures visualize some unique challenges the model faced. As you can see in Figure 2, it successfully segments the spine through its extreme curvature, even surviving its encounter with the edge. Figure 3 shows how it learns to segment the spine despite the presence of what appear to be Harrington rods surgically implanted to stabilize the spine; though, the predicted mask does appear to have included the upper screws of that mechanism. Figure 4 and Figure 5 both demonstrate failure cases which would prevent Cobb angle prediction. In Figure 4, it seems the obscuring organ results in a gap in the mask. To human eyes, the organ increases contrast, thus actually making it easier to identify the edges of the spine, but it seems the model cannot make that same conclusion. The gap in the predicted mask for Figure 5 may also have been affected by an organ, though the necklace may have had a hand to play in the inaccuracy as well.

Finally, I would like to draw special attention to the predicted mask shown in Figure 6. The model has segmented the spine quite well in this case. It is especially meaningful to me to see this result as it is, in fact, my own spine.

IV. Conclusion

The ability to segment the spine from a posteroanterior X-ray with 92% accuracy is an encouraging first step toward the ultimate goal of Cobb angle prediction. Next semester, I intend to build upon this work in the Deep Learning course by training a model to detect the individual vertebrae in segmented spine images. For now, I conclude with gratitude to Dr. Mahima Agumbe Suresh for educating me in the fundamentals of machine learning.

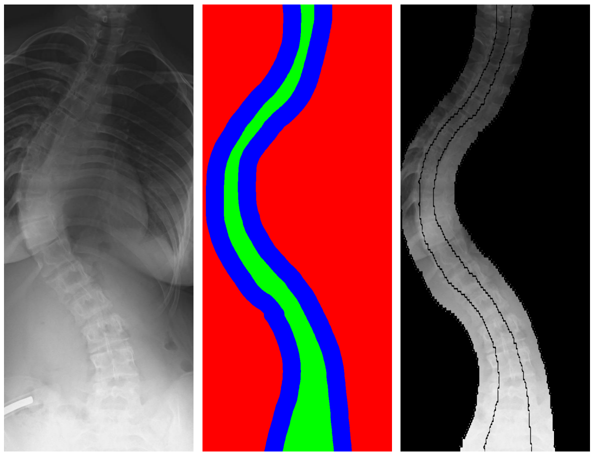


Fig. 2: Edge-intersecting spine segmentation

(a) Input (b) Trimap (c) Output

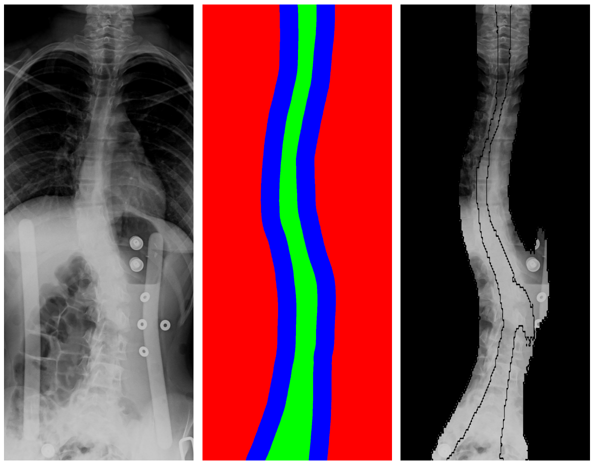


Fig. 3: Harrington rods slightly impact spine segmentation

(a) Input (b) Trimap (c) Output

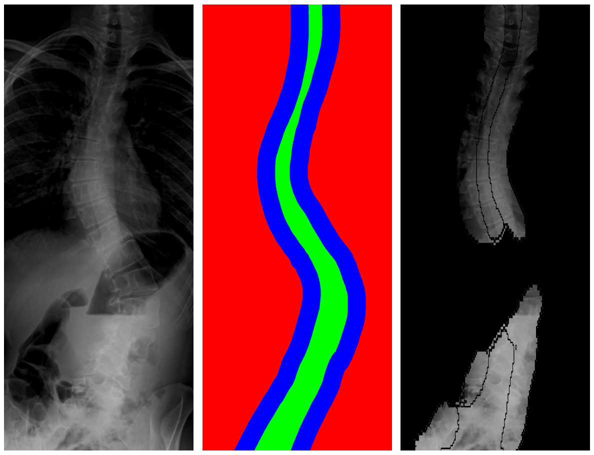


Fig. 4: Obscuring organ results in failed spine segmentation

(a) Input (b) Trimap (c) Output

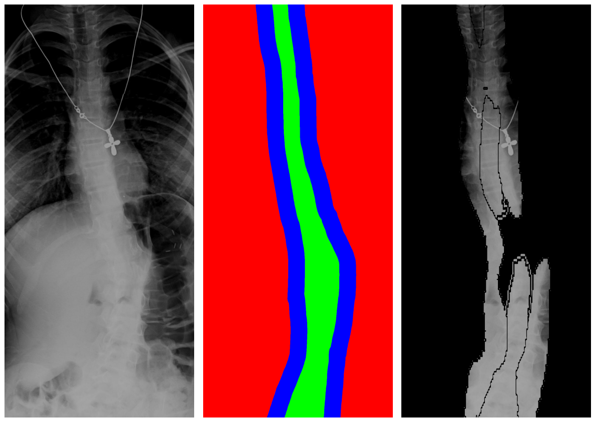


Fig. 5: Necklace impacts spine segmentation

(a) Input (b) Trimap (c) Output

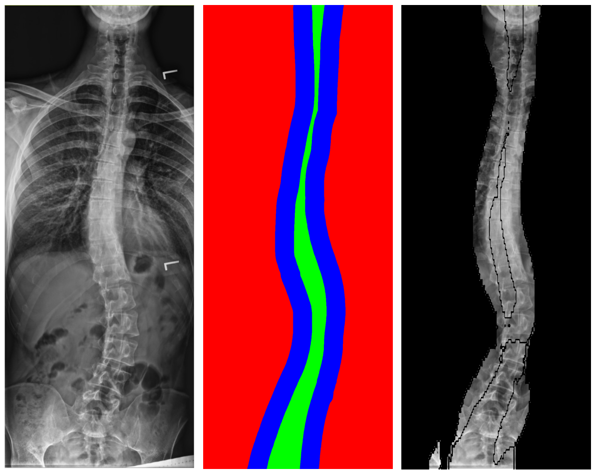


Fig. 6: The author's own spine segmentation

(a) Input (b) Trimap (c) Output

REFERENCES

[1] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015. https://arxiv.org/abs/1505.04597.

[2] M.-H. Horng, C.-P. Kuok, M.-J. Fu, C.-J. Lin, and Y.-N. Sun, "Cobb angle measurement of spine from x-ray images using convolutional neural network," 2019. https://doi.org/10.1155/2019/6357171.

[3] N. S. Foundation, "Scoliosis media and community guide," 2009. https://www.scoliosis.org/nsf2/wp-content/uploads/2015/06/ScoliMediaGuide_9June3.pdf.

[4] A. Selvia, "spine-segmentation," 2020. https://github.com/AndrewSelviaSJSU/spine-segmentation.

[5] F. Chollet, "Image segmentation with a u-net-like architecture," 2020. https://keras.io/examples/vision/oxford_pets_image_segmentation/.

[6] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar, "The oxford-iiit pet dataset," 2012. https://www.robots.ox.ac.uk/~vgg/data/pets/.

[7] T. Labs, "Learn better and faster image masking with topaz mask ai," 2020. https://www.youtube.com/watch?v=u2UBvsZv954.

[8] M. Setchell, "Interlacing an image and a trimap," 2017. https://stackoverflow.com/a/41778637/6073927,.

[9] M. M. Watanabe Kota, Aoki Yoshimitsu, "An application of artificial intelligence to diagnostic imaging of spine disease: Estimating spinal alignment from moiré images," *Neurospine*, vol. 16, no. 4, pp. 697–702, 2019.