Revisiting Background Segmentation Gaussian Mixture Models & Deep Learning

Sibusiso Mgidi
School of Computer Science
and Applied Mathematics
University of the Witwatersrand
1 Jan Smuts Ave, Johannesburg, 2000, South Africa
Email: 1141573@students.wits.ac.za

Neil Fabião
School of Computer Science
and Applied Mathematics
University of the Witwatersrand
1 Jan Smuts Ave, Johannesburg, 2000, South Africa
Email: 2216748@students.wits.ac.za

***Abstract*—In this computer vision project, we will investigate the use of semantic segmentation by predicting a segmentation mask on the puzzle piece images.The project explores computer vision techniques such as the Gaussian Mixture Models and U-Net deep learning model. In an effort to segment an image into foreground and background regions, our experiments were performed on 48 puzzle images. Since the dataset is relatively small, we used pre-trained weights from VGG16.**

1. Introduction

Computer vision is a field in computer science or artificial intelligence that investigates human vision, in an effort to replicate the process on computational machines. The term "human vision", does not simply mean the eyes or the ability to see images i.e taking a picture with your phone and display it on the screen. The purpose goes beyond replicating human ability of sight, including additional characteristics such as perception, that is the ability of humans to make sense of what they see. The most popular techniques used in computer vision including but not limited to image classification, object detection, semantic segmentation, instance segmentation, etc.
The aim of this project is to segment the puzzle piece in an image by predicting a segmentation mask on each puzzle image. This is a computer vision technique called Semantic Segmentation which classifies each pixel in an image to a predefined class. For this project, we have two classes namely the puzzle and background.
In this report, we implemented the GMM and U-Net in Python and Tensorflow respectively. Keras was used to apply data augmentation. The aim set out above was achieved by the following objectives:

- Implementation of the gaussian mixture models (GMM)
- Implementation of the U-Net
- We trained the U-Net model with pre-trained weights from VGG16.
- Applied 6-fold cross validation

For this project, we will answer the following questions:

1) Does GMM and U-Net segment puzzle pieces accurately?
2) Does data augmentation improve U-Net with a limited dataset?
3) Does k-fold cross validation increase models accuracy?

2. Related Works

The Mixture of Gaussian's or mostly mentioned as Gaussian mixture models is a probabilistic statistical technique that is used for image segmentation, implementations and improvements of this approach can be seen on the paper by [4] and [5]. The utilization of the HSV color space for enhancing background subtraction of floating objects in water was and improvement in noise reduction using modified GMM was successful.

3. Dataset

For this project, we used the puzzle dataset from the computer vision course. The dataset consists of 48 pre-processed puzzle pieces with corresponding masks having dimensions (768, 1024,3) and shape(768, 1024) respectively.

Figure 1. An example of a puzzle piece along with its corresponding binary mask.

3.1. Image Preprocessing

The puzzle images are dived into training 70%, validation 15%, and test 15 % images. This proportion yields 34

training images, 7 validation, and 7 test images.
For the GMM the pre-processing done in order to increase processing time was to simply downsample the image from its original size 1024x768 to 410x307 which corresponds to a 40% decrease.
On the other hand for the U-Net, we performed horizontal flip data augmentation increasing our training puzzle images from 34 to 68 images.

4. Methodology

The present section will follow on the ideas depicted in the previous section and will provide an indept insight into the segmentation techniques that will be used in this Honours research.

4.1. Gaussian Mixture Models

A computer understands that an image is composed of different pixel values which have different pixel bins, these are then stored in a matrix and then are read and comprehended by the computer.

4.1.1. Normal distribution. With the understanding that an image is simply a matrix with different pixels values and intensities, one can make use of probabilistic approaches to model this type of data. One of the introductory unsupervised statistical methods used during COMS4036A laboratory was fitting image data with a normal distribution, retrieving the mean and covariance vector which are used to calculate the probability of the data x and parameterize this by the world state w, with a multivariate normal distribution as seen below :

$$Pr(\mathrm{x} \mid w) = Norm_{\mathrm{x}}\left[\mu_w, \Sigma_w\right]$$

The normal distribution has its disadvantages one of which is being unimodal and neither foreground nor background regions are well represented by a pdf with a single peak and secondly not being robust even a single outlier can affect the estimates of the mean and covariance of our distribution.

4.1.2. Gaussian mixture models. Gaussian mixture models or simply referred to as Mixture of Gaussians is an unsupervised learning approach that uses a group of Gaussian or group of different Multivariate normal distribution, all with different mean vectors μ and different covariance matrices Σ per hidden variable as seen on Figure 2.

An example of a GMM algorithm is demonstrated in the paper by [5], follows the iterative process of Expectation and Maximization as seen below :

In this algorithm, the E step tries to find a function for the expectation of the log-likelihood evaluated using the current estimate for the parameters in this case the hidden variables present in the feature vector. This can also be said that it tries to calculate the responsibilities denoted as p_{ij} that is also Bayes rule as depicted in [5] and in [1] is denoted as r_{ik}. Lastly, the M step computes parameters maximizing the expected log-likelihood found on the E step,

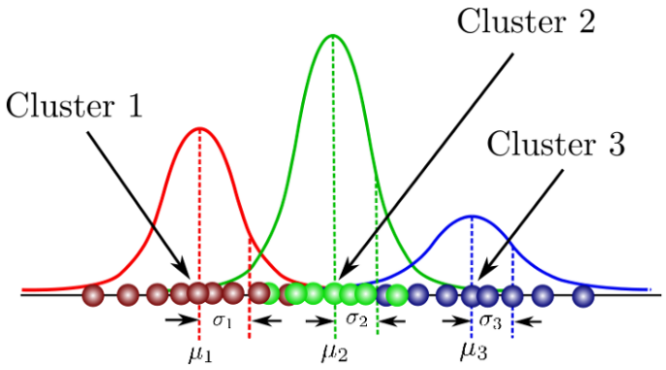

Figure 2. Example of GMM distribution

Algorithm 1 Gaussian mixture model EM-MAP algorithm

Input: Image feature vector
Output: Model for the segmented image

Initialisation : $\theta = (\mu_1, \ldots, \mu_k, \sigma_1, \ldots, \sigma_k)$

1: (E-step)

$$p_{ij}^{(r+1)} = P^{(r+1)}(i \mid x_j) = \frac{p_i^{(r)} N\left(x_j \mid \mu_i^{(r)}, \sigma_i^{2(r)}\right)}{f(x_j)}$$

2: (M-step)

$$\hat{p}_i^{(\mathrm{r}+1)} = \frac{1}{n}\sum_{j=1}^{n} p_{ij}^{(r)}$$

$$\hat{u}_i^{(r+1)} = \frac{\sum_{j=1}^{n} p_{ij}^{(r+1)} x_j}{n\hat{p}_i^{(\mathrm{r}+1)}}$$

$$\hat{\sigma}_i^{(r+1)} = \frac{\sum_{j=1}^{n} p_{ij}^{(r+1)}\left(x_j - \hat{\mu}_i^{(r+1)}\right)^2}{n\hat{p}_i^{(x+1)}}$$

3: **while** $\sum_i e_i < \varepsilon$ **do**
: *Iterate steps 1 and 2 until convergence*
4: **end while**

in other words, it tries to converge to the best possible values given the responsibilities calculated in the E-step such that the function is able to converge in the desired tolerance or epsilon value.

4.2. U-Net

We will also exploree the semantic segmentation deep learning technique known as the U-Net work by [3], which has demonstrated great impact in the area of biomedical images. U-Net model extends the Fully Convolutional Network (FCN) by [2], which learns to segment objects in an image in an end-to-end setting. It takes in an image of any size as an input and produces segmentation map with efficient inference and learning. The network architecture consists of the encoder path and the decoder path.

The encoder path has a sequence of 3 x 3 unpadded convolutional layers with ReLU activation followed by a max-pooling (2 x 2 kernel with 2 strides) for downsampling input images. The network applies max pooling on each channel of the input activation map separately in order to produce an output activation map. In the decoder layer, the feature maps from the encoder layer are expanded using 2 x 2 transposed convolutions. The feature maps are then

concatenated to upsampled decoder feature maps to generate a re-scaled high resolution segmentation map and a class for each pixel [3]. A schematic representation of the network architecture is shown in Figure 2.

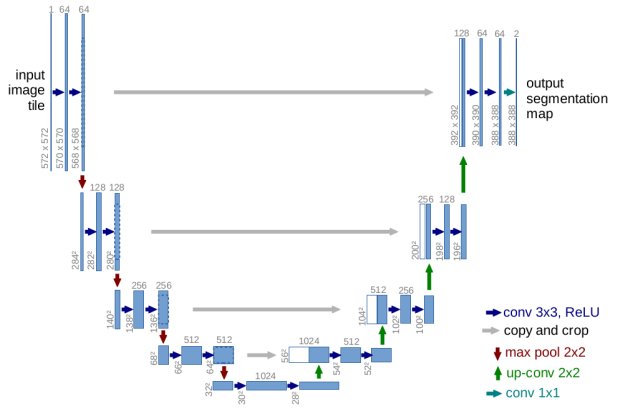

Figure 3. Illustration of U-Net architecture [3]

5. Results and Discussion

In this section, we analyze experiments and findings from the GMM and U-Net deep learning model separately followed by a comparison of the results from the two models.

Evaluation Metrics. To evaluate the performance of the models, one prominent point made by the lecturer was that when working with the predictive models given a ground truth it was not advised to rely merely on the percentage accuracy given by the model but thus inspecting the generated image and understanding how the true positive and true negative cases perform during the inference of the model. For this particular reason, the metrics used for this research project when applying the GMM were Cohen's kappa, confusion matrix, F1 score, and IoU(intersection over union). Most of the training was done with the training data and followed by an evaluation of the models using the validation data. On the 0ther hand, to evaluate the U-Net model we calculated the IoU for each image in the testing dataset and then averaged it to get the IoU metric value. The IoU measures the number of pixels common across the target masks and prediction masks divided by the total number of pixels present in all masks.

5.1. Gaussian mixture models

Three feature vectors were used during the testing of the GMM which the first only had RGB, the second RGB plus DOG (difference of Gaussians) as instructed in the handout and lastly, the extra feature to test was the HSV plus DOG. The reason for choosing HSV plus DOG was to understand how the Hue, saturation and value colour space could be improved since only testing the HSV by itself it demonstrated reasonable results but not in par with the RGB colour space, this intrigued to understand how this colour space can be used and what features would it mostly capture in a puzzle piece.

5.1.1. Results. The following observations were made during the evaluation of the GMM implementation using 35 epochs:

1) The first point to mention is that the mixture of Gaussian's outperforms the normal distribution, as explained in the methodology section with the main difference between the two is that instead of just having one normal distribution that will not fit the histogram of the feature vector appropriately, in turn, multiple normal distributions fit the centres of the different cluster present in the histogram. The result presented in Figure 14 in the appendix, where the image produced using only the RGB with one normal distribution did not demonstrate better results compared to the mixture of Gaussian's and similar point can be said in regards to the HSV colour space. Furthermore training with the GMM is more time and computantionaly intensive than just one normal distribution, since for instance after reducing the image to 40 percent the GMM for RGB running from 1 to 9 components can take 4 to 5 hours while the HSV can take 8hours.
2) Secondly, when choosing the convergence threshold or epsilon, most models will achieve a decent result with 0.01 but in order to reach the best results, one may need to use a threshold of 0.001. This implies that when using a threshold of 0.001 the convergence time may be longer especially when the random means are generated are far from the difference of 0.001, for instance, starting with a difference of 0.10 or 10 will take longer periods to converge as seen on Figure 13 in the appendix.
3) The third point is that the number of components highly depends on the image in question and there is no best number. For example, in the puzzle piece, as seen on Figure 15, there is the presence of hair, the colour blue, and pink, which this, in turn, may affect the result of the segmentation process. When clustering with 4 components this yielded the best performing model which presented Cohen's kappa score of 0.5156 and the IOU of 0.3206. While on the other hand, the number of false positives was 12276 and false negatives 11164. On the other hand clustering with 2 components, this particular image the scores were 0.5099 for Cohen's kappa score which is within the range of 0.50 - 0.40 considered a reasonable result, while the IOU being 0.3186 considers that our predicted mask is not close to the ground truth. With this understanding the for the confusion matrices generated similar conclusion can be made of clustering with 2 components with

false positives of 12486 and false negatives 11257 it is possible to conclude that a GMM using 2 components only present decent results in this particular case.

4) When testing the HSV plus DOG feature vector the considerable result was achieved compared with the normal distribution of the same feature but was not in par with the RGB colour space. The worth mentioning point when working with HSV colour space is that the boundary of the image is always captured with ease but when it comes to noise in the background the model does not accurately capture the features this may be given to the fact that this colour space intensifies some of the lighting and other characteristics of the picture. This implementation for the image in Figure 14 yielded Cohens kappa score of 0.7348 while best IOU was 0.4056 and best clustering number was 5 Gaussians.
5) Lastly the models using 6 fold cross validation performed better than expected, but due to time constraint the models that finished training on time were the RGB and RGB plus DOG. The inference for RGB using the cross validation as seen on Figure 16 yielded a greater true positive with the sum of the 6 confusion matrices achieving 562389 true positives and 185784 false negatives, while the Cohen's kappa score of 0.97 and F1 score of 0.98 which this lead to consider that this is good model. Although when evaluating the intersection over union the RGB model it only produced 0.49 which is close to a average model while for the RGB plus DOG as seen on Figure 17 produced an IOU with 0.47 leading to conclude that this is not the best model.
6) For images converted to the HSV colour space it is important to point our that as we increase the number of components the clutering algorithm tends to perform better than when just with few mixtures of Gaussians and this can be seen when using only 3 GMMs on Figure 17 and 5 GMMs on 15. This may be due since the HSV color space takes longer to train approximately 20 to 30 for 2 GMMs and as the number of GMMs increases so does the time to train and efficiency of the model.

5.1.2. Discussion. When fitting the GMM there may be a case where the data points produced by the Gaussian distribution may produce peaks or spikes and not a normal bell curve [6]. This is denoted as singularity issue, it mostly occurs when the multivariate normal distribution tries to model a point and the values produced return variance of zero. To resolve this and other issues, the following techniques were employed during the implementation of this project :

1) Resetting the mean and variance when singularity occurs. When in the M step or the maximum likelihood, one can detect if the variance or mean has a value of zero, once this occurs a reset of the two parameters will be deployed, to ensure random datapoint using the image feature vector will be assigned to avoid reaching the values where the function found the undesired peaks. This resets is a heuristic method presented in [6].
2) Using MAP instead of MLE by adding a prior. The second and recommended approach is to add in the EM step algorithm the MAP (maximum posterior), which for the models generated a prior is defined over the feature vector, the E step remains the same since the priors are fixed the only change comes in the update rule of the parameters for the distribution. In this new update rule we estimate the unseen parameters with the responsibilities generated in the E step and the M step in a special case for the covariance there is a need to multiply the identity matrix by another large positive number such that the initialisation gives the distributions a large covariance [6].
3) The other improvement that can be made to the GMM is to initialise the means with Kmeans clustering, this ensures that the centre points are within close range of the possible centres in the data for the specified number of components specified. When running on random clusters centres the issue that occurs is that it may take longer periods to convergence as seen on Figure 12 understanding that the point in which the cluster was assigned is far from the actual data points that exist in the feature vector.
4) The last issue found during this project was that when calculating the responsibilities for the background model using HSV colour space, the values achieved did not have a consistent shape with the values calculated using the model with the puzzle pixels. One responsibility array had the length of 5 while the other just 1, the reason for this is not yet identified but a solution found was to use the model for the puzzle pixels to infer using this mean and covariance matrix achieved by the feature vector for the background pixels and the results produced were an improvement as compared to the normal distribution as seen on Figure 14 in the appendix.

5.2. U-Net

The implementation of the U-Net deep learning model in this study follows the implementation of the original paper by [3], with just a few small edits to suit our dataset better. The model was trained on a GTX 1060 machine with 6 GPU memory, Tensorflow, Python3, and Keras2.

5.2.1. Experiments/Discussion. We ran 4 sets of experiments each with results reported in table form, which is listed below. These 4 batches consist of experiments covering our research question. Thus, we included U-Net without

data augmentation as seen in Table 1 and finally, U-Net with data augmentation as seen in Table 2. In each experiment performed k-fold cross-validation and evaluated how this influenced the model's performance. It is worth noting that for all experiments performed we used pre-trained weights from VGG16 and fine-tuned the network to fit our dataset. The input image size and batch size dramatically impacted model run time and memory. The original puzzle image dimensions are 768 $\times$ 1024. In order to obtained increased speed, the images were down-sampled to 192 $\times$ 256.
We carried out each experiment with Adam optimizer and found out that a learning rate of 0.001 yields good results for the deep learning model. We also used the stochastic gradient descent optimizer but it did not yield promising results so for both U-Net with and without data augmentation we used Adam optimizer.

5.2.2. Results. The results of the hyperparameter training and IoU measure from various training of U-Net without data augmentation and with data augmentation are summarized in Table 1 and Table 2 respectively.
For our initial experiment, we ran 3 batches of experiments. Firstly, we ran the U-Net without data augmentation and for batch size, we began with a small size of 2 simply because our dataset is not large enough. However, we found after a few trials that a lower batch size yielded better results because our validation dataset has 7 images. In spite of this, for our final tests, we varied our batch size in the range [2,4,6]. Due to the limited GPU memory, we limited the batch size to 6 and used a maximum of 50 epochs. The IoU is calculated for each image in the testing dataset and the overall IoU is averaged over 7 testing images to 50 epochs. It can be seen from the results that the IoU metric increases from 0.92 to 0.97 with an increase in batch size as seen in Table 1. After the initial experiment, we discovered that the U-Net gives promising results with IoU score of 0.97 on the testing dataset. In spite of this, the model was retrained with a batch size of 6 and we saw a decrease in IoU score.

IMAGE SIZE	BATCH SIZE	TIME(S) TO 50 EPOCHS	IOU
192 $\times$ 256	2	944.75	0.92
192 $\times$ 256	4	1270.05	**0.97**
192 $\times$ 256	6	1343.25	0.96

TABLE 1. THE OVERALL IOU AND TIME TO 50 EPOCHS ON TESTING SET.(**WITHOUT AUGMENTATION**)

In Figure 5, 6, and 7 the learning curve at batch size of 4 for best U-Net model without data augmentation on training and validation dataset can be seen. The Model loss, accuracy, and IoU score are plotted against the number of epochs. It is important to note from the Figure 5, 6, and 7 that the model is overfitting at about epoch 7. This is presumably attributed to the fact that our dataset is not large enough hence the model does not generalize well.

Due to the limited dataset size, it was possible to evaluate the distribution of IoU for individual images. This can be seen in Figure 4. Out of 6 testing images, only a single image with the least IoU value was select from the sample to easily evaluate if whether the IoU score increases with data augmentation or not.

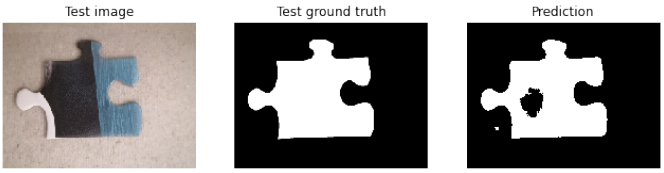

Figure 4. An example of a test image, a corresponding mask/label and U-Net's without augmentation prediction at batch size = 4 and IoU: 0.92 (experiment 2 in Table 1)

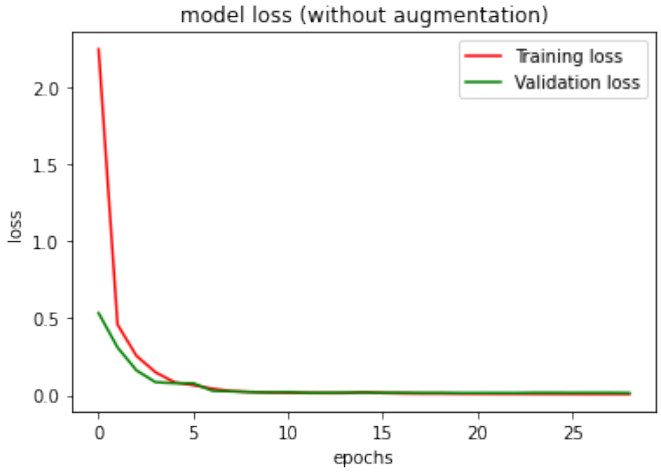

Figure 5. Model loss at batch size = 4 for best U-Net without data augmentation on training and validation dataset

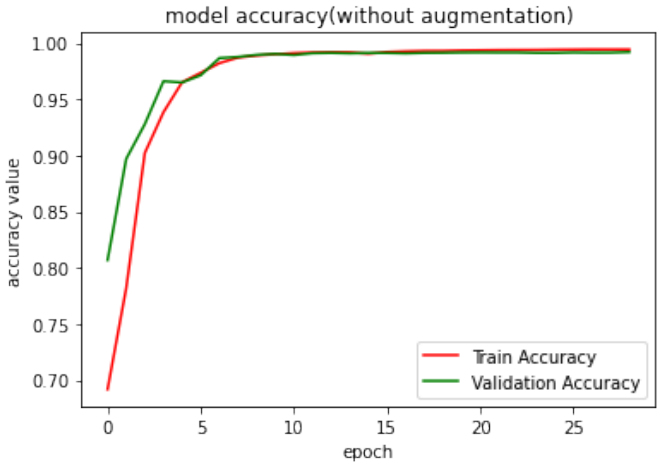

Figure 6. Model accuracy at batch size = 4 for best U-Net without data augmentation on training and validation dataset

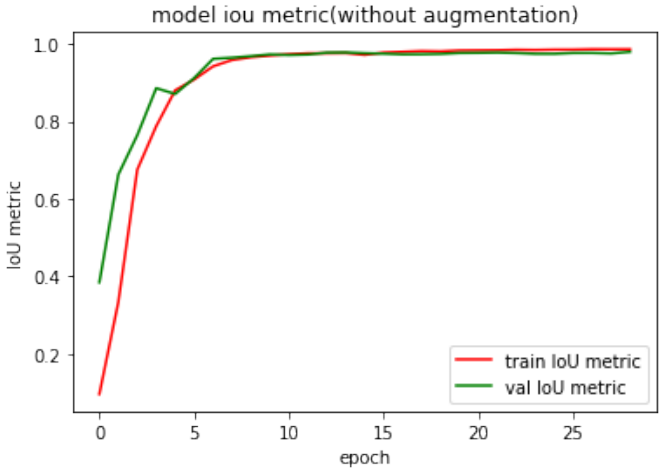

Figure 7. Model IoU score at batch size = 4 for best U-Net without data augmentation on training and validation dataset

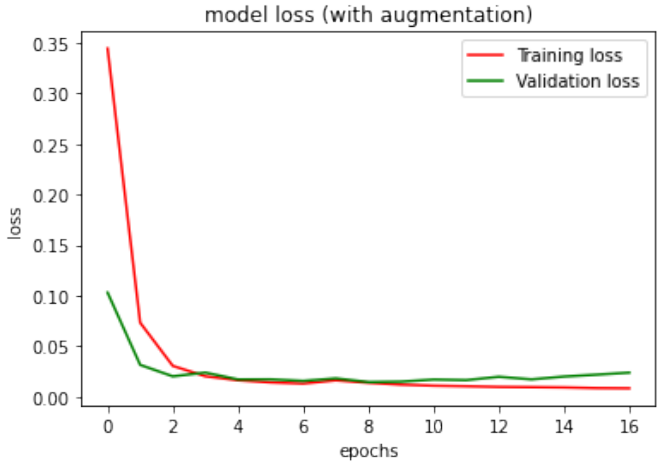

Figure 8. Model loss at batch size = 4 for best U-Net with data augmentation on training and validation dataset

We also retrained the U-Net with our horizontal flip data augmentation pre-processing pipeline. When data augmentation is applied, we saw an increase in IoU from 0.97 U-Net without augmentation to 0.98 U-Net with augmentation as seen in 2. This is a clear indication that while the U-Net pre-trained with weights from VGG16 works better with a limited dataset, it also benefits from data augmentation. Using U-Net with data augmentation we also evaluated how

IMAGE SIZE	BATCH SIZE	TIME(S) TO 50 EPOCHS	IOU
192 × 256	2	1110.9536	0.94
192 × 256	4	1579.639	**0.98**
192 × 256	6	1401.444	0.97

TABLE 2. THE OVERALL IOU AND TIME TO 50 EPOCHS ON TESTING SET.(**WITH AUGMENTATION**)

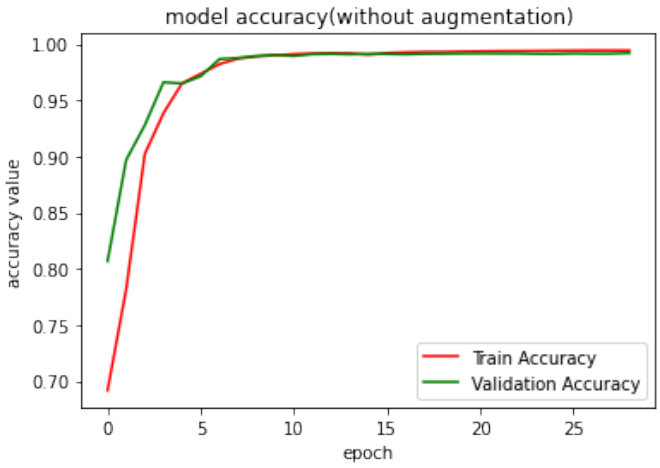

Figure 9. Model accuracy at batch size = 4 for best U-Net with data augmentation on training and validation dataset

the model performs in training and validation dataset while observing any signs of overfitting as seen in Figure 8, 9, and 11. From the learning curves, it can be that the model starts to overfit in from about epoch 9. This is due to the fact that the deep learning models require an extremely large dataset to generalize well.

Due to the limited dataset size, we also evaluated the distribution of IoU for individual images using the U-Net model trained with data augmentation. For the sake of consistency, we used the same image used in the U-Net without augmentation 1. It is evident from the test image that the U-Net with data augmentation yields better results compared to the U-Net without data augmentation as seen in 10.

Figure 10. An example of a test image, a corresponding mask/label and U-Net's with augmentation prediction at batch size = 4 and IoU: 0.98 (experiment 2 in Table 2)

Lastly, we also applied 6 fold cross-validation on the U-Net with augmentation and without augmentation. The carefully evaluated how both models perform looking at the accuracy and loss per fold. For U-Net without augmentation, we achieved an accuracy per fold in the range [99.341, 98.809, 99.551, 99.324, 99.573, 99.564] and the loss per fold [0.013, 0.073, 0.005, 0.02, 0.004, 0.004] for each fold from 1 to 6 respectively. It is evident that the U-Net without augmentation yields higher accuracy at fold 6 and the corresponding loss is also binary cross-entropy loss it is also low which clearly indicates that the model gives good results in a limited dataset. On the other hand, for U-Net with augmentation we achieved an accuracy of [99.1505, 99.262, 99.302, 99.366, 99.397, 99.325] and loss per fold of

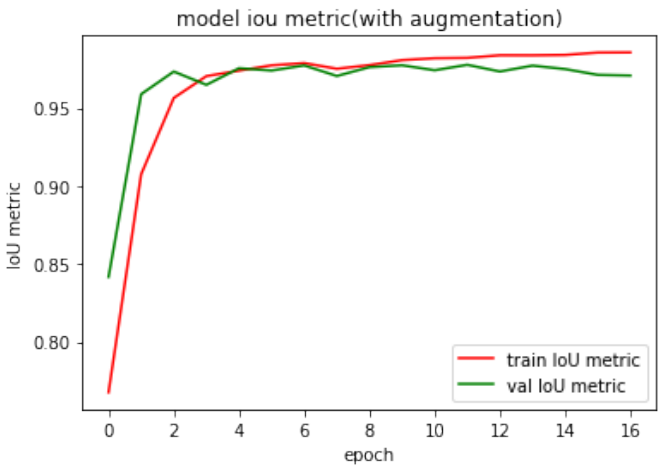

Figure 11. Model IoU score at batch size = 4 for best U-Net with data augmentation on training and validation dataset

[0.017, 0.013, 0.011, 0.010, 0.010, 0.011] from fold to fold 6 respectively. From the evaluation metric, it is evident that the U-Net with augmentation achieves the highest accuracy at fold 5.

6. Conclusion

We used the IoU metric value to compare the performance of the GMM vs. the U-Net model. The GMM across different features achieved an IoU between [0.40 - 0.42] whereas, the U-Net produced an average IoU of 0.98 with data augmentation. Unfortunately, we did not have enough time to experiment with other data augmentation techniques and evaluate how this would impact the performance of the U-Net. Therefore, we consider using U-Net for the puzzle image semantic segmentation problem because it yields better results compared to the GMM. This is because the U-Net can deal much better with a limited dataset than the GMM. Also, Applying transfer learning from the VGG16 pre-trained weights served as an advantage to the U-Net as opposed to using handcrafted features. From the results of the Mixture of Gaussian's, it is possible to understand that modeling an image with multiple normal distributions instead of only one will allow the final model to be able to capture all the possible components existent in the image. In addition to this training, an image using HSV color space may not be the best practice unless ones have a particular task that involves capturing certain image intensities that may not be able to be noticed using the RGB color space as mentioned in [4]. On the other hand, we believe that more work can be done to boost the performance of the U-Net to give better results for images with high and low contrast by exploring different types of data augmentation.

7. Contribution and Work Breakwdown

Implementation of the GMM is contributed by Neil Fabião whereas the U-Net model is built by Sibusiso Mgidi. Both team members contributed equally in writing the project report.

References

[1] Simon J. D. Prince. 2012. Computer Vision: Models, Learning, and Inference (1st. ed.). Cambridge University Press, USA.

[2] Long, Jonathan, Shelhamer, Evan, and Darrell, Trevor.Fully convolutional networks for semantic segmentation.InCVPR 2015, 2015.

[3] Ronneberger, Olaf, Fischer, Philipp, and Brox, Thomas.U-net: Convolutional networks for biomedical imagesegmentation. InMICCAI, 2015

[4] X. Jin, P. Niu and L. Liu, "A GMM-Based Segmentation Method for the Detection of Water Surface Floats," in IEEE Access, vol. 7, pp. 119018-119025, 2019, doi: 10.1109/ACCESS.2019.2937129.

[5] Farnoosh, R. and Zarpak, P.B., 2008. Image segmentation using Gaussian mixture model.

[6] Christopher M. Bishop. 2006. Pattern Recognition and Machine Learning (Information Science and Statistics). Springer-Verlag, Berlin, Heidelberg.

Appendix

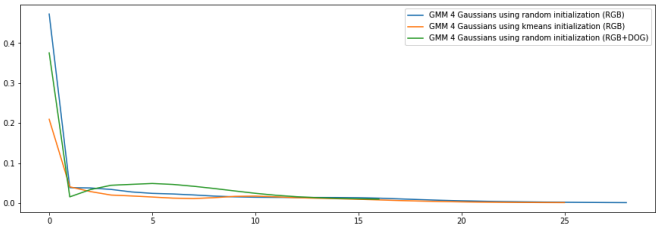

Figure 12. Plot of convergence using Kmeans clustering with tolerance 0.001

Figure 13. Best generated mask from puzzle piece with multiple components with tolerance 0.0001

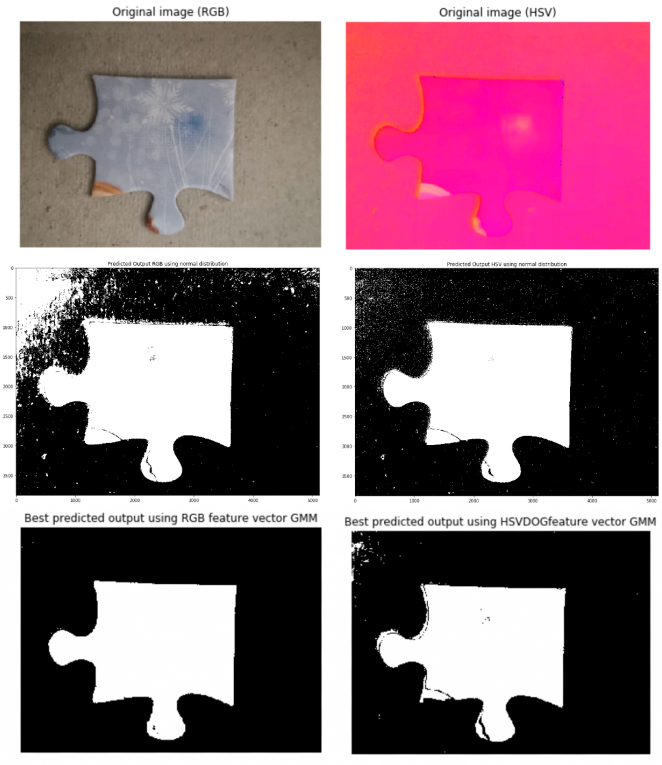

Figure 14. Comparing normal distribution and mixture of gaussians

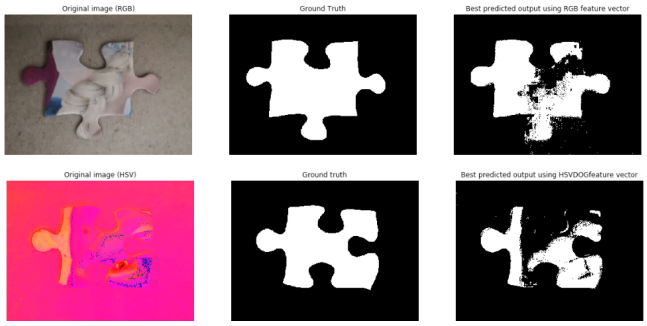

Figure 15. Example of puzzle piece with multiple components

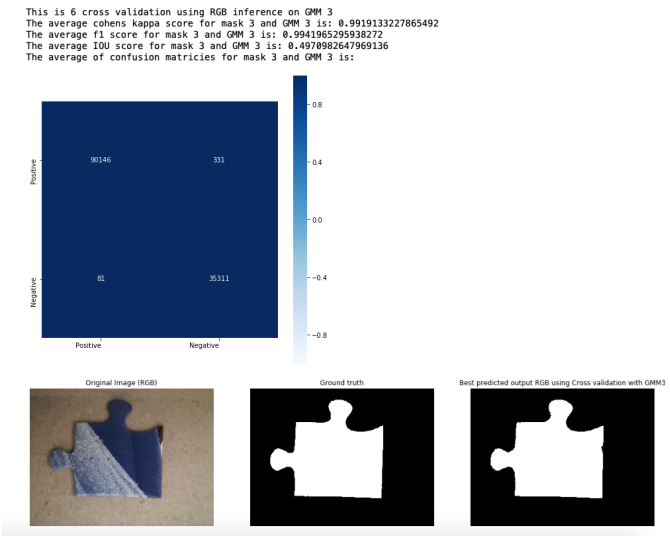

Figure 16. 6 cross validation using RGB feature vector

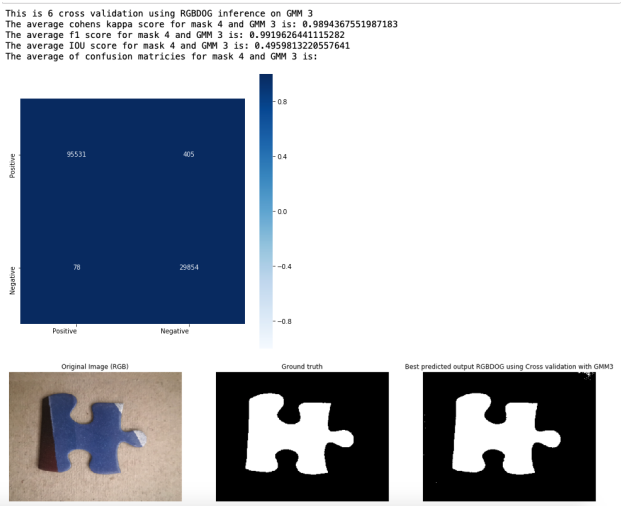

Figure 17. 6 cross validation using RGBDOG feature vector