C137 Project: HuBMAP + HPA - Hacking the Human Body Kaggle Challenge

Robert Pitkin, Xinyang Zhao, Zhifei Ma

1 Introduction

With the developments of single-cell-level information processing techniques, ambitious efforts are devoted to creating a digital reference atlas of the human body.[2] As such, the task of this project is to map the dynamic distribution of human cells and give functional annotations based on a 3D organization of whole organs and thousands of anatomical structures. In the past, such work was done with manual annotations of tissue units; a process that was often extremely time-consuming and expensive. However, with the advent of deep learning techniques, many methods have been developed to automate this process without sacrificing significant accuracy. The goal of this work is to help accelerate the understanding of the relationships between cells, tissue, and organ and make contributions to the open source project which helps researchers to tackle common health problems and achieve improved longevity that benefits the entire human race. The final atlas can also be utilized to identify cells that play important roles in metabolism and significantly influence human health.

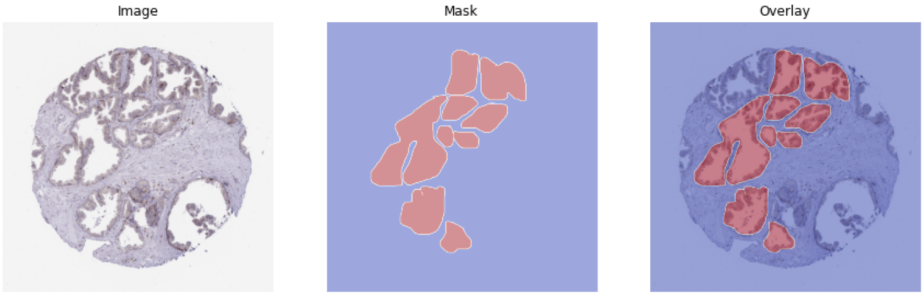


Figure 1: Example of FTUs sample overlay with mask

For this project, we apply DNN models to a new problem: segmenting multi-organ functional tissue units (FTUs). Traditionally DNN models are used to classify particular cells in a specific organ of the human body, but our goal is to evaluate various deep learning techniques and generalize them to multiple FTUs detection. More specifically, we will perform semantic segmentation on medical images to classify cells that belong to five different organs, including the kidney, prostate, large intestine, spleen, and lung. Since the solutions of the Kaggle challenge are not published, there is no official code and the problem remains "unsolved". However, we plan to solve this real-world challenge by examining a particularly popular learning technique in the field of object segmentation, the U-Net architecture. Unlike most deep learning models we learned in class, U-Net has a symmetric structure that outputs pixel-level segmentation maps through a decoder.[12] While the U-Net has proven success in single use-case organ and tissue segmentation tasks, it has

yet to be seen if the same performance levels can be reached in such a generalized domain.

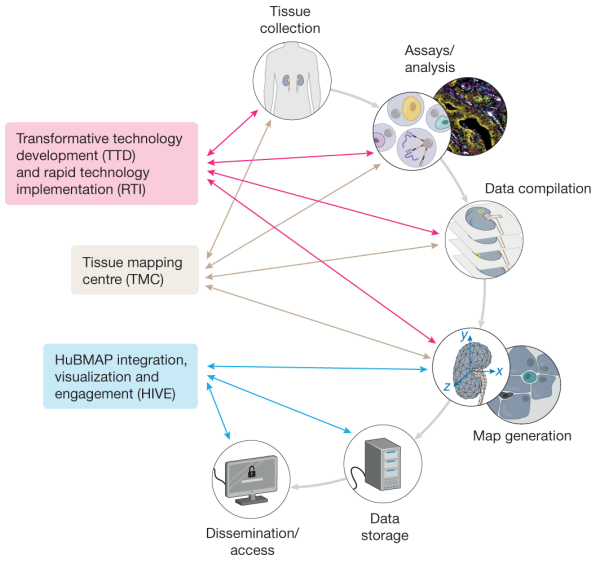


Figure 2: HuBMAP initiatives collected, processed, and shared the data

Based on the Kaggle challenge description, the inputs are 351 unique images in 3000 by 3000 pixels that contain at least one pathologically unremarkable FTUs from healthy human donors, along with labels and metadata including image ID, the organ where the sample was taken from, whether the data was provided by Human BioMolecular Atlas Program(HuBMAP) in figure 2 or Human Reference Atlas(HPA), the image height in pixels, the image width, the pixel size in micrometers, the tissue thickness of the biopsy sample, the run-length encoded location information for lossless data compression, the patient's age, and the patient's sex.

To measure the success of our project, we want to achieve at least 0.77 percent accuracy (about the top 0.9 percent of all the teams participating in the competition) for the hidden test datasets available on Kaggle. Starting with a simple and correct implementation of the U-Net structure, we hope to establish a baseline performance upon which we can continuously iterate. One possible avenue explored in previous research involved a parallel dilated convolution structure to maintain a high resolution of the output image, while still increasing the receptive field of the existing CNN structure.[11]

2 Related Work

While medical image segmentation has long been a focus of study in medicine and computer science, the application of deep learning to automated multi-organ segmentation methods has only recently been developed in the past decade.[7] Some of the earliest developments, starting in 2013, involved using stacked auto-encoders, also known as deep auto-encoders, in an effort to learn rich latent representations of objects in medical imagery.[14] Such auto-encoders produced strong results in specific use cases such as detecting lung lesions and proved the promise of applying deep learning to multi-organ segmentation, but overall were limited by their computational complexity due to the restricted usage of fully-connected layers.[10]

Auto-encoder approaches were soon overtaken by early convolutional-based segmentation approaches, particularly after the development of architectures such as ResNet and DenseNet with

the ImageNet competition.[6][8] One such approach, called DeepOrgan, employed a sliding windows approach along with a deep convolutional network in order to localize organs and segment them with a "coarse-to-fine" classification on image patches and regions.[13] However, due to the volumetric nature of medical imaging, namely CT scans, convolutional approaches were soon extended into three dimensions by using 3D convolutional networks which could properly utilize the available spatial information to localize and segment organs.[10] One notable approach by Hamidian et al. [5] used a fully-convolutional 3D network approach for region proposal along with a discriminant CNN to achieve much faster performance and better accuracy than many 2D sliding windows approaches.

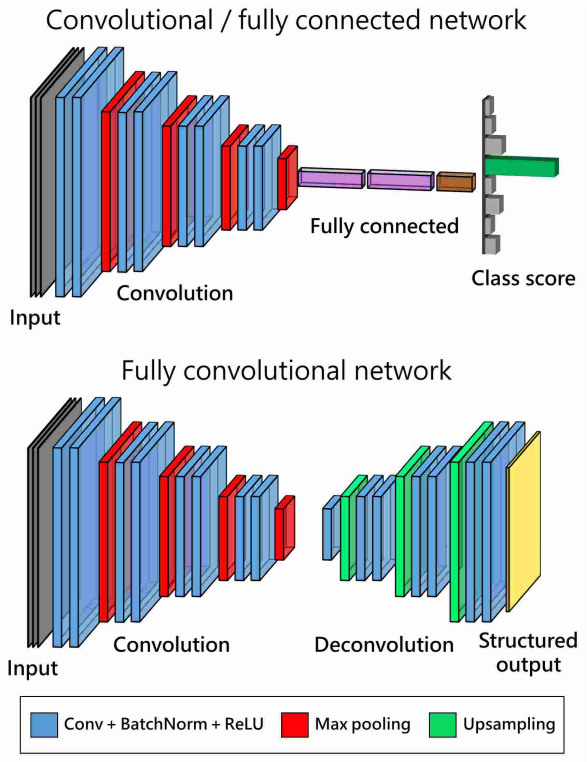


Figure 3: CNN versus FCN

Fully-convolutional networks in figure 3, also known as FCNs, quickly became the architectures of choice for object segmentation tasks with the U-Net architecture being one of the most well-known. The U-Net architecture is based on an encoding and decoding path of an image that utilizes standard convolutions in the former path and de-convolutions, or transposed convolutions, in the latter path as well as long skip connections to enable end-to-end segmentation in a single forward pass. [12] The U-Net outperformed many state-of-the-art approaches and perhaps more important, can achieve high accuracy with a relatively smaller dataset, which is crucial in a field such as medicine where acquiring data is a time-consuming and expensive process. One additional improvement made to the U-Net architecture was the addition of residual connections at every encoding and decoding step, the replacement of max-pooling layers with convolutional layers to increase the efficiency of back-propagation calculations during training, and the usage of dice loss instead of binary cross-entropy, which better suited the segmentation task. This new architecture is known as the V-Net architecture, and it excelled in the volumetric segmentation of MRI scans.

While we plan to compete in the recently completed Kaggle competition, "HuBMAP + HPA - Hacking the Human Body", there was also a recent competition from the same organization called "HuBMAP - Hacking the Kidney". All five of the winning algorithms utilized U-Net architectures, albeit with varying hyperparameters, loss functions, and more notable details.[4] However, as the new competition plans to address, these methods haven't been employed to generalize across multiple organs or remain robust across different organ datasets. Thus, our hope is that we can apply similar methods to the winning submissions for the kidney challenge, but improve them and have them successfully generalize to other types of organs.

3 Background

U-Net is a network architecture and training strategy that learns segmentation in an end-to-end setting, which means the model outputs high-resolution segmentation maps simply based on raw input images. It's commonly used to perform biomedical image segmentation for a single class given limited annotated samples.

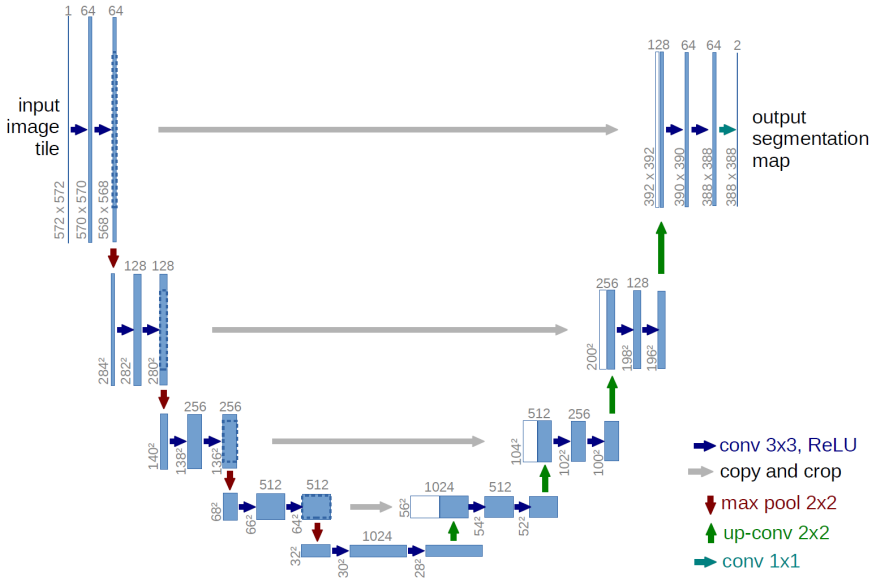


Figure 4: U-Net Architecture

Each blue box in figure 4 represents a multi-channel feature map. Similar to the other convolutional neural networks we learned in class, U-Net is consists of a large number of different convolutional operations, but it can be evenly decomposed into two reversed sections. The first half of the model is usually referred as the contraction path. It utilizes 3 by 3 convolution layers with ReLu activation function and 2 by 2 max pooling layers to reduce the size of the feature maps and increase the network depth, in order to better select the features of expected outputs. The later half of the model, known as the expansion path, up-scales back to the size of the segmentation mask through 2 by 2 up-convolution layers and 3 by 3 convolution layers. Although we lose localization information during this process, it can be recovered through the concatenation of feature maps in the previous corresponding step that is at the same level, otherwise known as a skip connection. The last 1 by 1 convolution layer maps the final output into several channels for classification, each one representing a binary mask of the class in the image. For the scope of

this project, there is only a single classification channel where each pixel is considered as part of the FTUs or not.

The original paper that proposed U-Net has a input image tile with a shape of 572 x 572 pixels in particular for annotating electron microscopic images. Its structure could be easily modified to apply to different contexts such as detecting cavities from dental X-ray images.

4 Method

The training data we have are 351 RGB images from the HPA and HubMAP atlases. All HPA images are 3000x3000 pixels and the HubMAP images can range from 4500x4500 pixels down to 160x160 pixels. Each image contains a biopsy slide containing at least one functional tissue units of various organs throughout the human body, namely the kidney, prostate, large intestine, spleen, and lungs. The training images are in tiff format provided under a given path, while each file name is an id corresponds to more parameters in a csv file, which contains the run-length encoding(rle) for labeled mask. To visualize the a training sample with its mask, we write helper functions to decode rle to image and image to rle for submitting our prediction on Kaggle. Below in figure 5 is a demonstration of how rle works as a lossless data encoding method. Basically, it records the number of pixel bytes with a 0 before encountering the next pixel flipping to 1.

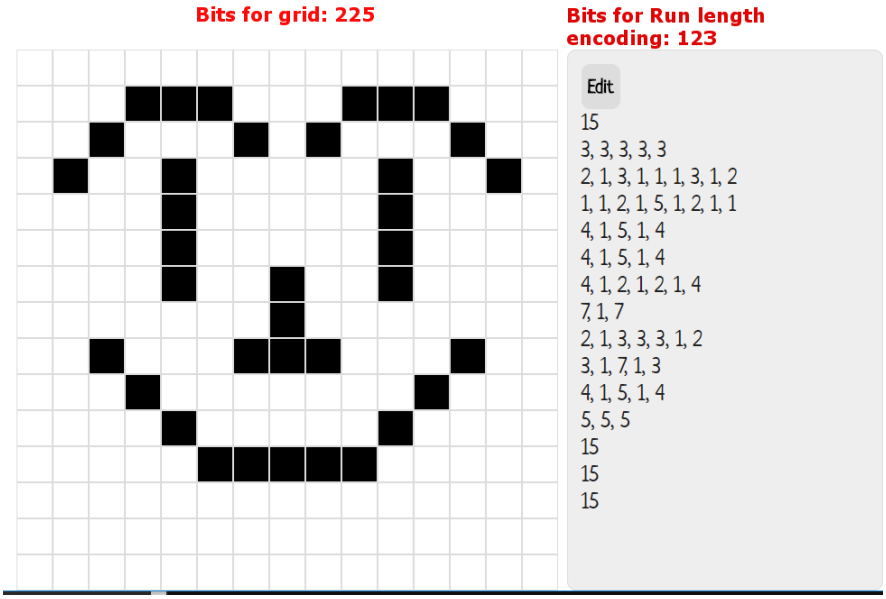


Figure 5: Pixels to run-length encoding

As such, it's obvious that the problem being addressed is an binary object segmentation problem; to segment out the functional tissue units in the biopsy slides with high accuracy, which is evaluated using the DICE metric. As a result, the desired output of the model is a single-channel segmentation mask where FTUs are represented by 1s and background while non-FTUs are represented as 0s. The desired output is not only performing pixel-level binary classification, but we have to transform the mask images back to rle encoding in string format in order for the results to be accepted by Kaggle, then Kaggle will run its hidden test cases at the background and produce a final test-set accuracy.

The inputs of the model are PyTorch tensors representing images and their masks. Due to the large range of image sizes in the dataset and in the interest of model efficiency, we resize every input image to be 512x512 pixels, which is nearly identical to the original input size of the standard UNet of 572x572 pixels. Additionally, due to the limited size of the dataset, we perform two types of data augmentation - horizontal and vertical flips. Since the images are on a microscopic level, there is little sense of orientation within the biopsy slide, which makes the images compatible with both types of flips. We utilized the following pipeline and packages to process tiff files to PyTorch tensors:

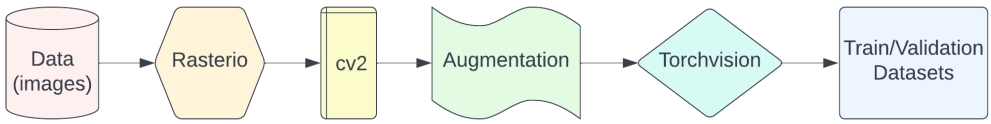


Figure 6: Flow chart of data preparation process

- Rasterio: returning an opened dataset object through a path string.
- cv2: resizing images with different interpolation methods.
- torchvision.transforms: transforming numpy array to pyTorch tensor.

After intensive input data processing, the initial model we tried was a standard U-Net as defined in [12]. Besides the most basic setup, one improvement we made was to use same convolutions all the way through and pad the images with zeros instead of letting the convolution operations change the image size. In this way, the only changes that occur to the image size are the 2x2 max-pooling downsampling steps and the Conv2DTranspose upsampling steps, which more accurately reflects the intention of the U-Net.

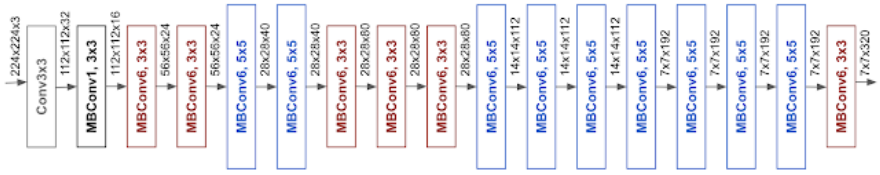


Figure 7: Baseline network architecture for EfficientNet-B0

However, after training the simple U-Net model and reading about some of the previously successful approaches to HuBMAP data in [4], we believed that it's performance was limited by the size of the dataset and decided to instead use a transfer learning approach by incorporating a pre-trained EfficientNet [15] trained on the ImageNet dataset [3]. In order to use transfer learning while still using a U-Net approach, we learned how pre-trained deep networks can be used as the U-Net encoders (the first half of the "U") and then the inverse of the network can be built as the decoder (the second half of the "U") with randomly initialized weights. Once skip connections are added across the network, we now have a U-Net that's half pre-trained and only requires fine-tuning for the first half of the weights. Rather than implementing this from scratch, we were able to find a segmentation model library implemented in Python that has these types of models with our selected pre-trained network, EfficientNet [16]. The architecture of EfficientNet mainly uses a unique type of convolution called the mobile inverted bottleneck convolution (MB-

Conv). These convolutions are blocks of depthwise separable convolutions stacked together with improved efficiency compared to standard convolutions. In addition to these efficient convolutions, inverted residual connections are added between each block. These types of connections are unique because they connect convolution steps in the middle of blocks, when there tend to be fewer filters rather than at the end of each block where large volumes are commonly found. The entire EfficientNet architecture used is shown in 7.

As for actually training our model, we use the Dice loss function because it can compare the pixel-wise agreement between the ground truth, in this case our masks, with predicted segmentation of the FTUs produced by our model, which has been shown to be particularly effective in the domain of medical image segmentation [9]. A really intuitive visualization published in the original U-Net paper demonstrates this dice coefficients really well. The coefficients double count the overlap between predicted mask and the ground truth, then divide by the number of all pixels that are considered as FTUs, regardless of if they are false negative or false positive. Therefore, the dice loss is just 1 minus the dice coefficients.

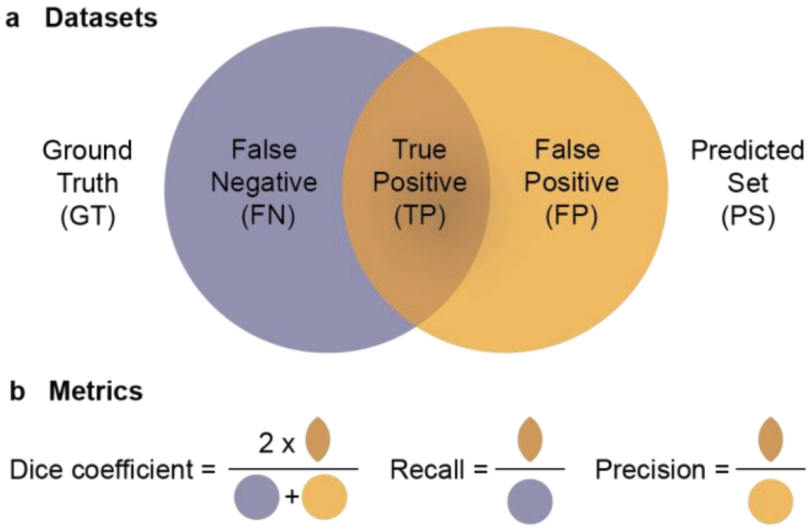


Figure 8: Dice Coefficients defined at Godwin's paper[4]

5 Experiments

For our experiments, we implemented both a transfer learning U-Net with a pre-trained EfficientNet encoder and a standard U-Net. The initial dataset had 351 images and we used basic horizontal and vertical flipping (and both) to quadruple the size of the dataset to 1404. Unfortunately, due to issues with Kaggle's submission system, we had to split the training data into three parts using a ratio of 7:2:1 for the training set, the validation set, and the test set respectively. With this final dataset, we proceeded to go through the training and testing procedure for each model.

In this section, we will evaluate our models' performance and compare results between the two models and among instances from five different organs.

Out best performance is achieved by the simple U-Net. With this model, we gain a dice value

accuracy of 64.51% on the test set, which would be 449th out of 1174 total groups on the Kaggle challenge leaderboard. In particular, the model performs well on the kidney task with a test accuracy of 93.36%, which would have been 385th (top 70%) on a previous Kaggle challenge by the same sponsor which focused solely on the kidney [1].

The U-Net model with the pre-trained EfficientNet encoder has similar performances across the board with a 59.78% test accuracy on the entire set and performs best on the kidney, with a test accuracy of 83.78%.

Aside from the performance, we observe some interesting diversity in all the results we get, the two main aspects of which are as follows:

- **Comparison between two models** Generally the standard U-Net outperforms a transfer learning U-Net built using a pre-trained EfficientNet encoder by about 5%. But since the latter model has a more delicate structure and doesn't need to do more than just fine-tune the encoder part, the EfficientNet-based model gains an advantage in both training efficiency and model size without a sharp decrease in performance level. Our initial inspiration for using such an approach was due to the successful approaches used on the previous Kaggle challenge [1]. All five of the top winning approaches used some sort of pre-trained encoder, with three of them using the EfficientNet in particular. [4]

Model	Test Accuracy	Model Size(MB)
Standard U-Net	64.61%	124.3
EfficientNet + U-Net	59.78%	25.4

Table 1: Accuracy and size of two models

- **Comparison among five organs** For both of the two models, we get a very unbalanced accuracy performance for test data from five different organs. For kidney instances, we reach a high Dice value accuracy of over 80% for both models, while for lung instances, the accuracy is only around 5-10%. We attribute such results to the significant differences between the organs and their appearances as well as the large differences in data quantity for each task within our dataset (far more kidney images than lung or spleen).

Organ	Standard U-Net	EfficientNet + U-Net
Kidney	93.36%	83.78%
Prostate	75.53%	73.75%
Large Intestine	50.65%	49.12%
Spleen	47.44%	43.32%
Lung	11.35%	4.11%

Table 2: Accuracy on five different organs

6 Conclusion and Future Work

In this paper, we apply U-Net models to segmenting multi-organ functional tissue units. We find that a transfer learning U-Net model with a pre-trained encoder achieves a lower accuracy than a standard U-Net in this segmentation task. However, the standard U-Net suffers from its large size and long training exertion. Thus, whether to apply a pre-trained encoder can be considered as

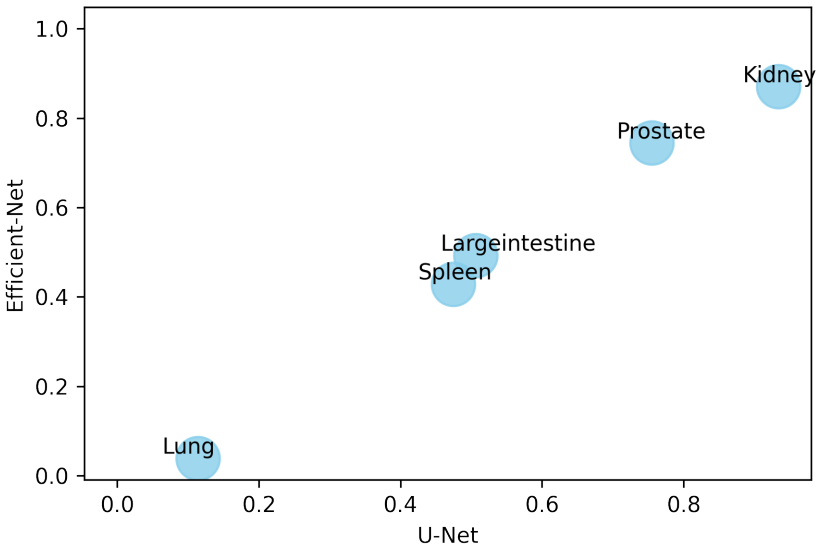


Figure 9: Accuracy on five different organs

a trade-off between efficiency and accuracy, but could provide more avenues for hyperparameter tuning or increasing performance. However, our strong performance from the basic U-Net architecture casts some doubt over how transferable strategies from the previous Kaggle challenge [1] are to a generalized case.

The uneven performance of our models among instances from different kinds of organs is significant and deserves special attention and treatment. One possible approach to solving this issue would be to deal with the five kinds of organ tissue separately, that is, to train five segmentation models for each organ's data, so that interference between different kinds of instances can be reduced to an acceptable level and each single model can be more specific and perform better at its individual task. To accomplish this, a classifier should be trained to first decide which of the five models the current unseen instance should be passed to, and then that trained segmentation model would produce the corresponding mask. With such a complex network architecture, using pre-trained encoders for the segmentation models, in this case, would be very beneficial as they are far lighter than the simple U-Net architecture we used. Another potential approach relates to the imbalance of categories in the data set. Even though we do apply data augmentation to the data set, extra efforts could be devoted to categories that are comparatively scarce or on which the current model has a bad performance. However, we leave such approaches for future work.

7 Contributions

- Writing the proposal - entire group, split evenly
- Coding
  - Implementing the models and training/testing functions - Rob
  - Dataset preparation - Mona and Xinyang

  - Data augmentation - Mona and Xinyang
  - Helper functions for using Kaggle and training - Mona and Xinyang
- Running experiments and collecting data - Xinyang and Rob, 50/50
- Creating the poster - Mona
- Writing the final report - entire group, split evenly

8 Additional Links

- Code - https://github.com/rpitkin19/CS137FinalProject
- Poster - https://tinyurl.com/cs137finalposter

References

[1] Hubmap - hacking the kidney.

[2] Katy Börner, Sarah A Teichmann, Ellen M Quardokus, James C Gee, Kristen Browne, David Osumi-Sutherland, Bruce W Herr, Andreas Bueckle, Hrishikesh Paul, Muzlifah Haniffa, et al. Anatomical structures, cell types and biomarkers of the human reference atlas. *Nature cell biology*, 23(11):1117–1128, 2021.

[3] Wei and Socher Richard and Li Li Jia and Li Kai and Fei-Fei Li Deng, Jia and Dong. Imagenet: a large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255, 2009.

[4] Leah L. Godwin, Yingnan Ju, Naveksha Sood, Yashvardhan Jain, Ellen M. Quardokus, Andreas Bueckle, Teri Longacre, Aaron Horning, Yiing Lin, Edward D. Esplin, John W. Hickey, Michael P. Snyder, N. Heath Patterson, Jeffrey M. Spraggins, and Katy Börner. Robust and generalizable segmentation of human functional tissue units. *bioRxiv*, 2021.

[5] Sardar Hamidian, Berkman Sahiner, Nicholas Petrick, and Aria Pezeshk. 3d convolutional neural network for automatic detection of lung nodules in chest ct. *Proceedings of SPIE–the International Society for Optical Engineering*, 10134:1013409, 2017.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, page 770–778, Jun 2016.

[7] Mohammad Hesam Hesamian, Wenjing Jia, Xiangjian He, and Paul Kennedy. Deep learning techniques for medical image segmentation: Achievements and challenges. *Journal of Digital Imaging*, 32(4):582–596, Aug 2019.

[8] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017.

[9] Shruti Jadon. A survey of loss functions for semantic segmentation. In *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*. IEEE, oct 2020.

[10] Yang Lei, Yabo Fu, Tonghe Wang, Richard L. J. Qiu, Walter J. Curran, Tian Liu, and Xiaofeng Yang. Deep learning in multi-organ segmentation. Jan 2020. arXiv:2001.10619 [physics].

[11] Shengyuan Piao and Jiaming Liu. Accuracy improvement of unet based on dilated convolution. In *Journal of Physics: Conference Series*, volume 1345, page 052066. IOP Publishing, 2019.

[12] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. May 2015. arXiv:1505.04597 [cs].

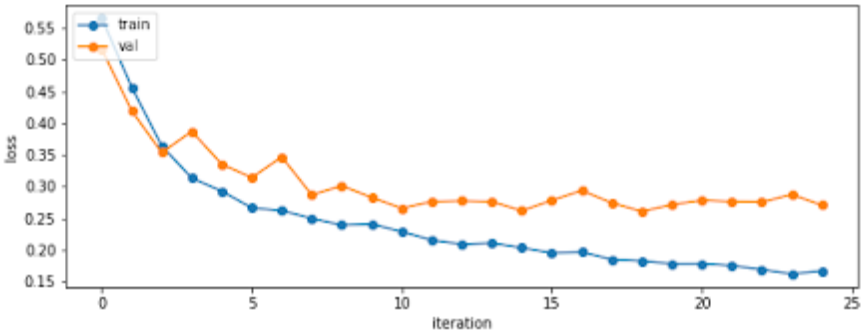
[13] Holger R. Roth, Le Lu, Amal Farag, Hoo-Chang Shin, Jiamin Liu, Evrim B. Turkbey, and Ronald M. Summers. Deeporgan: Multi-level deep convolutional networks for automated pancreas segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Lecture Notes in Computer Science, page 556–564, Cham, 2015. Springer International Publishing.

[14] Hoo-Chang Shin, Matthew R. Orton, David J. Collins, Simon J. Doran, and Martin O. Leach. Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4d patient data. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1930–1943, Aug 2013.

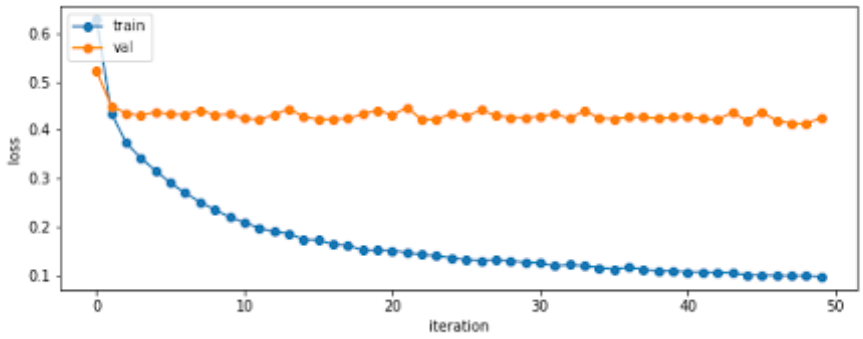
[15] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *CoRR*, abs/1905.11946, 2019.

[16] Pavel Yakubovskiy. Welcome to segmentation models's documentation! — segmentation models documentation.

A Training and validation loss curves



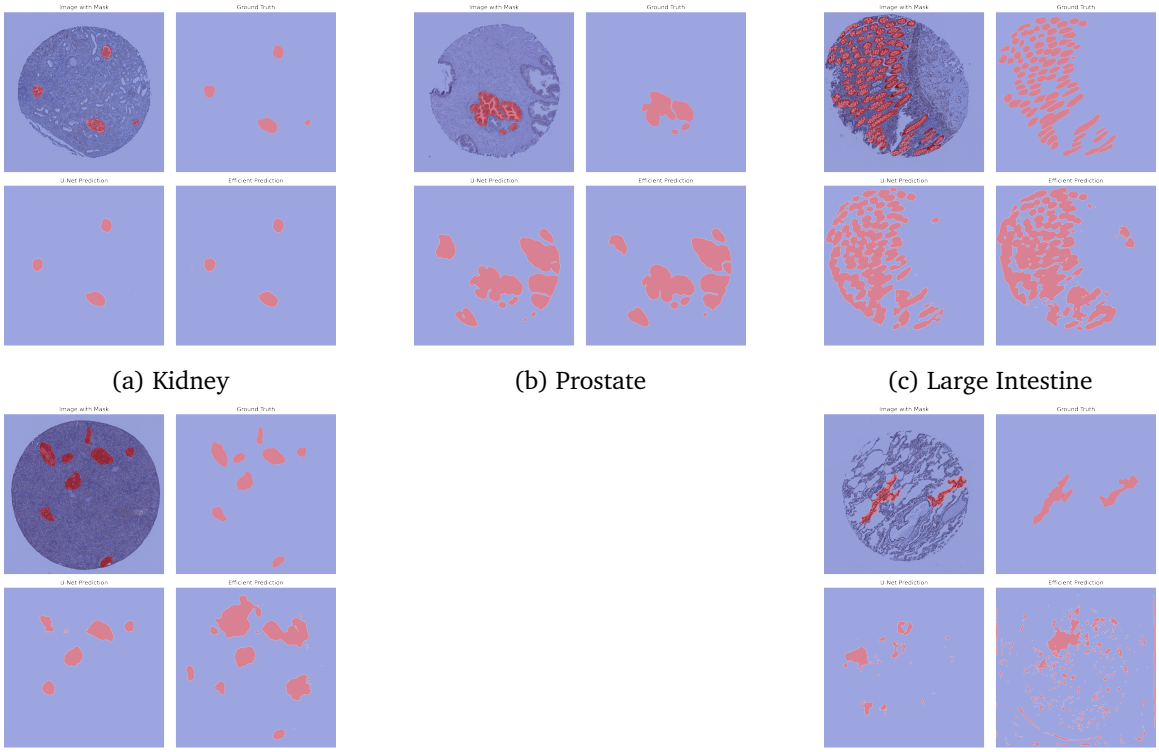
(a) Training and validation loss curves for the standard U-Net



(b) Training and validation loss curves for the transfer learning U-Net

Figure 10: Loss curves

B Unbalanced accuracy performance on five different organs



(a) Kidney (b) Prostate (c) Large Intestine

(d) Spleen (e) Lung

Figure 11: An example instance: accuracy imbalance among organs