Semantic Segmentation of Puzzle Pieces

Jamie Burns
School of Computer Science and Applied Mathematics
University of the Witwatersrand
Johannesburg, South Africa
1628649@students.wits.ac.za

Wesley Earl Stander
School of Computer Science and Applied Mathematics
University of the Witwatersrand
Johannesburg, South Africa
1056114@students.wits.ac.za

***Abstract*—This report details our implementations of different image segmentation techniques to semantically segment puzzle pieces for the generation of masks. We show that the U-Net is the best performing semantic segmentation technique outperforming both the Gaussian mixture model and the baseline. All models were evaluated utilizing 6-fold cross validation and the ROC AUC metric was used to compare performance of models. The baseline model achieved a score of 0.944, the GMM achieved a score 0.970 and the U-Net achieved a score of 0.997 establishing the U-Net as the go to option for semantic segmentation of puzzle piece images.**

***Index Terms*—Image-processing, U-net, VGG16, GMM**

I. Introduction

The problem of image segmentation can be described with two main problems: determining the correct mathematical distribution to capture the distribution of the data and determining the correct features to extract the right discriminative information for the model to learn and generalise adequately. The following paper solves each problem utilizing the correct respective approach to determine which approach is more effective in comparison, for accuracy. To solve the problem of determining the correct mathematical distribution, a technique which utilizes a set of Gaussian distributions called a Gaussian mixture model is used. To solve the problem of extracting the correct features from the data, a convolution neural network is utilized which is built on the features that a VGG16 encodes. The usage of the VGG16 with the *imagenet* weights to encode features is purely to expedite the process of training and testing of different data augmentations by leveraging transfer learning.

II. Architecture

A. Baseline Model

For our baseline, we trained two Normal distributions, one for the background pixels and one for the foreground pixels. Both models were trained using the Maximum Likelihood.

B. Gaussian Mixture Models

A Gaussian mixture model (GMM) is a probabilistic model, which is used to represent normally distributed subpopulations within an overall population. They have common applications in tasks such as image segmentation and have been used in speaker recognition systems. GMMs can be trained using the iterative Expectation-Maximization (EM) algorithm or Maximum A Posteriori (MAP) estimation. For our implementation, we focused on the EM approach.

C. U-net

The U-net used is a CNN architecture that extends a VGG16 architecture that is used for image segmentation. The VGG16 allows for the usage of the pre-trained weights from imagenet to help with the encoding of the image. The encoder is the VGG16 architecture and the decoder is the remaining path of the network. The U-net architecture has 28,804,545 parameters which 14,089,857 of them are trainable due to freezing the VGG16 part of the network. This is a significant reduction from the usual 31,031,685 parameters in a standard U-net. The utilizing of the VGG16 reduces training time by utilizing transfer learning. The input to the network has to be square so the image is resized to 768x768 to make sure that down-sampling by the 2x2 max-pooling will occur on an evenly square image at each layer to ensure seamless tiling on the output map. The VGG16 architecture is built up of five blocks where each block is connected by a 2x2 max pooling operation with stride two for down-sampling. The first two blocks have two 3x3 convolutions and the last three blocks have three 3x3 convolutions each followed by a rectified linear unit (ReLU). The VGG16 network concludes with the final convolution of block five. The expanding path to decoding the data occurs symmetrically. The decoding path also has five blocks that are connected by up-sampling followed by a 2x2 convolution. The first three blocks have three 3x3 convolutions and the last two have two 3x3 convolutions followed by a rectified linear unit (ReLU). The last four blocks of the decoder path have the output of the respective encoder block concatenated to the feature channel. At the final layer, a 1x1 convolution is used to map the sixty-four component vector to classes. The full architecture is shown in figure II-C. Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. the x-y-size is provided at the lower left edge of each box. White boxes represent copied feature maps. The arrows denote the different operations.

III. Explanation and Training

A. 6-fold cross validation

We utilized 6-fold cross-validation to evaluate the accuracy of the model and set the hyper-parameters of the models. The technique involves separating a shuffled set of input data and dividing it into 6 equal-sized sets. For each run respectively a different set is used for validation and testing. The final 4

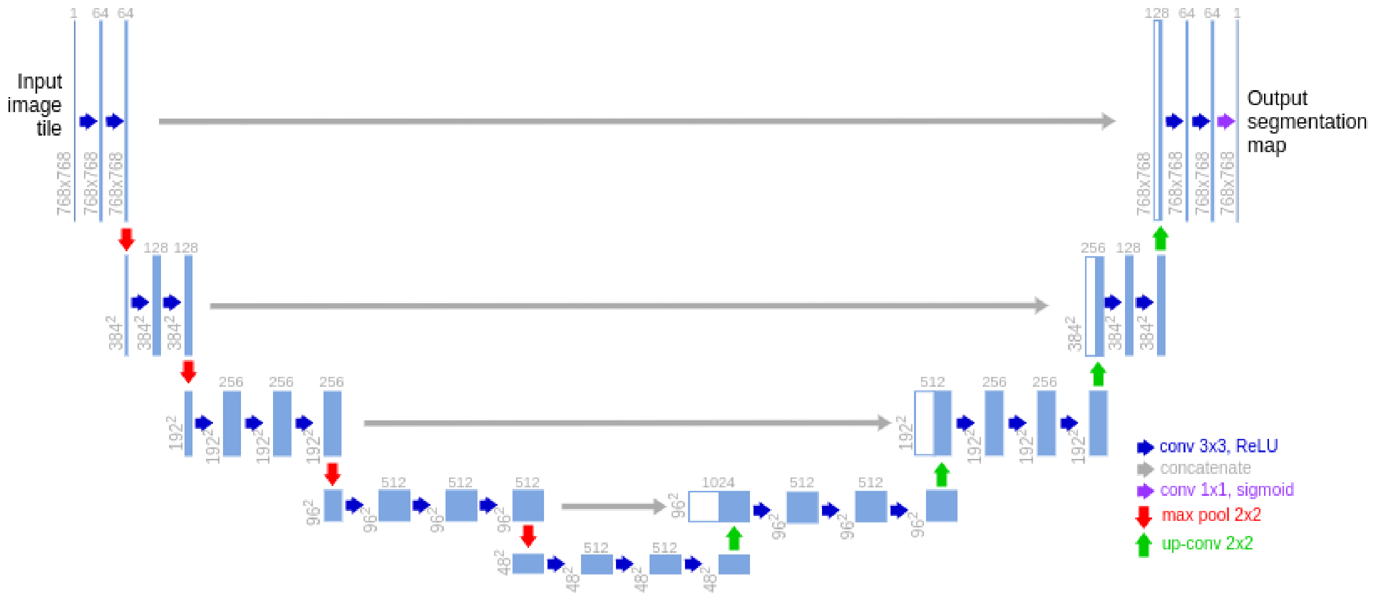

Fig. 1. U-Net Architecture - Extending a VGG16 Network

sets for each run are used to perform training on the model. The testing accuracy from the model is averaged across all 6 runs to determine the overall accuracy of the model and hyper-parameter combination.

B. Baseline Model

To train both the background and foreground models Maximum Likelihood (ML) was used. ML finds the set of parameters θ_i which *maximise* the probability of the data x_i. As mentioned Section in II-A, Normal distributions were used to model both the foreground and background pixels, and as such the Mean Vector and Covariance matrix was calculated as follows:

$$\hat{\mu} = \frac{\Sigma_{i=1}^{I} x_i}{I} \tag{1}$$

$$\hat{\sigma}^2 = \Sigma_{i=1}^{I} \frac{(x_i - \hat{\mu})}{I} \tag{2}$$

Then the probability density of each model was substituted into equation 3, to find the probability of a pixel belonging to the foreground.

$$P(l = f|\vec{x}) = \frac{\lambda Norm_{\vec{x}}\left[\vec{\mu}_f, \Sigma_f\right]}{\lambda Norm_{\vec{x}}\left[\vec{\mu}_f, \Sigma_f\right] + (1-\lambda) Norm_{\vec{x}}\left[\vec{\mu}_b, \Sigma_b\right]} \tag{3}$$

Where λ is the probability of a pixel being a foreground pixel(calculated on the training data).

C. GMM

The EM approach iteratively updates the Mean, Covariance and 'weights' of the respective Gaussians, until the Maximum Likelihood (Described in Equation 7) converges. The update rules for the aforementioned parameters are shown below:

$$\lambda_k^{[t+1]} = \frac{\Sigma_{i=1}^{I} r_{ik}}{\Sigma_{j=1}^{K} \Sigma_{i=1}^{I} r_{ij}} \tag{4}$$

$$\vec{\mu}_k^{[t+1]} = \frac{\Sigma_{i=1}^{I} r_{ik} \vec{x}_i}{\Sigma_{i=1}^{I} r_{ik}} \tag{5}$$

$$\Sigma_k^{[t+1]} = \frac{\Sigma_{i=1}^{I} r_{ik} (\vec{x}_i - \vec{\mu}_k^{[t+1]})(\vec{x}_i - \vec{\mu}_k^{[t+1]})^T}{\Sigma_{i=1}^{I} r_{ik}} \tag{6}$$

$$\text{Maximum Likelihood} = \Sigma_{j=1}^{k} log \left(\Sigma_{i=1}^{I} r_{ik}\right) \tag{7}$$

where

- $\lambda_k^{[t+1]}$ is the weight of each Gaussian.
- $\vec{\mu}_k^{[t+1]}$ is the Mean for each Gaussian.
- $\Sigma_k^{[t+1]}$ is the Covariance for each Gaussian.
- I, K is the number of pixels and number of Gaussian's respectively.
- Finally, r_{ik} is probability that *pixel I* belongs to *Gaussian K*

When the Maximum Likelihood converges, the optimal parameter set for each of the Gaussians has been found. The input into the model consists of a vector with dimensions MxN, where M is the total number of pixels in the training/testing set and is calculated by taking the product of the number of training/testing images with the height and the width of each of the images (fixed at 256x192). N is the number of features per pixel, and is dependent on the feature set (further explained in section IV-B). Upon convergence of the Maximum likelihood, the GMM models are subbed into 3, in place of the Normal distributions. One of the issues experienced while training the GMM was the Covariance matrix going singular. This happens when the Gaussian (to which the Covariance matrix belongs), collapses to a single point. To prevent, a regularization diagonal matrix was added to each of the Covariance matrices to prevent them from going singular. The regularization diagonal matrix contains extremely small values (1e-6) in the diagonal, which keeps the relative covariances positive definite and non-singular.

D. U-net

The process whereby the network evaluates an image involves utilizing the encoder to reduce the size of the input by increasing the number of feature maps. The decoder then takes the feature maps and minimizes the number of feature maps to return the matrix to its original size, whereby each pixel is interpreted and mapped to the classes in the ground truth to produce a segmentation. The U-net trains by comparing the output of the segmentation map to the ground truth and back-propagating the loss through the decoder part of the network. The encoder which utilizes a VGG16 network has it's weights frozen and does not train. The VGG16 leverages transfer learning by utilizing *imagenet*'s weights that have been pre-learnt on a set of natural scenes to determine effective learned features. This reduces overall learning time [1].

The training occurred on a batch size of two as the total number of images were very little and the GPU couldn't store more than a batch size of two. The batch size of two was required as a batch size of one would have caused a bad gradient to step on outliers, whereby there is a full step size updating the weights on the outlier which can move the model into a local minimum or in the wrong direction to finding a minimum thereby reducing the overall performance of the model. The batch size of two hence provides smoothing to the gradient steps and increases the performance of the training. The learning-rate was optimized and it's optimization is discussed in section V-B2

The models were trained to minimize the loss of the validation set of images and masks. A technique whereby only the best model achieved according to validation loss would propagate its weights through the network for the next epoch was used to ensure that no over-fitting would occur. The minimization of the validation loss ensured that the network didn't over-fit to the data and would provide comparable results on the test images. The loss was calculated utilizing a binary cross-entropy function described in equation 8 as the classifier is a binary classifier. The loss function further increases the accuracy as it provides a logarithmic correlation to the probability of each class for each prediction. This ensures that good predictions return low values and bad predictions return high values. The cross-entropy initially has a larger value than the entropy of the data so the minimizer aims to minimize the Kullback-Leibler Divergence (a measure of dissimilarity between two distributions) between the actual entropy and the predicted entropy (cross-entropy) which increases the accuracy of predictions.

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot \log(p(y_i)) + (1-y_i)\cdot(1-p(y_i)) \quad (8)$$

Two forms of training occurred and both were tested with 6-fold cross-validation. The first form of training utilizes two approaches whereby the images are trained with 32 images, validated on 8 and tested on another 8. The two techniques include the default images and a set of cleaned images as described in section IV-A2. The second form of training utilizes two forms of data augmentation to increase the total amount of images and trains the model to recognise more general classifications. The data augmentation occurs on the set of 32 training images to increase the training set to 300 images batched into 150 time-steps. The validation and testing images both remain at 8 images. The two forms of data augmentation are a mild augmentation as described in section IV-A3 and the second is a more rigorous augmentation as described in section IV-A4. The results of these different training techniques are discussed in section V.

IV. Experimental Setup

A. Data Description

1) Initial Data: The initial data that is used is a set of 48 images and respective masks of size 1024x768. Each image represents a different puzzle piece in a puzzle depicting an artwork based on the movie *Frozen*[1]. The labels in the form of masks are explicitly generated by hand to get the most accurate training.

2) Data-preprocessing for U-Net: The images and masks were reshaped from 1024x768 to 768x768 for the U-net architecture. The images and masks required even x- and y-sizes for each 2x2 max-pooling operations to allow for a seamless tiling of the output map [2]. The images were further reduced in size to 192x192 to allow for the GPU to hold the entire model to expedite the training and cross-validation process as well as provide comparable results to the other models. The model then utilizes 192x192 input image tile for all experiments.

One set of masks were pre-processed to clean any artifacts such as the image not being binary, the image not being grey-scale and the image having small errors in the background of the mask. The respective techniques were a rounding, conversion to grey-scale and a binary morphological opening with a disk of radius 10. This set is the cleaned set to determine if the model performed better with this pre-processing.

The following augmentations utilize rotational shift to account for images that were not taken at the same orientation, width and height shifts if the image wasn't exactly centred as well as shear stretching if the image was taken at an inclination. The parameters also include a vertical and horizontal flip to account for pieces that aren't the same orientation and a zoom variation is implemented to account for different image capture distances from the camera.

[1]A Disney movie with characters Elsa, Anna, Olaf and Kristoff.

3) Mild Augmentation: The mild augmentation utilizes the following hyper-parameters which fit in the bounds of the data expectations.

- rotation shift of range 20 degrees
- width shift range of 10%
- height shift range of 10%
- sheer stretch range of 10%
- horizontal flipping
- vertical flipping
- zoom range of 10%
- fill mode or nearest

4) Rigorous Augmentation: The rigorous augmentation utilizes the following hyper-parameters which fit in the bounds of the data expectations.

- rotation shift of range 45 degrees
- width shift range of 20%
- height shift range of 20%
- shear stretch range of 20%
- horizontal flipping
- vertical flipping
- zoom range of 20%
- fill mode of nearest

B. GMM Feature Sets

1) RGB: RGB (Red, Green and Blue) features are fundamental colour features for a given image. RGB can be defined as an additive colour model where red, green and blue hues of light are added together to create different colours. For our approach, the RGB features were treated as the baseline feature set, from which we hoped to improve upon with more descriptive features. RGB features consist of 3 channels, and as such, each pixel had exactly 3 features ($N = 3$).

2) RGB and Degree of Gaussian: The next feature set we focused on was a combination of RGB and DoG (Degree of Gaussian) features. DoG is a feature enhancement technique which involves taking the difference between two Gaussian blurred images. The formula for DoG is described in Equation 9 as :

$$g(x,y) = \frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}} - \frac{1}{2\pi K^2\sigma^2}e^{-\frac{x^2+y^2}{2K^2\sigma^2}} \tag{9}$$

Where σ is the Variance and $K > 1.0$ ensures the subtracted Gaussian has a larger variance than the initial Gaussian. For our implementation the dimensions of the Gaussian's were 7x7, $\sigma = 2$ and $K = 1.25$. Similarly to the RGB feature set, the DoG also returns 3 features per pixel. Thus after combining the RGB and DoG feature sets, each pixel has 6 features ($N = 6$).

3) MR8 Features: The final feature set we considered were the MR8 rotationally invariant features. These features are constructed by taking the maximal activations of the similar but rotated filters in the RFS bank. The RFS filter bank consists of 38 filters, which include both a Gaussian (Described in Equation 10) filter and Laplacian of Gaussian (LoG) (Described in Equation 11) filter, as well as, 3 bar filters and 3 edge filters each at 6 different orientations.

$$g(x,y) = \frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{10}$$

$$g(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right]e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{11}$$

For both the Guassian and the LoG filters the size was set to 7x7 and $\sigma = 3$. Similarly, for the bar and edge filters the size was also 7x7, while, $\sigma_x \in \{1,2,3\}$, $\sigma_y \in \{1,2,4\}$ and $\theta \in \{\frac{\pi}{6}, \frac{\pi}{4}, \frac{\pi}{3}, \frac{\pi}{2}, 0\}$ (Please note the equation for the bar and the edge filters can be found in the Appendix)

C. Hardware Description

- CPU 9th Generation Intel® Core™ i7-9750H 6 Cores - 12 Threads
- RAM 64 GB DDR4 2666 MHz
- OS: Ubuntu
- CUDA Driver Version / Runtime Version 11.1 / 10.0
- Name: "GeForce RTX 2060"
- CUDA Capability Major/Minor version number: 7.5
- Total global mem: 5926 MB
- Total constant mem: 65536 B
- The size of shared memory per block: 49152 B
- The maximum number of registers per block: 65536
- The number of SMs on the device: 30
- The number of threads in a warp: 32
- The maximal number of threads allowed in a block: 1024
- Max thread dimensions (x,y,z): (1024, 1024, 64)
- Max grid dimensions (x,y,z): (2147483647, 65535, 65535)

V. Evaluation Results and Discussions

A. Baseline

Below are the results for the baseline model. To measure the performance of the baseline model (on the respective feature sets), the Area Under ROC (Receiver Operating Characteristic) metric was used (AUC ROC).

Features	**ROC AUC Score**
RGB Features	**0.944**
RGB and DoG Features	0.931
MR8 Features	0.8969

B. U-net

1) Comparison of Learned Features and Handcrafted Features: The learned features do resemble the handcrafted features but are at times very different. The first row and the fifth row resemble features such as the prewitt filter that captures gradient information although the learned features capture diagonal gradients and the prewitt filters capture horizontal and vertical gradients. The other learned features do not resemble common handcrafted features. The second and last row have different features for each colour and hence have a much higher fidelity than what any person could hand craft in a feature set. The network was able to

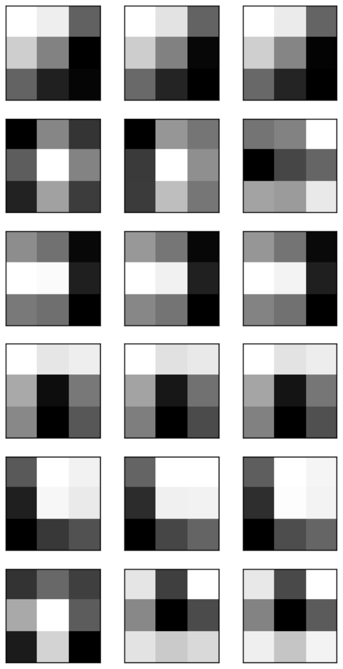

Fig. 2. Learned Features in U-Net

determine features for each respective colour channel that do not resemble any known filters through it's training process. This is evidence that the U-Net has far better discriminating power than a human's hand-crafted features and should always be used as it can learn features that predict better than hand-crafted features.

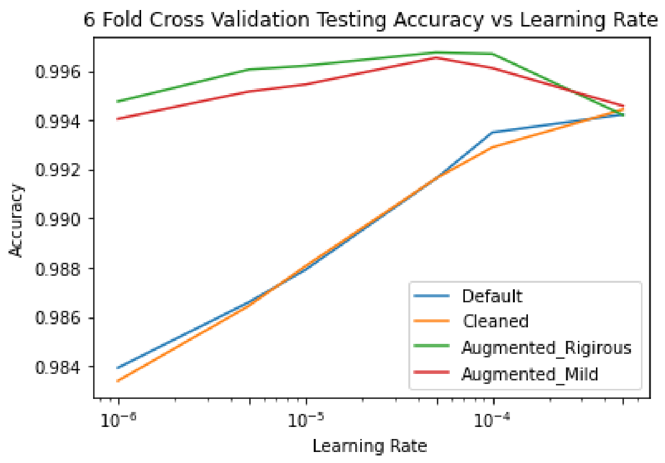

Fig. 3. 6 Fold Cross Validation Testing Accuracy of Respective Models at each Learning Rate

2) Learning Rate and Model Optimization: From Figure 3 the highest 6-fold cross validation accuracy on the testing data was 99.68% achieved by the rigorous augmented data with a learning rate of 5×10^{-5} peaking at the same learning rate as the mildly augmented data. The intensification of the augmenting of the training data led to a slight increase in accuracy overall as compared to the more mild augmented training data. The augmentation of data was highly necessary as it produced a significant increase in accuracy over the smaller training data set used in the other models. It can also be seen that the cleaning of the data led to a slight decrease overall in testing accuracy and is not recommended.

C. GMM

To optimise the number of clusters for both the foreground and background models, 6-fold cross-validation was done for each combination of 2, 4 or 6 clusters for both models (background and foreground). Rather than keeping the number of clusters consistent between models, we rather tried every combination for each model. To measure the model performance, the mean Area Under ROC (Receiver Operating Characteristic) metric over the 6 folds was used (On the varying number of clusters). The results of each of the for each feature is shown in Figures V-C, V-C and V-C.

1) RGB: For the RGB model, the highest mean *AUC ROC* score achieved was 0.97. This was achieved with 4 foreground clusters and 2 background clusters.

2) DoG RGB: For the RGB DoG model, the highest mean *AUC ROC* score achieved was 0.96. This was achieved with 6 foreground clusters and 6 background clusters.

3) MR8: Finally, for the MR8 feature set, the highest mean *AUC ROC* score achieved was 0.943. This was achieved with 2 foreground clusters and 4 background clusters.

D. Comparison of Models

From Table V-A and Figures V-C, V-C and V-C we see the *GMM* model outperformed the *Baseline* for every feature set. This was to be expected as the GMM model can use more than one Normal distribution to model the distribution of the foreground/background pixels. Although the overall performance of the models was as expected, the performance within each of the feature sets was surprising. The highest performing feature set, on average, was the RGB feature set, followed by the RGB DoG features and finally, the worst-performing features were the MR8 features. This was the exact opposite of what was expected, but the poorer results of the RGB DoG and MR8 feature sets can be attributed to the higher dimensional data. It is important to note, we also tested the use of principal component analysis (PCA), to reduce the dimensions of the MR8 features. We used 3 Principle components (which captures 96% of the variance). The results were very comparable to the unreduced, and in some cases performed worse. As such these results were omitted from the paper. Finally, within each feature set, we note that the number of foregrounds and background clusters affected the results for each of the feature sets.

- For the RGB feature set, we saw the highest performance was with 4 foreground clusters and 2 background clusters. This was to be expected as there are more patterns in the foreground than in the background, intuitively, this means more clusters would be needed to model the foreground than the background

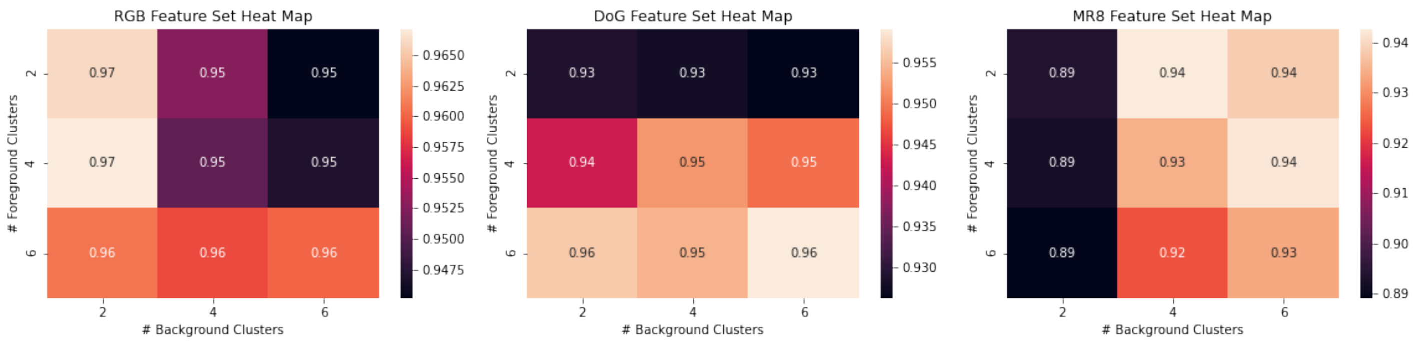

Fig. 4. Feature Heat Maps Results (a) RGB Results; (b) DoG Results; (c) MR8 Results.

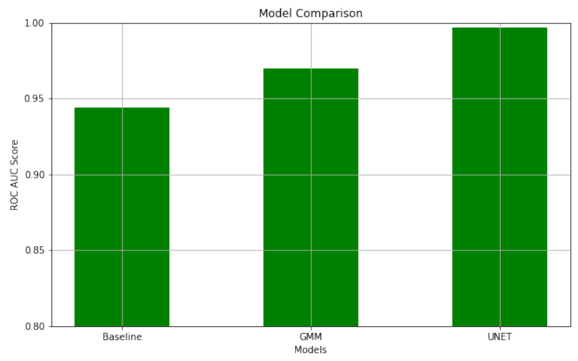

Fig. 5. Comparison of models utilizing ROC AUC Score

- Unlike the RGB features, the best results for the DoG RGB feature set were seen with 6 foreground clusters and 6 background clusters. This potentially could be attributed to the higher dimensionality of the data. (In future work it would be interesting to explore with a larger amount of clusters).
- Finally, the MR8 feature set performed the best when 2 background clusters and 4 foreground clusters were used. These results are also counter-intuitive, which again could be attributed to the higher dimensions of data. Further testing with a large number of clusters may have provided slightly different results.

The U-Net performed significantly better than both other models. The U-Net utilized more abstract features that do not resemble hand-crafted features and hence performed significantly better. The U-Net outperformed the best GMM model utilizing the RGB feature set. The U-Net was trained using the same input data in the form of the RGB channels. The features that were learned were more robust and had better discriminating power than a Gaussian mixture model as the amount of layers and convolutions in the U-Net is significantly more than the clusters in the GMM. This coupled with the reduced training time on a GPU allowed for the U-Net to train in comparable time and perform significantly better than the GMM. The U-Net was able to determine features for different colour channels and compound these features through many layers to achieve a near perfect ROC AUC Score of 0.997 utilizing 6-fold cross validation on the testing set. The score is significantly higher than any other model trained prior and has established the U-Net as the go to solution for semantic segmentation of images.

The U-Net in this paper outperformed the accuracy of the U-Net in the similar task of semantic segmentation of tumor detection which achieved an accuracy of 96.1% [1]. Similarly we both used transfer learning with a pre-trained VGG16 utilizing *imagenet*'s weights which were trained on a large set of natural images. Noticeably the segmentation network performed better on the puzzle pieces which are artworks than the CAT scans of brain tumors which are for the most part gray-scale. This suggests that the colour information is important in the process of semantic segmentation of an image and improves the overall performance. This can also be seen in the highest performing model of the GMM and baseline as they were utilizing RGB information as their feature set and performed the highest among the feature sets.

VI. Conclusion

In this paper we explored three different approaches to semantically segment puzzle pieces into foreground and background. The baseline model consisted of a single normal distribution to model each class and due to it's simplicity performed the worst as expected. The best performing GMM utilized 4 foreground clusters and 2 background clusters on the RGB feature set and was able to more accurately segment the data. It can be seen from the learned features that some colour channels required different features to the other channels and hence the U-Net having learnt these features performed the best. Finally, the best performing model was the rigorously augmented training data fed into the U-Net and achieved a 6-Fold-Cross validation accuracy on testing data of 99.68%.

References

[1] A. A. Pravitasari, N. Iriawan, M. Almuhayar, T. Azmi, K. Fithriasari, S. W. Purnami, W. Ferriastuti *et al.*, "Unet-vgg16 with transfer learning for mri-based brain tumor segmentation," *Telkomnika*, vol. 18, no. 3, pp. 1310–1318, 2020.
[2] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

Appendix

A. *Work distribution*

- Jamie Burns - GMM, Baseline model and related work
- Wesley Stander - U-Net and related work
- Both - Work that overlaps