ECE4095 Final Year Project Report

Applying semi-supervised learning to image classification using transformers

Junxian Wang (28801229)

Significant Contributions

- Re-trained and evaluated the original TransUNet architecture's performance
- Designed and converted TransUNet to use a semi-supervised learning framework Self-loop uncertainty

Executive Summary

Medical image segmentation has been a challenging task for deep learning, and one recent advancement in this field is to use image transformers together with a deep convolutional network. TransUNet is the model that has promising results using this technique, that is based on the U-net architecture. However, since obtaining labelled data in the medical field is challenging, it is of public interest to test the possibility of converting TransUNet into a semi-supervised learning model, that do not require all sample data to have labels. This project consists of the following contributions. The first contribution is the evaluation of the performance of the image segmentation model based on image transformers, TransUNet. This test proved to be consistent with the original paper and shows its robustness to different testing data. Then, the second contribution is a research of current semi-supervised learning frameworks that are viable for TransUNet, and the choose one for implementation. In the end, 5 candidate frameworks were selected, and the chosen semi-supervised learning framework is Self-loop uncertainty for its low memory usage. Thirdly, the chosen framework is implemented onto TransUNet. The implementation is complete, however, due to time constraints and one technical issue, during testing, the implementation was not able to perform the correct training. Hence the test result of the implementation was not included. Future work on this project could solve the technical issue preventing the training and perform the planned training and evaluation of the model. Further tests can be conducted on revised model parameters to investigate their effect on the performance and the model's overall robustness.

Table of Contents

Table of Figures

1. Introduction

The aim of this project is to evaluate the ability for an image segmentation network based on image transformer to perform semi-supervised learning.

Recently, image transformer-based image segmentation networks show promising futures compared to classical approaches based solely on convolutional neural networks. One of the approaches is titled TransUNet, which combines the traditional U-net model with an implementation of image transformer titled Vision Transformer. However, in the field of medical image segmentation, supervised learning methods can be expensive to implement in a practical situation. This is because a labelled set of data usually requires both the patients' consent and the manual labour of a skilled medical staff that can precisely identify the segmented parts. Properly segmented image requires even more technical knowledge than data used for image classification, as the exact boundary between parts need to be carefully labelled. Therefore, it is in the public interest to explore semi-supervised or unsupervised learning techniques in the field of medical imaging AI.

To the author's best knowledge, there is yet an attempt at modifying this model to perform semi-supervised learning.

Hence, my project objective is to evaluate a recent promising image segmentation network's suitability to be converted to a semi-supervised learning network. Specifically, the sub-objectives are as follows. The first objective is to evaluate the performance of the TransUNet model in a supervised setting and see if the results are consistent with the proposed results. Then, the second objective is to perform a literature review and find suitable semi-supervised learning frameworks that could be applied to this model. The third objective is to implement the chosen framework on the TransUNet model. Lastly, its success will again be evaluated by comparing the performance of the model in both supervised and semi-supervised setting with the same set of data.

2. Literature Review

The literature review is a key part of this project, as suitable semi-supervised learning methods need to be researched and tried. In the following sections, firstly the traditional image segmentation networks will be introduced, as the model of choice is based on the understanding of this work. In addition, the model used in this project, TransUNet, is introduced in the second section. Furthermore, a semi-supervised learning category, pseudo-labelling, is introduced. The semi-supervised framework of choice is also within this category. Lastly, other related semi-supervised framework candidates are also introduced.

2.1 Traditional image segmentation networks

Traditionally, image segmentation tasks were mostly performed by models based on deep convolutional neural networks. As computing power increased, more advanced models are based on a stack of different convolutional networks, with larger and deeper number of parameters being trained. In the field of medical image segmentation, one of the most iconic approach is the U-Net architecture [1]. It consists of an encoder path and decoder path. The encoder path is a series of convolutional networks and max pooling layers that scales down the image progressively while increasing the channel size. After the encoder, the model consists of convolutional layers with similar channel depth to perform the segmentation. Then it is a similar progressive layers of up sampling convolutional layers, until the original image size is obtained. Hence, the model forms a U shape. At down-sampling and up-sampling layers with the same channel width, skip connections were put in to reference the data before being down-sampled to be used for up-sampling. This network is the major model that inspires the model that this project is mainly based on, TransUNet [2].

2.2 Image transformer based image segmentation network

TransUNet is one of the state-of-the-art model on medical image segmentation, with an innovative use of transformers. TransUNet improves on the classical U-Net by incorporating the newly evolved image transformers, or Vision Transformer (ViT) [3], to be more specific. ViT utilizes transformers that were traditionally used for natural language processing to the field of computer vision. It is an image classification method that is able to reason the relevance of different colour patches on the same image, which is something that convolutional network lacks. Having seen this potential, TransUNet replaces the convolutional network in the encoder that are not used for down-sampling to a stack of transformer layers. In addition, TransUNet also replaces the traditional convolutional networks used in U-Net to ResNets, proposed in the paper: Deep Residual Learning for Image Recognition [4]. ResNet solves the problem of deep convolutional networks having worst performance than shallower networks, provided the depth has reached a threshold. ResNets adds skip-connections within the deep stack of convolutional layers, on top of the existing stacks to feed prior data to later layers, and hence avoided the problem of overfitting due to too many parameters being optimized. Therefore, TransUNet can benefit from both the

image transformers and a deep convolutional encoder and decoder stacks to achieve some of the best performance in this field.

2.3 Semi-supervised learning category: Pseudo-labelling

However, TransUNet is a supervised learning method. This means all sample data fed into the model must be correctly labelled, and there must be an abundance of data in order to achieve satisfactory performance. In the field of medical image segmentation, such data is hard to come by. Semi-supervised learning is a type of learning that allow a majority portion of the data to be unlabelled, meaning that it can benefit from unlabelled data as well. Additionally, less time and effort are needed from skilled medical staff, and they can dedicate their time into more productive works. Successful application of semi-supervised framework onto TransUNet can potentially allow better image segmentation models to be trained, as unlabelled data are more abundant, which could help future medical staff to work more productively.

In order to convert TransUNet to a new model that can perform semi-supervised learning, several methods have been studied. Pesudo-labelling is a category of methods that intend to generate “pseudo-labels” for the unlabelled data by utilizing the information in the data itself, and from the trained model in the current state. One of an older approach is introduced by the paper by Dong-Hyun Lee [5] as the baseline approach. In Lee’s paper, the pseudo-labels are generated simply by the model in the current state, and Lee suggested multiple techniques that can improve the performance of this simple approach, such as adding a Denoiser Auto-Encoder to be trained beforehand to aid in model, or utilizing the parameter drop-out technique to avoid overfitting. Also belonging in the pseudo-labelling category, the model of interest in this work is from the paper: Self-Loop Uncertainty, A Novel Pseudo-Label for Semi-Supervised Medical Image Segmentation [6], written by Li et al. Li’s work is mainly inspired by the popular work: Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles [7], by Noroozi and Favaro. In Noroozi’s work, the unsupervised learning capability of a model based on convolutional neural network is greatly enhanced by a classification task. This task requires the model to break the image input to a 3 by 3 grid of patches, and the patches are then rearranged by some permutation. The model is then tasked to recognize the index of the permutation, in the set of K different permutations. Using this knowledge, Li’s semi-supervised framework trains an encoder-decoder image segmentation model to also perform a jigsaw puzzle classification task, concurrently with the normal segmentation task. Each image is permuted K times, and segmented. In the end, a weighted sum of the restored and segmented puzzle images is combined to form the pseudo label. It has demonstrated superior performance over several other frameworks.

2.4 Other semi-supervised learning frameworks

In addition, several other semi-supervised learning methods were researched, but were not chosen due to different reasons. Virtual adversarial training [8] is a method that builds on the idea of adversarial training, which minimizes the loss in the adversarial direction. The adversarial direction is the direction that will incur the most amount of change from current predictions and is a hidden problem that exists in many classification algorithms. VAT utilizes this in a different way. By relaxing the adversarial direction, VAT utilizes the "virtual" adversarial direction and generates virtual labels for unlabelled data, which can perform semi-supervised learning without sacrificing too much performance. The mean teachers model [9] is another semi-supervised model is an improved version of the method of Temporal Ensembling method. Two models are concurrently trained, the student model and the teacher model. The student model learns and updates from every data sample, while the teacher model calculates an exponential moving average (EMA) of the student's model weights. This teacher model is then used to generate labels for unlabelled data for the student model to learn further. Deep Co-training [10] is another method that uses multiple models for semi-supervised learning. It employs several models trained concurrently. Each model is trained to be different from each other from the supervised samples, through feeding each model an adversarial example of the sample fed into another model. It can also be improved by feeding each model a different subset of the labelled data. For unlabelled data, the models are optimized to give a consistent result by tying the loss function with the difference between the predictions. This way, unlabelled data are segmented with a higher accuracy at the end by combining the predictions from different models together.

3. Overview

This project aims to convert the image transformer-based image segmentation network TransUNet [2] to a semi-supervised learning model. Based on research and trial implementations, the framework of choice is based on Self-loop uncertainty [6], due to its ease of training and understanding. Even though it performs optimization multiple times for each sample data, it does not train multiple models at the same time, which is the primary reason for this to be chosen due to the limited GPU memory on the author's machine. For example, mean teachers model have been tested on the author's machine using a classfication model, however, due to the TransUNet model already taking much of the 11GB of GPU memory, concurrent training of two TransUNet models is physically impossible unless extra resource can be obtained.

The semi-supervised framework and modified TransUNet will be discussed individually in the next section: Detailed Discussion. This section is dedicated to a high-level view of the software module interactions.

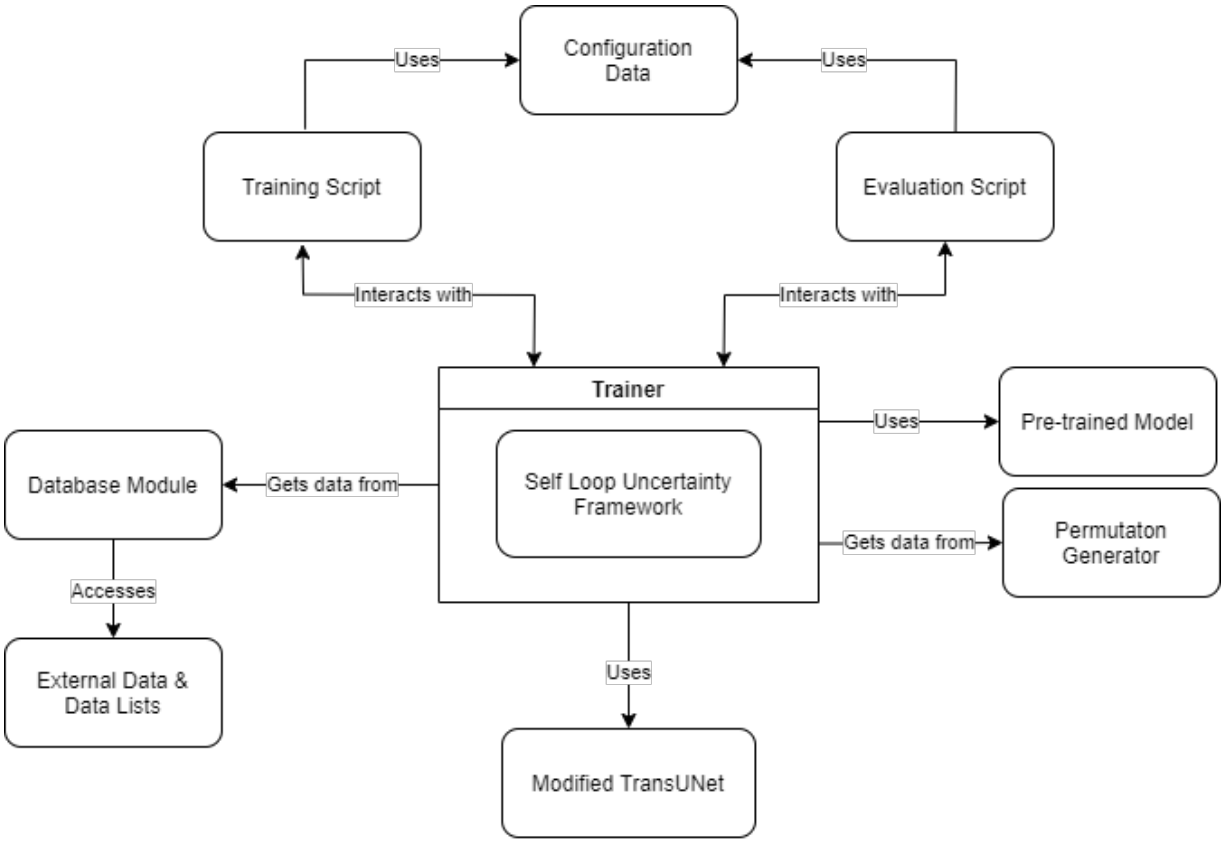

Figure 1: Overview of the software modules in this project

3.1 The central piece: Trainer

As shown in Figure 1, the central piece of software module is the trainer that coordinates the interaction between different modules. It handles both training and evaluation. Inside the trainer, it handles data retrieval, pre-trained model adoption, permutation data retrieval, and general instantiation of other models for the training. The trainer encloses the self-loop uncertainty framework as it is the main training framework that this project focuses on.

3.2 External access of trainer

For external accesses, the trainer gets data from an external data base module. This database module is dedicated to the retrieval of data from the disk, relating the indexing information with each data, automatic random selection of batches, and any transformation applied to the data. The database module is specifically also designed to output jigsaw transformations of each data sample and its transformation index, which is a crucial part of the self-loop uncertainty framework. The trainer also obtains pre-trained model data from the disk. These data are for boosting the initial convergence of the training. In addition, the trainer also obtains permutation data and index from a permutation generator, this data is passed into the database module to generate specific jigsaw transformations. However, the most critical role of the trainer is to coordinate both the training and the evaluation of the modified TransUNet model.

3.3 Starting scripts

At the top, two scripts use the trainer module, the training script and the testing script. These scripts interact directly with the user by obtaining command line arguments, selecting the model of choice, obtain the configuration data, and pass in the model of choice to the trainer. For the configuration data, it is stored locally on disk for ease of modification and separate the connection to the rest of the programs. All model related parameters and training / testing relative parameters are stored in the configuration file.

4. Project Modules

In this section, the two most crucial modules are discussed in detail, namely the trainer module and the modified TransUNet module. Other modules are selectively discussed based on the complexity.

4.1 Trainer module

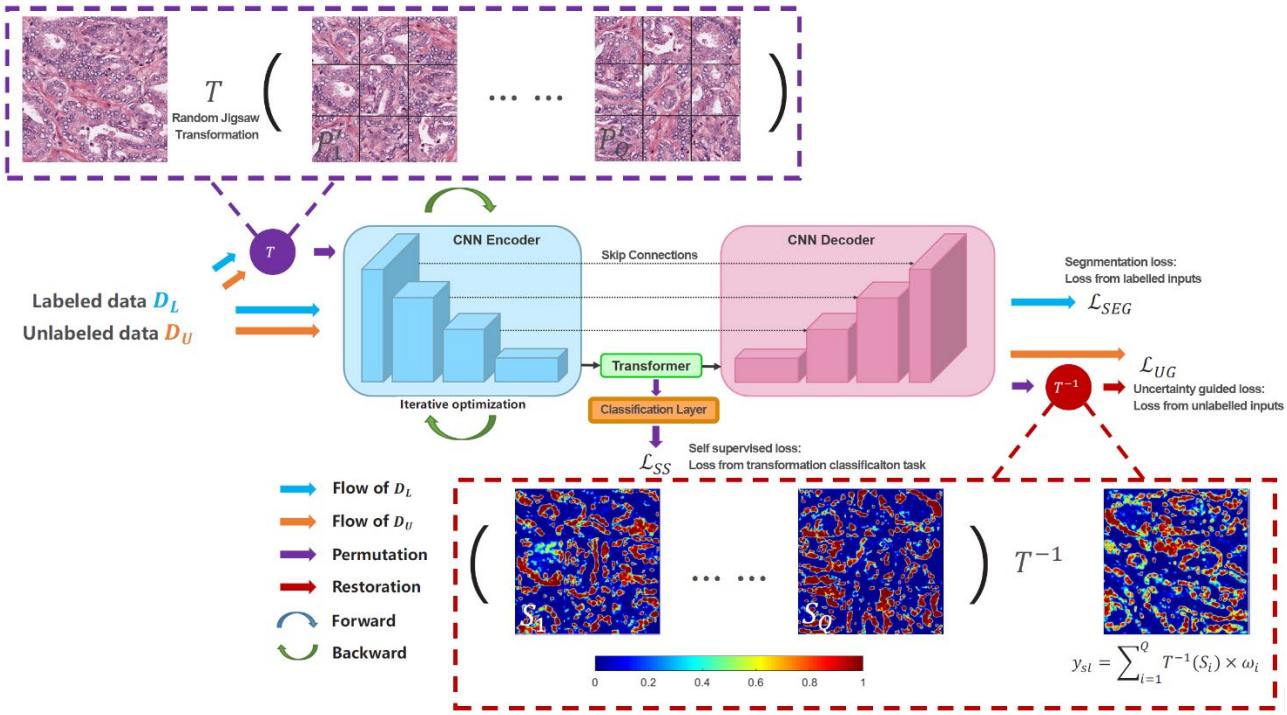

Figure 2: Overview of the trainer architecture of the project

An overview of the project's trainer architecture is shown in Figure 2. This is the module that handles the training and evaluation of the image segmentation models. The general framework follows that of the Self-loop uncertainty paper. At the left, both labelled data and unlabelled data are fed into the framework. They firstly get transformed by a transform T. T is a transform that breaks each input image into a grid of 3 by 3 patches and rearrange based on a randomly given transformation permutation. In this framework, 2 critical parameters govern the jigsaw permutations, K and Q. K is the number of permutations out of all the possible permutations to be possibly selected for application, or the range of possible permutations the model is exposed to. Q is the number of permutations applied for each iteration on an image. Therefore, for each data (labelled and unlabelled), T is applied Q times to obtain Q different transformed "jigsaw" images. The sequence of Q images along with the original unmodified image is put through the image segmentation network together.

Here is when the labelled data and unlabelled data are treated differently. For labelled data, each transformed version of the data is firstly put through the CNN encoder, then simultaneously put through the image transformer and a classification layer. The

classification layer outputs a prediction on the permutation applied for each of the Q transformed images, and then the self-supervised loss L_{SS}, is calculated using cross entropy with the ground truth permutation index. This loss is optimized for Q times for the CNN encoder and classification layer parameters, hence the iterative optimization arrow around the CNN encoder. Then the original unmodified data is put through the transformer, CNN decoder, and arrive at the segmentation output of the network. This output is compared with the data label as in the original TransUNet using cross entropy and outputs L_{SEG}, the segmentation loss, and optimized for the whole model.

For unlabelled data, again, each transformed version of the data is put through the encoder and classification layer to arrive at L_{SS}. However, this time L_{SS} is not optimized but used in later parts of the model. All the transformed data is then put through the image transformer and CNN decoder block to arrive at segmentation outputs. The segmented outputs are firstly transformed back to the original image, and then combined using a weight sum of the different outputs, and the weight is the individual L_{SS} as it represents the confidence of each output. This combined output is then used as the pseudo-label for the input unlabelled data, and its segmentation loss is calculated and optimized.

For the implementation, the trainer module is also responsible for retrieving the data via the database module, logging different information to console and files, and saving the model at different training iteration checkpoints.

4.2 Modified TransUNet module

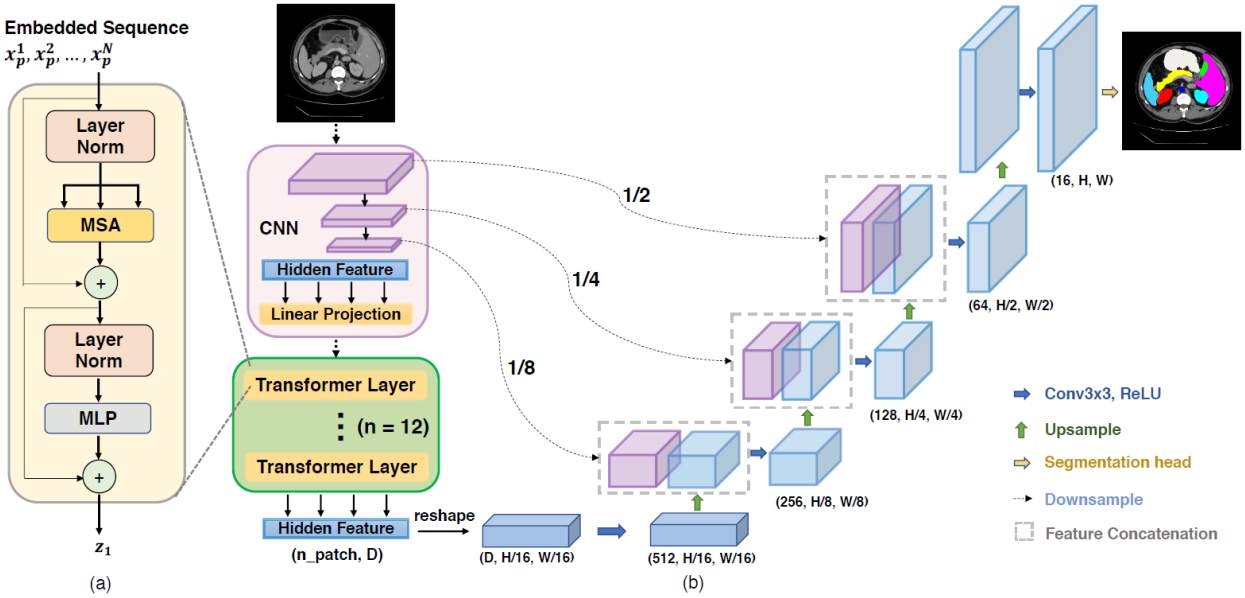

Figure 3: Overview of the network structure of TransUNet [2]

A detailed view of the TransUNet architecture is shown in Figure 3. This network is chosen because of its innovative use of image transformers and superior performance. In this project, most of the original structure is kept like the original paper. For a labelled sample, the image first goes through several layers of convolutional neural networks, the CNN encoders. Each layer downscales the image by ½ and adds to the depth of the channels. At the end of the CNN encoder, the hidden features are converted to a linear projection, and the transformer further slices the image into smaller patches. Each patch is attached a positional encoding for the transformer to reason it's relevant to other patches. The image patches then further go through several layers of transformer layers, the transformer encoder block. In the end, the output from the transformers gets combined back to a full image, and then goes through a similar stack of CNN layers, called the CNN decoder block. This block upscales the image back to the original size incrementally, with the help from the skip connections feeding the data before the corresponding downscale to them. In the end, the original image is constructed with labelled segmentations.

During implementation, the CNN layers follows the structure of ResNets [4], and image transformer block follows the structure of Vision Transformers (ViT) [3]. For this project, during training, the batch size is reduced to 12 for the training of the TransUNet without modification of semi-supervised learning.

In addition, the TransUNet is modified to further include a classification block at the end of the CNN encoder to facilitate the jigsaw classification. In the self-loop uncertainty paper, the authors have used the same encoder network for both down sampling the data and outputting the permutation label predictions. However, it has been found to be confusing to have 2 completely different types of data outputs from the same network. Instead, having looking at

the original paper that propose the jigsaw puzzle task [7], it is clear that for the permutation index classification, additional linear NN layers are added at the end of the convolutional layer stacks to perform the classification, as well as a SoftMax layer for outputting the precise permutation index. Therefore, additional classification linear layers are added. Due to the shear amount of data outputted by the TransUNet CNN encoder (14 * 14 * 1024), originally a total of 6 linear layers were added to progressively decrease the output size to K (the number of possible permutations in the training). However, since TransUNet has already been taking 8.6 GB / 11 GB of the GPU memory, the testing often faces memory bottlenecks.

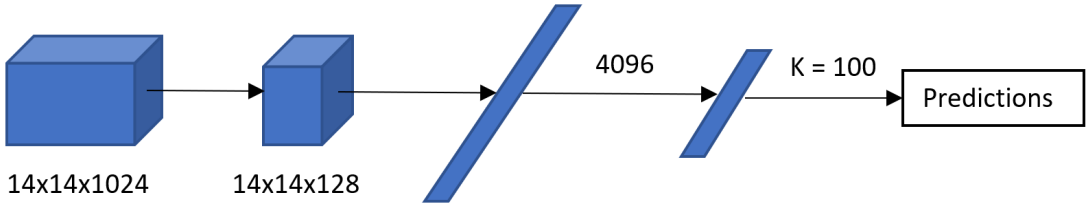

Figure 4: Jigsaw classifier network structure

Hence, the Jigsaw classifier network at the end of the CNN encoder is depicted in Figure 4. The CNN encoder output (14 * 14 * 1024) is first put through a 1*1 convolutional layer that reduces the depth of the channel to 128 (division by 8) before feeding into a single linear layer (with drop out and ReLu activation) that outputs data with a size of 4096. Then, the data is put through a linear classification layer without any activation, and output data with a size of K (which is 100 in this case) as the predictions. This set up can work under the GPU memory constraint of 11GB, although the training can only be run at a maximum batch size of 3.

4.3 Other software modules

The training script is firstly responsible for setting up the models based on user input, this includes the type of model, and some other model related parameters. Then the script is responsible for loading the pre-trained model if a new training is started or load an existing model if the training is resumed. The evaluation script contains similar functionalities, apart from handling the extra testing data that is stored in a different format.

The database module contains classes that takes in the Synapse dataset and output random batches of the data. It is further modified by including a Jigsaw transformation class that loads pre-computed permutation list and randomly modify the taken data with the transformation. Scaling is required as the original data cannot be directly cropped to 3 by 3 grid of patches. This is scaled back when the data is put through the image transformer, however, small loss in quality can be observed from this scaling process.

5. Test Results and Discussion

5.1 Testing the original TransUNet's performance

The first experiment in this project is to reproduce a model based on TransUNet and compare the results with the one published in the paper. To do this, the same dataset used in the paper has been acquired with the author's permission, and the TransUNet implementation is also obtained from the author.

The first attempt was to use Google Colab as the main training platform. Since the model can only be trained in one single pass, Google Colab does not offer the amount of time for a complete training. Therefore, modifications were made to rewrite the training script to be able to save the model and retrieve the model at different epochs. Still, Google Collab has proven to be time consuming to train, since after a certain time period the training stops, and one must wait for a certain amount of time before the training can be resumed. In addition, each time Google Colab need to redownload the python environment from scratch.

Therefore, the experiment was move to a local desktop computer. The computer is configured to run WSL2, Windows subsystem for Linux 2, to perform Linux based programs on Windows. In addition, the local desktop computer has the following specifications:

- AMD Ryzen 5600x 6-core Processor
- 16 GB of DDR4 memory running at 3200Hz
- NVIDIA GeForce 1080TI GPU with 11 GB of dedicated memory

To avoid the problem of memory limitation on a single GPU, the TransUNet model is trained with the following specifications:

- Maximum epoch: 150
- Batch size: 12
- Deterministic training mode
- Base learning rate: 0.005
- Input resolution: 224 by 224
- Patch size: 16
- CNN uses Pre-trained ResNet-50

The original paper uses the Synapse multi-organ segmentation dataset. Within it, it uses 30 abdominal CT scans in the MICCAI 2015 Multi-Atlas Abdomen Labeling Challenge, with 3779 axial contrast-enhanced abdominal clinical CT images in total. In total, the training took around 4 hours to complete

For the evaluation, the data used are from 8 abdominal organs (aorta, gallbladder, spleen, left kidney, right kidney, liver, pancreas, spleen, stomach with a random split of 18 training cases (2212 axial slices) and 12 cases for validation. In total, the evaluation took around 15 minutes

to complete. The performance is record by two criteria, Dice similarity coefficient (DSC) and Hausdorff Distance within 95 percentiles (HD95). The obtained data is as follows.

Table 1: Comparison on evaluation results with original paper (individual organs use DSC % only)

Data source	Mean DSC	Mean HD95	Aorta	Gallbladder	Kidney (L)
Original Paper	77.48	31.69	87.23	63.13	81.87
Project's experiment	77.46	31.86	87.39	59.51	83.77

Data source	Kidney (R)	Liver	Pancreas	Spleen	Stomach
Original Paper	77.02	94.08	55.86	85.08	75.62
Project's experiment	75.68	94.13	60.28	83.70	75.23

As can be seen in Table 1, the re-trained model performs in a very consistent manner with the original result. Even though for the individual organs, the result can differ for a maximum of 4.42 % (for Pancreas), the average result only differs for 0.17%. This highlights the robustness of the TransUNet model, even with a different set of evaluation data, can perform consistently in the supervised setting.

5.2 Testing the semi-supervised TransUNet performance

With technical issues solved for the conversion, the testing of the converted TransUNet did not perform as expected. The network is again put to train under similar physical environments as in the test for the original TransUNet test. For this test, the key training parameters are detailed as follows:

- Maximum epoch: 150
- Batch size: 3
- Deterministic training mode
- Base learning rate: 0.00125
- Input resolution: 224 by 224
- Patch size: 16
- CNN uses Pre-trained ResNet-50
- Total number of permutations allowed (K): 100
- Number of random permutations per sample (Q): 10

This training parameter is similar to the one used in training the original TransUNet, except that the batch size is reduced to 3 due to memory constraints, and base learning rate reduced accordingly. The parameter K and Q are used similar to the one proposed in the self-loop uncertainty paper.

This test was not complete due to the time limit. The main reason is that too much time was spent on some technical difficulties. This includes the configuration of training environment, the actual training, and also the memory problem of trying to implement the paper in the original form.

The first major time spent is the use of Google Colab. Since this project requires a different python environment, all the additional modules need to be redownloaded from scratch in each fresh run of Google Colab, and some of the modules takes a long time to download. In addition, Google colab need to be constantly checked for whether it has stopped running, and sometimes it will prevent the user from continue training until some time later. This is only solved after the local computer has been configured to run WSL2 with GPU successfully.

The second roadblock is that the initially proposed solution from Self-loop did not include any linear layers, and the output simply from convolutional layers cannot perform the desired classification task. Therefore, linear layers were added, and the memory limit was to be solved.

Later on, another major roadblock is the explosion of jigsaw classification losses after around 1000 iterations. Since the training took a while, this problem was only fixed after a long time spent at debugging, which is that the model cannot optimize all Q iterations of losses in a single pass. Each loss must be optimized individually.

In the end, the implementation of the framework and the conversion of TransUNet is complete, and the labelled data path is successfully run. However, for the unlabelled data path, although the implementation is complete, one problem needs to be solved. The problem is that the input image cannot be perfectly sliced to the number of patches, therefore scaling was used. However, this scaling disrupts the original TransUNet's input size constraints. Further tests need to be conducted to alter the TransUNet's internal size computation. After this test, this path is not completely tested, so more potential problems could arise.

6. Conclusion and Future Work

At the end of the project, several of the project objectives were met. Firstly, the training and evaluation of the TransUNet architecture has been successfully finished with great results, reproducing the results from the papers. Secondly, a range of different semi-supervised learning methods were researched and tried for their suitability, 5 candidates were selected and one of them is chosen to be used for the project, Self-loop uncertainty. This has been well documented in the literature review section. Thirdly, an application of the semi-supervised learning framework was performed by converting TransUNet to use it. However, even though the implementation is done, some technical difficulties were met, and the implementation was unable to be trained and tested for its performance due to time constraints.

To improve on this project, one of the most important things is to solve the current technical difficulty faced. Then the planned training and evaluation of the model can be conducted. In addition, training with slightly different model parameters can be performed to further evaluate the effect of those parameters and test for the overall robustness of this design.

GitHub Repository: https://github.com/JXKun980/TransUNet

YouTube Video: https://www.youtube.com/watch?v=SSogNzhKBJE

References

[1] O. Ronneberger, P. Fischer, and T. Brox, 'U-Net: Convolutional Networks for Biomedical Image Segmentation', *arXiv:1505.04597 [cs]*, May 2015, Accessed: May 26, 2021. [Online]. Available: http://arxiv.org/abs/1505.04597

[2] J. Chen *et al.*, 'TransUNet: Transformers Make Strong Encoders for Medical Image Segmentation', *arXiv:2102.04306 [cs]*, Feb. 2021, Accessed: Mar. 31, 2021. [Online]. Available: http://arxiv.org/abs/2102.04306

[3] A. Dosovitskiy *et al.*, 'An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale', *arXiv:2010.11929 [cs]*, Oct. 2020, Accessed: May 26, 2021. [Online]. Available: http://arxiv.org/abs/2010.11929

[4] K. He, X. Zhang, S. Ren, and J. Sun, 'Deep Residual Learning for Image Recognition', *arXiv:1512.03385 [cs]*, Dec. 2015, Accessed: May 26, 2021. [Online]. Available: http://arxiv.org/abs/1512.03385

[5] E. Arazo, D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness, 'Pseudo-Labeling and Confirmation Bias in Deep Semi-Supervised Learning', Aug. 2019, Accessed: May 26, 2021. [Online]. Available: https://arxiv.org/abs/1908.02983v5

[6] Y. Li, J. Chen, X. Xie, K. Ma, and Y. Zheng, 'Self-Loop Uncertainty: A Novel Pseudo-Label for Semi-Supervised Medical Image Segmentation', *arXiv:2007.09854 [cs, eess]*, Jul. 2020, Accessed: Oct. 21, 2021. [Online]. Available: http://arxiv.org/abs/2007.09854

[7] M. Noroozi and P. Favaro, 'Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles', *arXiv:1603.09246 [cs]*, Aug. 2017, Accessed: Oct. 21, 2021. [Online]. Available: http://arxiv.org/abs/1603.09246

[8] T. Miyato, S. Maeda, M. Koyama, and S. Ishii, 'Virtual Adversarial Training: A Regularization Method for Supervised and Semi-Supervised Learning', *arXiv:1704.03976 [cs, stat]*, Jun. 2018, Accessed: May 26, 2021. [Online]. Available: http://arxiv.org/abs/1704.03976

[9] A. Tarvainen and H. Valpola, 'Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results', *arXiv:1703.01780 [cs, stat]*, Apr. 2018, Accessed: May 26, 2021. [Online]. Available: http://arxiv.org/abs/1703.01780

[10] J. Peng, G. Estrada, M. Pedersoli, and C. Desrosiers, 'Deep Co-Training for Semi-Supervised Image Segmentation', *arXiv:1903.11233 [cs]*, Oct. 2019, Accessed: May 26, 2021. [Online]. Available: http://arxiv.org/abs/1903.11233