Self Supervised Learning for Lower Limb Muscle Segmentation on MRI

Anoosha Pai S
Stanford University
anoopai@stanford.edu

Erick Siavichay
Stanford University
esiavich@stanford.edu

Nitish Gudapati
Stanford University
gnitish9@stanford.edu

Abstract

Segmentation is one of the most important tasks in medical image analysis. Unavailability of large and high quality labeled data set poses a challenge to implement Deep learning models. Self-supervised learning (SSL) approaches with various pretext tasks overcome these limitations. Towards this, we implemented a U-Net architecture, with context prediction as the pretext task followed by instance segmentation as the downstream task. Our dataset consisted of 2D MRI images of lower human limbs, with ground truth segmentation labels of six muscles. We consider three different masking percentages and kernel sizes for inpainting along with four strategies for transfer learning (freezing either encoder or decoder layers in U-Net, neither, or both layers). It was found that 16×16 sized kenels with 10% image masking had the largest Dice score on the validation set when the encoder layers were frozen and the decoder layers were finetuned. We compared the performance of this model to the baseline, a fully supervised segmentation model using U-Net. The performance of the SSL model was about 4.60% better than the baseline model.

1. Introduction

Deep convolutional neural networks (CNNs) have been highly successful in computer vision tasks such as image classification, object detection, etc. In the medical imaging domain, CNNs have shown powerful performance on challenging tasks such as disease classification and cancer detection.

Segmentation is one of the most important tasks in medical image analysis. High fidelity semantic segmentation is crucial for various quantitative and functional estimations for applications in prognosis, diagnosis and treatment not only in research as well as clinical. While manual segmentation is still considered to be the gold-standard, recent advances in CNNs [1, 2, 3] have enabled accurate automatic segmentation of bones and soft tissues from different imaging modalities. Large amounts of training data with labels are crucial for the good training and performance of these models. For natural images, crowd sourcing can be leveraged to annotate the images since it can be achieved by simple human knowledge. For medical images, however, image annotations are limited by the available resources of expert image analysts. In most cases, the desired dataset consists of more unlabelled images than the labelled ones. CNNs trained on a small labelled image data not only give unsatisfactory performance but also do not have the ability to exploit the potentially large number of unlabelled images that may be available in the medical data collection. Furthermore, models pretrained on natural images are not always suitable for medical images since the two data regimes varies substantially and cannot be generalizable. Therefore, improving the performance of deep learning models in a data and label limited regime is a challenging problem.

Self-supervised learning (SSL) has gained more popularity in recent years and is shown to perform well in label-limited regimes. SSL performs supervised feature learning where the supervision tasks are performed on the data itself. Consequently, a very large number of training instances with supervision become available. A SSL model is trained via two tasks: (1) A pretext task, where portions of unannotated images are either masked or jumbled and the model is trained to predict or restore the context of the image, respectively. (2) A downstream task, such as image classification, segmentation, etc., where the model uses the pre-trained weights from task (1) and is then finetuned. Pretraining a CNN based on such self-supervision results in suitable initialisation of weights, which can be subsequently used to train the data with limited manual labels. Therefore, self-supervised learning is a good option to explore the unlabelled images to improve the CNN performance in cases where limited labelled data is available. Recent work [4, 5, 6, 7] has demonstrated that SSL has a higher performance when trained for a pretext task on a relatively large unlabelled dataset and then fine tuned for the downstream task on a small annotated dataset.

We hypothesise that the performance of a self-supervised approach for instance segmentation is at least similar or on par to the performance of a fully supervised network in a data-limited medical image domain. The inputs to our algo-

rithms are medical images of thighs. We then use a U-Net architecture which uses a sequence of CNNs for semantic segmentation to output segmentations of muscles in the image.

2. Related Work

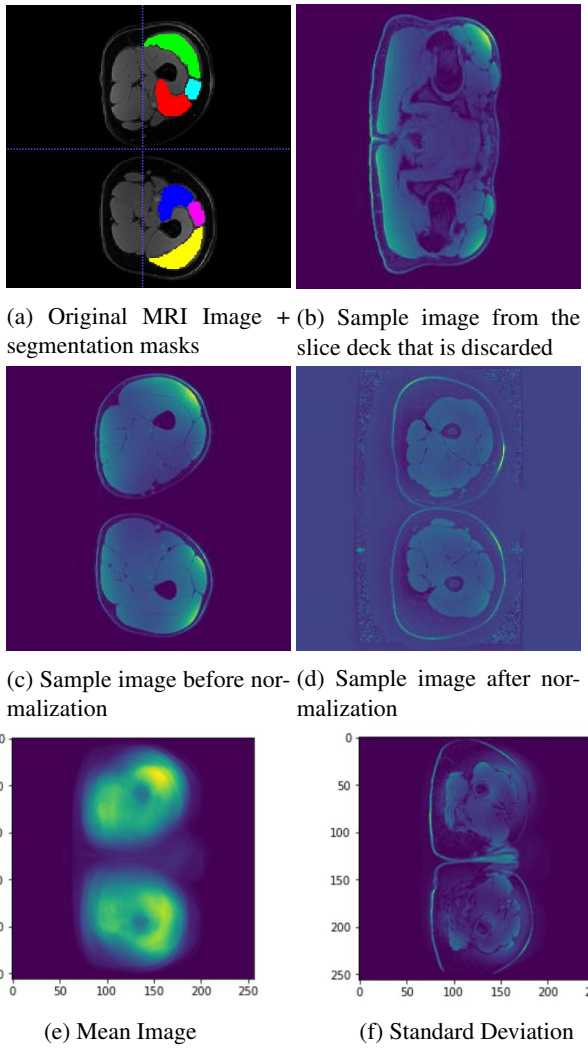
A U-Net architecture developed by Olaf Ronneberger et al. [2] for Bio Medical Image Segmentation is used in this work. The architecture is a fully convolutional network (FCN) that utilizes variations of encoder-decoder layers for semantic segmentation to extract features at different spatial fields of view. The first path of the architecture is the encoder comprising of a stack of convolutional layers with ReLU activations and max pooling layers that downsamples the input image. The second path is the decoder which upsamples the features from the encoder using transposed convolutions. Additionally, the network skips connections by concatenating the output at each level of the decoder with feature maps from the corresponding encoder level.

Image-level representations in pre-training can be generated using context prediction [4] or context restoration [7] and eventually be used for finetuning. Towards the context prediction task, Pathak et al., [4] have trained a neural network with context encoders, which not only understands the content of the entire image, but also produces a plausible hypothesis for the missing part(s). It was found that a context encoder learns a representation that captures not just appearance but also the semantics of visual structures. Features learned due to context pre-training were competitive with the existing methods achieving significant boost over the baseline for classification, detection, and segmentation tasks. This work is relevant to ours as we wish to perform context prediction pretext using the encoder designed in this study.

Another study by Zhang et al [8] analyzes different approaches to context restoration when the masked out pixels tend to be ambiguous. As in our case, these contexts tend to be difficult to restore when the surrounded pixels have very similar color palettes or other highly similar features. In this paper, the authors perform knowledge distillation on the pretext models, which teaches a smaller network to behave nearly exactly as the larger network. These learned semantic priors "not only help to understand the global context but also provide structural guidance for the restoration of local textures." Afterwards, these distilled models are used for inpainting.

A recent study by Chen et al., [7] proposed a novel SSL strategy based on context restoration with an aim to better exploit unlabelled images. The context restoration self supervised task in this work has three major features: (1) learning semantic image features, (2) different types of subsequent image analysis tasks use these image features, and (3) simple implementation. The context restoration strategy was further validated for three common problems in medical imaging: classification (scan plane detection in fetal 2D ultrasound images), localization (localise abdominal organs in CT images), and segmentation (segment brain tumours in multi-modal MR image). It was shown that SSL based on context restoration tasks allows for learning useful semantic features in all three tasks and led to improvements in performance compared to other models. Although, we will not implement a context restoration pre-training task in our current work, this study is relevant for us in implementing the transfer learning approach adopted exclusively for a medical imaging regime.

3. Data



(a) Original MRI Image + segmentation masks

(b) Sample image from the slice deck that is discarded

(c) Sample image before normalization

(d) Sample image after normalization

(e) Mean Image

(f) Standard Deviation

Figure 1: Data Visualization and Normalization

The dataset comprises of 3D volumes of Magnetic Resonance Imaging (MRI) data of the lower limbs for five sub-

jects over five months. All MRI images processed in this work were 2D slices (256×256). Each slice was considered as one data point and was treated separately. Each dataset is comprised of four imaging contrasts, namely fat, water, in-phase, and out-of-phase. Each contrast consists of axial, coronal and sagittal slices covering both limbs in every image. For every contrast and for every subject at one time point, there are about 60-80 axial slices, which results in a total of 1900 images overall. The six muscles (muscles Vastus Medialis, Vastus Lateralis, Rectus Femoris on right and left) have been manually segmented by radiologists based off 2D axial slices of the 3D data. The data was collected at the Stanford Medical Center for another research study. Fig. 1a illustrates a sample MRI slice with ground truth segmentation masks overlayed.

Data from subject 5 for months 3 and 4 (about 140 images) was omitted since the data was found to be corrupted. We carry out the pretext task and the downstream task on the water-contrast images since these images were used as a baseline for manual segmentation. Furthermore, among stack of slices, the first 15% and the last 15% of the images were discarded as the image slices were either poor quality (bad contrast) or were missing the desired classes (Fig 1b).The final dataset consisted of about 1560 images overall (Fig. 1d).

The 23 volumes (subject-month combination) is shuffled and split into 19 volumes for training, two volumes for validation and two for testing making 75% (about 1150 images) - 12.5% (about 205 images) - 12.5% (about 205 images) as training, validation, and test sets, respectively. All the images were normalized to have zero mean (Fig. 1e) and unit standard deviation (Fig. 1f) of the training split.

4. Methods

4.1. Mean Squared Error Loss for Pretext Models

In these models, we only care about generated pixel values that are as close as possible to the original image. We decided to choose the Mean Squared Error (MSE) loss to examine how close generated vs actual pixels are. For pixel p at location i, j, we say the mean squared error over all N pixels is defined as

$$MSE = \frac{1}{N} \sum_{i,j} (p_{i,j} - \hat{p}_{i,j})^2$$

We take the MSE across all images and average this loss.

4.2. Cross Entropy Loss for Downstream Models

For downstream models, the goal was to classify pixels as one of seven classes (0 for background, 1-6 for each of the muscle groups). Since this is a multi-class classification task (multiple classes appearing on the segmentation map), we used use Cross Entropy (CE) loss. The generated segmentation image contains 7 channels for which CE loss was computed. For a particular class and image, the CE between the true class y_i and predicted class $\hat{y}_i$ for each pixel i for a total of N pixels is defined as

$$CE = -\frac{1}{N} \sum_{i} y_i \log \hat{y}_i$$

The CE was then averaged for each class across all images.

4.3. Dice Loss and Score

Dice score has been used to evaluate the performance of downstream task models. A Dice score signifies the percentage of the overlap between two sets, which is a suitable metric for comparing the overlap of true and predicted shapes, or segmentation masks in our case. The Dice score for two sets of pixels A and B is defined as

$$Dice = \frac{2|A \cap B|}{|A| + |B|}$$

The Dice loss is therefore defined as $1 - Dice$.

4.4. Baseline: Fully-Supervised Downstream Model

As a baseline, a U-Net [9] was trained on a fully-supervised segmentation task without any pre-training [1]. Fig. 2 presents the U-Net architecture. The network is trained to produce a 7-channel image corresponding to the six classes of muscles plus a background class. Batch-aggregate Dice scores across all classes was used as a similarity metric to evaluate the performance accuracy of the segmentation task. The model was trained and tested on a batch size of 1. The original U-Net architecture optimizes a loss function consisting of cross-entropy loss plus the Dice loss.

We decided to implement the U-Net model since its architecture has been extensively used for biomedical imaging, which is suitable for our use dataset. The architecture consists of an encoder layers that downsample the input image, squeezing the image into a latent space where the most important features exist, then multiple concatenation layers, and then upsampling decoder layers where the encoded image is transformed back to its original space. The encoder and decoder blocks have a set of 2D convolution layers, batch-normalizations and maxpooling layers. The

[1] Our code uses some components of the original paper. The structure of the U-Net architecture is retained, with changes made in the post-processing layers. We changed the loss function for pretext models to MSE loss. We also changed the way Dice score and loss were computed for the downstream task. We created a dataloader, training and testing scripts, and modified the last layer during pretext training to fit our experimental needs. We wrote our own code for saving the models, transfer learning, and modified the architecture to enable freezing of weights.

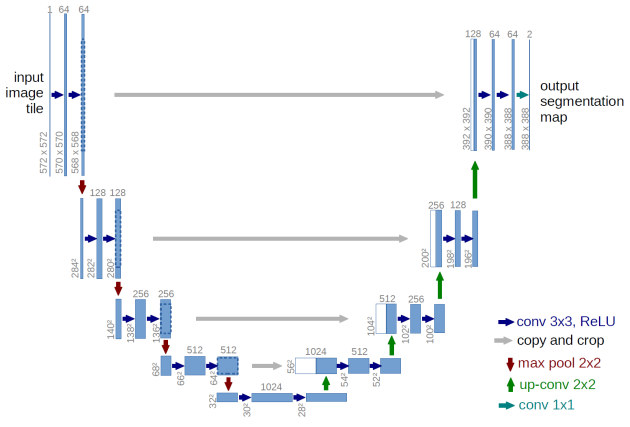


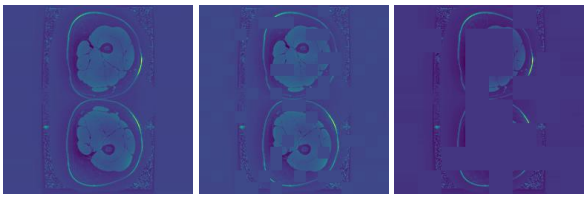
Figure 2: U-Net Architecture, *Source: [2]*

post-processing block contains fully connected layers. All the models in this study were run with mixed precision and bilinear settings. Mixed precision allows for less accurate computations but faster training times, which was essential during the initial stages of traning and validation. The bilinear option allows the upsampling layers to apply a bilinear transformation to its inputs.

4.5. Self-Supervised Pretext Model

U-Net [9] was modified to perform a pretext task (context prediction) followed by a downstream task (instance segmentation). For the context prediction pretext task, the input to the U-Net consisted of a pair of the original image and a inpainted image. The inpainted image was generated by masking out blocks of pixels (setting the value to 0) via our masking percentage and kernel size parameters as seen in Fig. 3. The positions of the masks were sampled randomly without replacement to ensure that the masks don't overlap each other. To ensure that the models learned spatially invariant features, every image was applied with a unique set of masks so that the model could learn to arbitrarily inpaint any image region. The masking percentage is the percentage of pixels in the original image that is desired to be masked out. The kernel size is the size of a particular masking kernel. For example, a masking percent of MP and masking kernel K will have multiple $K \times K$ masking kernels and all of the kernels will add up to masking MP percent of the original image.

The objective for this self-supervised pretext model is to reconstruct the original image from the corrupted image to learn better weights and biases that will be transferred to the downstream task. The loss function of the architecture was modified to minimize the mean squared error (MSE) between the pixels of the original image and the context predicted image. The last layer was modified to predict a single channel of pixels instead of a channel for each class



(a) Original, normalized and restored image (b) Mask percentage = 10%, Kernel size = 16×16 (c) Mask percentage = 50%, Kernel size = 32×32

Figure 3: Illustration for pretext inpainting task

(a total of 7).

4.6. Downstream Model

The downstream models have the same architecture as the baseline models. During training, however, instead of randomly initializing weights, we import the best performing weights that the pretext models have learned as the initial weights for the downstream models

5. Design Choices and Experiments

A series of experiments were performed leading to the design choices in the implementation of SSL and transfer learning. All the experiments for both, the pretext and downstream tasks were executed for 3 epochs. A batch size of 5 and 1 were used for the pretext and downstream task, respectively.

5.1. Experiment 0: Best Network Parameters for the Baseline model

We first optimize a baseline model from which we compare our experimental models. We tested the learning rates in [1e-6, 5e-6, 1e-5], momentums in [0.85, 0.9, 0.95] for RMSprop, and we tested different optimizers (RMSProp vs Adam). We compute the validation Dice score and the test Dice score to compare the rest of our experiment results with. We also tested with batch size 1 and 5.

5.2. Experiment 1: Best Pretext-Task Parameters: Mask Percentage and Kernel Size

All combinations of three mask percentages (10%, 25% and 50%) and three kernel size (32 × 32, 16 × 16, and 8 × 8 pixels) were considered for this experiment. We tested with batch size 1 and 5.

5.3. Experiment 2: Best Network Parameters for Pretext Task: Learning rate and Optimizers

All combinations of three learning rates (1e-6, 5e-6, 1e-5), two optimizers (Adam and RMSProp) and three values of momentum for RMSprop (0.85, 0.9, 0.95) were tested. We also tested batch sizes 1 and 5.

5.4. Experiment 3: Best Transfer learning Parameters: Freezing Weights and Biases

Here we compared downstream performance of the network for different transfer learning strategies. We tested four options for transferring weights and biases: freezing only encoder layers, freezing only decoder layers, freezing, both encoder and decoder layers, and not freezing any layer. We trained using batch sizes 1 or 5.

5.5. Experiment 4: Best Network Parameters for Downstream Task: Learning rate and Optimizers

All combinations of three learning rates (1e-6, 5e-6, 1e-5), two optimizers (Adam and RMSprop) and three values of momentum for RMSprop (0.85, 0.9, 0.95) were tested. Batch size was tested between 1 and 5.

Experiments 1 and 2 were performed in conjunction with one another to determine the best network parameters for various pretext-task parameters. The network parameters for which the mean square error was minimum was chosen for subsequent experiments.

Experiment 1 and 3 were performed in conjunction with one another to determine the best pretext-task and transfer learning parameters. The best pretext model was picked for which the validation Dice score was the highest.

Experiments 3 and 4 were performed in conjunction with one another to determine the best network parameters for various transfer learning parameters for the downstream. The best downstream model was picked for which the validation Dice score was the highest.

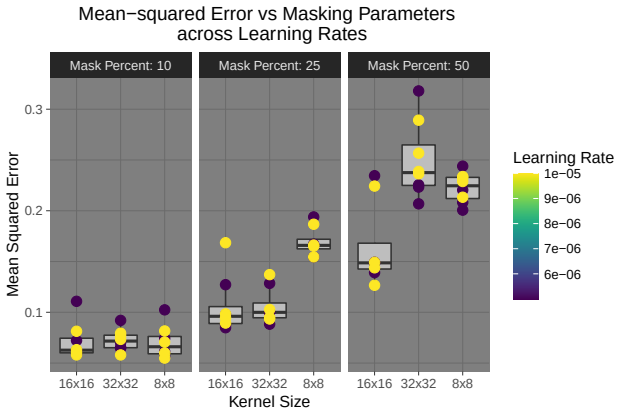
We found that training with batch size 1 usually performed the best except with the pretext tasks.

5.6. Design Choices for Baseline Fully-supervised Model

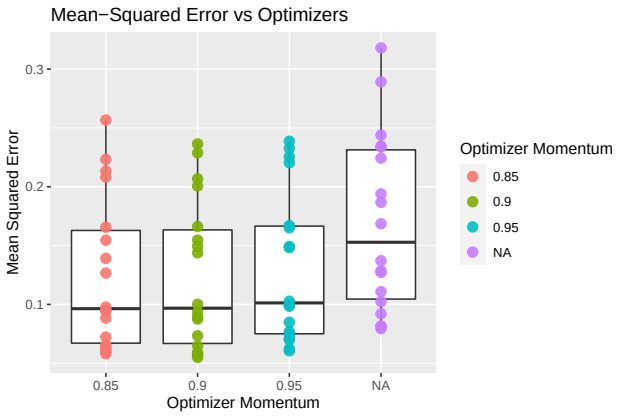
After experimenting, we found that the Dice loss was minimized when choosing the following parameters for this model: learning rate 1e-5, RMSprop optimizer with momentum 0.9, and a batch size of 1.

5.7. Design Choices for Pretext Task

From Fig. 4a, the kernel size of 16x16 was found to have the smallest MSE across all masking percentages. Learning rate of 1e-5 consistently gave lower MSE and was hence the chosen hyperparameter for the pretext model. RMSprop as an optimizer performed consistently better than Adam across all mask parameters (lower MSE, Fig. 4b). momentum values of 0.85 and 0.9 showed better performance compared to 0.95. We chose a value of 0.9 since the variance in its performance was lower than that of 0.85. We also went with batch size 5



(a) Mean-squared error vs masking percentages and kernel sizes across different learning rates



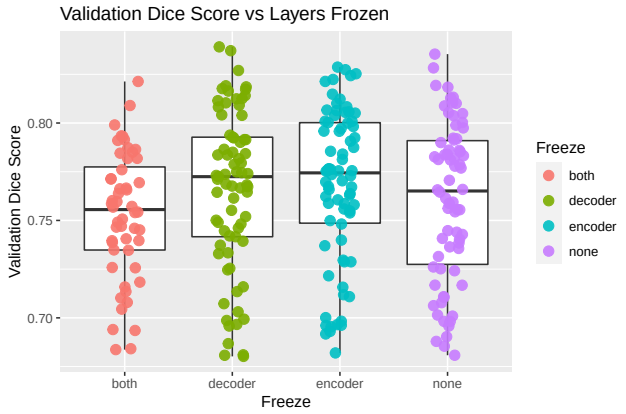
(b) Mean-Squared Error vs Optimizers across all masking percentages and kernel sizes

Figure 4: Plots for experiments 1 and 2: Pretext Task

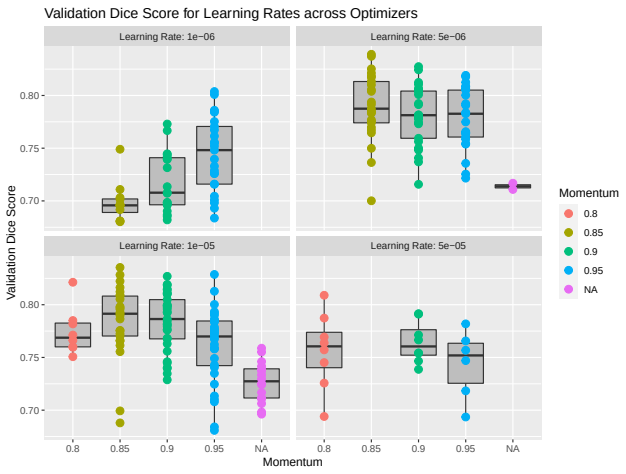
5.8. Design Choices for Downstream Task

All the values for Validation dice score > 0.7 was analysed. It was found that the validation Dice score was the lowest when both, the encoder and the decoder layers are frozen Fig. 5a. This occurs because the architecture then simply reduces to a single layer CNN with only its last layer being fine tuned. Additionally, not freezing either the encoder or the decoder layers performed better than the freezing both layers. This shows that fine-tuning after transfer learning is a better approach. Furthermore, the mean validation Dice score for freezing only the encoder layers is comparable or slightly better to that of freezing only the decoder layers. However, the variances in the output dice score is smaller for transferring the encoder weights than the decoder weights and hence was chosen.

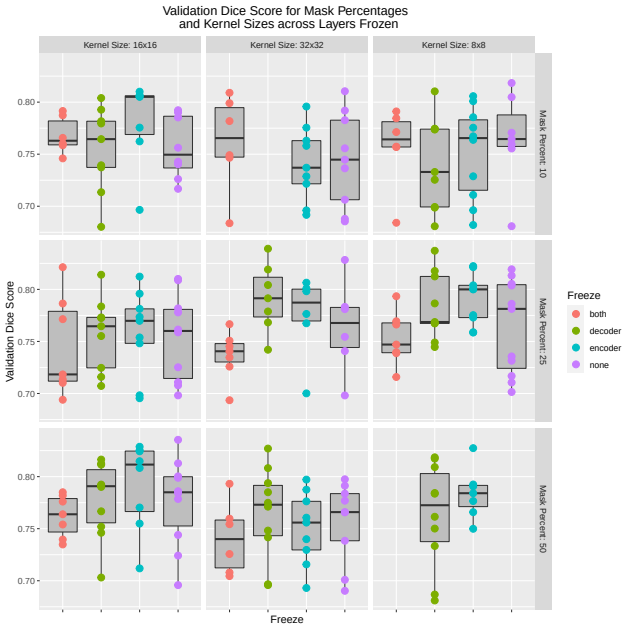
Fig. 5b shows that for validation Dice score > 0.7, RMSprop always performed better than Adam across all MP and KS, weight freezing strategies, and learning rates. It



(a) Validation Dice Score vs Layers Frozen



(b) Validation Dice Score for Learning Rates across Optimizers



(c) Validation Dice Score for Mask Percentages and Kernel Sizes across Layers Frozen

Figure 5: Plots for experiments 1, 3 and 4: Downstream Task

can also be deduced from these plots that the validation Dice score at LR=1e-5 is consistently higher than the other learning rates across optimizers and momentum. Furthermore, although momentum values of 0.85 and 0.9 have the same performance at LR=1e-5, momentum of 0.85 has higher variance and was excluded. Finally, we chose RMSprop optimizer with a momentum of 0.9 and a learning rate of 1e-5 for our downstream task. We also chose a batch size of 1.

KS=16×16 and MP=10 (from Fig. 5b, plot on the top left corner) was found to have the best performance with average Validation Dice score $> 80\%$. Furthermore, these masking parameters also have lower variance in dice score, across all parameters. Additionally, KS=16×16 and MP=50, also has a mean validation Dice score > 0.8. Hence, we also include these parameters for testing.

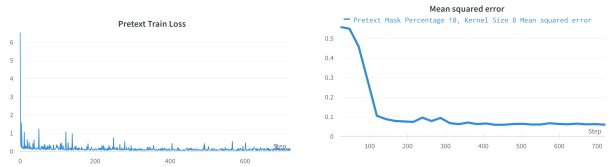
6. Results

Scores	Baseline	Mask Percent = 10	Mask Percent = 50
Validation	0.7916	0.7923	0.7912
Test	0.6226	0.6513	0.6250

Table 1: Dice Scores. The middle and the right columns represent Encoders Frozen with Kernel Size of 16

The final test Dice scores are presented in Table 1. Here, we see that compared to the baseline test scores, freezing the encoder layers with a mask percentage of 50 performs 0.38% better and freezing the encoder layers with a mask percentage of 10 performs 4.60% better.

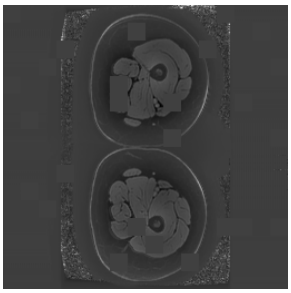
Fig. 6 shows the Validation loss and the mean-squared error for the best model of pretext inpainting task. Fig. 7 presents the masked images, target images and the predicted inpainting images for the best two pretraining models. Fig. 8 shows the Validation loss and the Mean-squared error for the best model of downstream segmentation task. Fig 9 presents the ground truth and the predicted segmentation images for the best downstream segmentation model.



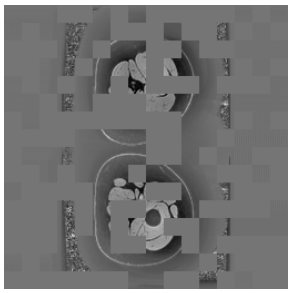
(a) Validation training loss

(b) Validation Mean-squared error

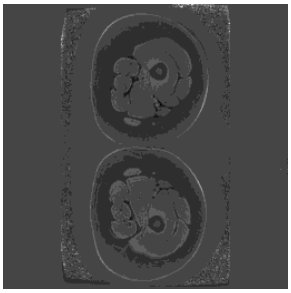
Figure 6: Best pretext model training



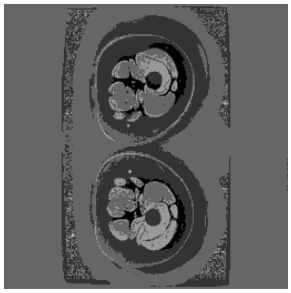
(a) Inpainted input image: MP= 10%



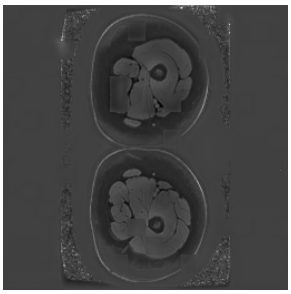
(b) Inpainted input image: MP= 50%



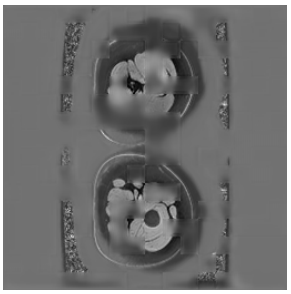
(c) Original image



(d) Original image

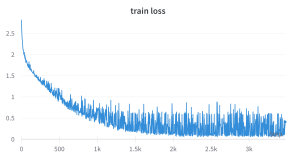


(e) Predicted image: MP= 10%,



(f) Predicted image: MP= 50%,

Figure 7: Best pretext output for kernel size 16×16



(a) Validation training loss

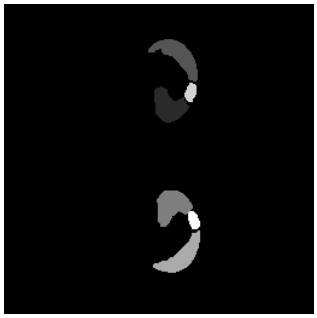
(b) Validation Dice Score

Figure 8: Best downstream model training

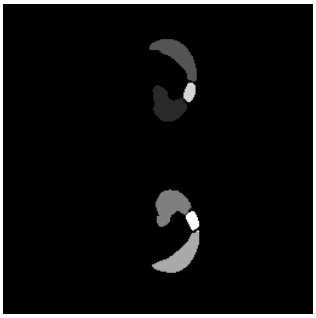
7. Discussion

We observe that self-supervised U-Net model with inpainting as its pretext task trained for downstream instance segmentation performs better than a fully supervised network on the same data.

We, however, observe a large gap between the validation and test Dice scores. Given the small amount of data (especially the much smaller test and validation sets of about 200 samples each), our models would have high variance at test time. In our case, it is noticed that a smaller masking percentage yielded a better performing model. When we look at several of our validation loss functions for both pretext and downstream models, the loss was found to be noisy, as it jitters between higher and lower values even though in general the loss follows a decreasing trend. This could be due to having a small, less diverse and similar type of data.



(a) Ground truth segmentation masks



(b) Predicted Segmentation masks

Figure 9: Best downstream model output for kernel size 16x16

The placement of the masking kernels also have an impact of how good the learned weights are after the pretext task. We made sure masking kernels did not overlap and we also distributed them effectively across the image. If less important or ambiguous regions were to be masked out, the pretext model would not learn the important features needed to effectively restore the corrupted image. We saw in our experiments that some restorations were poor and led to a worse performing downstream model compared to the baseline models.

Usually, we often want to overfit to our training dataset and then fintetune the model to reduce the bias and overfitting via many techniques such as regularization. In our case, it was consistantly difficult to overfit since we had a low amount of data. We can potentially train for more epochs but we found our experiments saturate at about a 0.78-0.83 validation dice score.

8. Conclusion

We see that Self-Supervised learning based on U-Net architecture performs on-par or slightly better than fully supervised baseline segmentation for a data limited regime. We observed that downstream training with a pretext task for a mask percent of 10 and a kernel size of 16 performs better than the baseline, while downstream training with a pretext task for a mask percent of 50 and a kernel size of 16 performs on-par with the baseline segmentation

task. The reason for could be that the model learns more features from the pretext inpainting task by trying to predict the masked patches from a larger pool of data without segmentation labels. Freezing the weights of the encoder layers (latent representation of the features) and training the segmentation masks adapts the decoder layers and the post-processing layers to the segmentation task without unlearning/re-learning the image features.

8.1. Limitations and Future Work

This study forms a preliminary analysis of SSL applied to small medical imaging dataset and has limitations. 1. The dataset considered is small (2000 images). 2. We use the same dataset to perform both pretext training and downstream training. 3. The inpainting parameters (mask percentages and kernel sizes) considered in this study is limited. 4. We explored only inpainting for our pretext task.

Future work could include a much larger dataset to accurately analyze the difference in performance of supervised vs self-supervised methods. One could also randomly distribute our images across training, validation, and testing datasets instead of distributing by subject and month. Furthermore, since we had a small dataset, we can cross validate our results in the future since this technique is often useful for small datasets.

Future studies could be performed to investigate how to best perform SSL on a given dataset by including larger and deeper models. Our work could be extended in this direction by implementing larger and greater number of encoder and decoder layers. Experimenting with different yet medical-imaging domain appropriate pretext tasks (context prediction, context restoration, affine transformations, changes in contrast and texture, etc.) for different type of medical images to pretrain a model could also boost downstream performance. Additionally, other deep learning architectures such as Transformers with long-range dependencies could be implemented for this application. These powerful models could learn better features during the pretext and downstream experiments and thus yield better Dice score performance. Further investigations can be conducted on the performance of network ensambles for medical imaging applications. For example, masked autoencoders [10] based for pretext task followed by CNN based architecture for a downstream task could also provide improved performance. It would also be interesting to look at performance of these models in both data-limited and label limited regime.

One can also focus on using other images in similar medical domains of pretraining data. For example, if the downstream goal of the network is muscle segmentation on 2D MRI images of lower limbs, we can experiment with if the network can be pretrained on images from different parts of the body (like brain, lower limb, pelvis images etc), different modalities and contrasts (like CT, MRI, X-Rays).

9. Contributions

Anoosha was responsible for gathering and curating all of the data. She wrote the code to compute Dice scores for our particular tasks. She was responsible for changing the loss function in the pretext models and downstream tasks. She also ran her share of the experiments on AWS.

Nitish was responsible for maintaining the Github repository. He created the masking scripts to generate mask images for our dataset. He wrote code for transfer learning from pretext to downstream, and he also wrote the code to freeze layers in the original architecture. He also ran his share of experiments on AWS.

Erick was responsible for creating the dataloaders to be able to load our own data into the architecture. He configured the initial implementation of U-Net. He also created the training and testing scripts. He ran his share of experiments on his AWS machine.

10. Acknowledgements

We are grateful to Dr. Akshay Chaudhari for conceptualizing the project idea, his guidance and support during the course of project. We are also thankful to Andrew Schmidt and Valentina Mazzoli from the Stanford radiology Department for providing the data and groundtruth segmentations. We extend our gratitude to Our wonderful teaching assistants, Bohan, Druva and Sumit for their assistance and insightful inputs in our work.

We made use of the code from the public GitHub repository for U-Net from https://github.com/milesial/Pytorch-UNet. Our code uses some components of this original paper. The structure of the U-Net architecture is retained, with changes made in the post-processing layers. We changed the loss function for pretext models to MSE loss. We also changed the way Dice score and loss were computed for the downstream task. We created a dataloader, training and testing scripts, and modified the last layer during pretext training to fit our experimental needs. We wrote our own code for saving the models, transfer learning, and modified the architecture to enable freezing of weights.

References

[1] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017. 1

[2] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical*

image computing and computer-assisted intervention, pages 234–241. Springer, 2015. 1, 2, 4

[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017. 1

[4] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting. *CoRR*, abs/1604.07379, 2016. 1, 2

[5] Marie-Morgane Paumard. Semantic-based automated reassembly of heritage fragments. In *CeROArt. Conservation, exposition, Restauration d'Objets d'Art*, number 12. Association CeROArt asbl, 2020. 1

[6] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European conference on computer vision*, pages 69–84. Springer, 2016. 1

[7] Liang Chen, Paul Bentley, Kensaku Mori, Kazunari Misawa, Michitaka Fujiwara, and Daniel Rueckert. Self-supervised learning for medical image analysis using image context restoration. *Medical image analysis*, 58:101539, 2019. 1, 2

[8] Wendong Zhang, Junwei Zhu, Ying Tai, Yunbo Wang, Wenqing Chu, Bingbing Ni, Chengjie Wang, and Xiaokang Yang. Context-aware image inpainting with learned semantic priors. 2021. 2

[9] UNet GitHub Repository. https://github.com/milesial/Pytorch-UNet. 3, 4

[10] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021. 8