Final Report - Image 6

names

Executive Summary

Short description of the problem. The main findings. Key figure if appropriate. The practical relevance of the analysis.

Aim and Background

A clear description of the problem, articulating the aim of this project. Provides appropriate multidisciplinary context and motivational background explained well in an appropriate language. Including background of the data

Method - Data Collection & Developed Models

The dataset was generated by 10x Genomics Xenium instrument on a fresh frozen mouse brain coronal section. The data bundle contained various components, including a cell morphology image where the intensity corresponded to the presence of the nucleus in each cell, cell boundaries indicating the spatial locations of the detected cells, and RNA abundances for each cell. The cells were then grouped into 28 distinct clusters, and the cluster labels were provided (Reference to data bundle). An Rmarkdown file was employed to generate the images. This process involved utilising the cell boundaries of the detected cells along with their corresponding labels to create per-cell images for a randomly selected subset of 1000 cells.

Prior to introsucing augmentation, several preprocessing steps were employed to maximize the effectiveness of our approach. These steps were undertaken to optimize the dataset before the introduction of data augmentation. The preprocessing steps included the exclusion of noise beyond the cell boundary vertices associated with the cells. This process ensured that the resulting images exclusively contained the cell itself, without any extra elements or noise. Additionally, the images were subjected to a mask and resize, ensuring that each cell shared a uniform size and eliminated any inconsistencies in the image generation process. The pixel intensities were also centered in order to maintain consistency across all images. By considering the substantial number of cell clusters present in the dataset, the image data was augmented by doubling the samples from 1000 to 2000. This augmentation technique was implemented to expand the sample size while maintaining a minimal increase in training time.

Baseline Model

Initially, a Random Forest classifier was employed, utilising the pixel values of the provided images, to predict the corresponding cell clusters. However, the limited complexity of the model hindered its capability to capture the composite shapes of the cells, resulting in a error rate of 95%. Also, the high number of clusters within the dataset, totaling 28, further constrained its predictive performance. The reliance of handcrafted features or simple statistical measures from the Random Forest classifier was not effective in capturing the complex and abstract features present in images, leading to sub optimal performance in image classification tasks. This lead to the introduction of CNN.

(Talk about why CNN is optimal - using some literature review)

The CNN model architecture implemented in this study comprised of two convolutional layers. The first convolutional layer consisted of 32 filters, followed by a pooling layer. Subsequently, the second convolutional layer was comprised of 64 filters, followed by another pooling layer. The output from the second convolutional layer was then flattened into a one-dimensional form, preparing it for the subsequent classification process. For classification, fully connected layers and dropout layers were employed to mitigate overfitting and enhance generalization capabilities. There were in total 1,632,962 trainable parameters available. A binary cross-entropy loss function was employed, along

with the Adam optimiser. Prior to feeding the image into the CNN model, a transformation step is required to convert the images into a four-dimensional array. This array consists of two size 64 arrays, which correspond to the dimensions of the image. Typically, for RGB images, three size 64 arrays are utilized. However, for black and white images, two size 64 arrays are sufficient to represent the image data. (refer to literature). Given the constraints of limited computational resources, a shallow architecture was deliberately chosen for this study. This decision aimed to accommodate the resource limitations associated with deeper CNN networks and expedite the training process, ultimately achieving interpretable results within the given constraints.

Model Variations

These model variations offer alternative and more dependable viewpoints regarding the impact of data augmentations on CNN models. They also provide a platform for exploring optimal configurations and settings for future studies in this field. By examining different approaches and their outcomes, these variations contribute to a deeper understanding of the role and effectiveness of data augmentations in CNN models. In total, 6 models were used for each augmentation.

Class Weights

In this particular model variation, class weights were introduced during the training process. This approach was adopted to address the substantial class imbalance observed in the training set. By assigning class weights, greater emphasis was placed on classes with a lower count, while reducing the importance of classes with a higher count. To ensure flexibility and applicability in future implementations, the class weights were determined based on a ratio between the count of a specific class label and the total class count. (some literature for class weights)

Categorical Cross-Entropy Loss Function

For this specific model variation, a transition was made from using binary cross-entropy to categorical cross-entropy as the loss function. Although both loss functions share similarities, categorical cross-entropy is more commonly employed in multi-class classification scenarios involving more than two classes (reference to literature). It quantifies the dissimilarity between the predicted class probabilities and the true class labels. This differs from binary cross-entropy, which measures the disparity between the predicted probabilities and the true labels. It can be seen later in the results section that the validation loss produced by these two loss functions were required to be interpreted in two different scales.

RMSprop Optimiser

Finally, this model variation utilises RMSprop optimiser instead of the Adam optimiser in the original model. These optimisers differ in many aspects, including their update rule, bias correction and convergence methods. (From this literature study), although Adam is generally considered to have faster convergence and better performance on a wide range of tasks, it may be more sensitive to hyper parameter settings and can sometimes overfit or converge to suboptimal solutions. RMSprop, being a simpler algorithm, may be more robust in certain cases and requires fewer hyper parameters to tune.

Method - Evaluation Strategies

During the implementation process, the augmented training set was incorporated alongside the original training set. The rotating and resizing augmentations were performed using the EBImage package (reference to EBImage package). To introduce Gaussian noise, a randomly generated number from a normal distribution with a specified mean and a standard deviation of 0.2 was added to each image. The augmented training set was subsequently utilised in training each variation of the CNN model, with the validation loss recorded for each epoch and stored in a CSV file for further analysis and evaluation.

To comprehensively assess the impact of each augmentation, different noise levels were carefully selected. For Gaussian noise, three distinct subcategories were incorporated: low (0.2 mean), high (0.8 mean), and random (0.2 to 1 mean). Similarly, the same approach was applied to rotation and resizing augmentations. Three subcategories were defined: low (90 degrees, 16x16), high/medium (180 degrees, 32x32), and random (1 to 359 degrees, low or medium resolution). Resizing were limited to below 64 due to the lack of improvement of quality past that. By

adopting this strategy, a more insightful analysis of the effect of each augmentation on the model's performance and robustness was facilitated.

Results - Effect of Data Augmentation

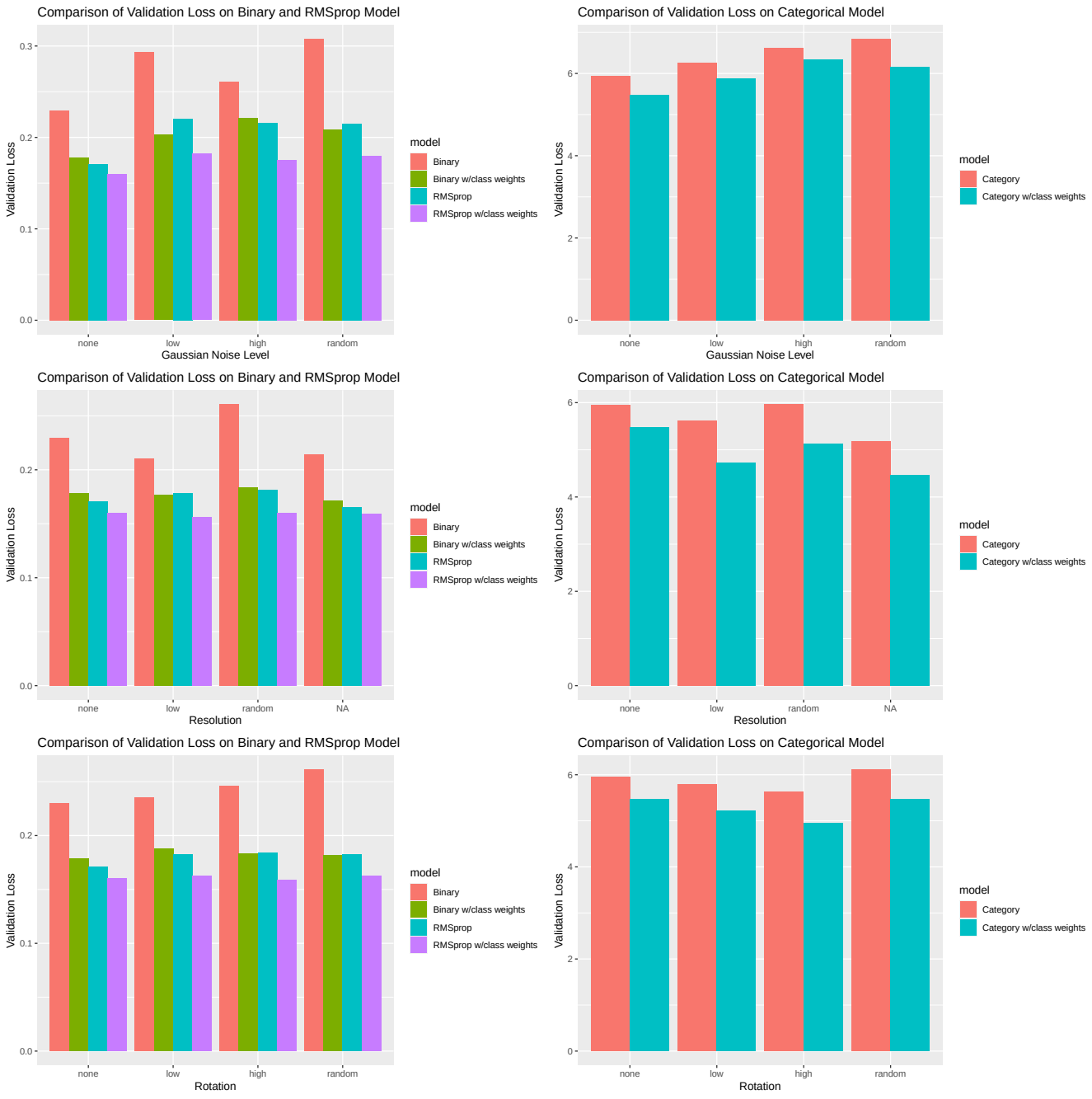

Figure 1: Validation Loss for Different Gaussian Noise Levels on CNN Models.

Gaussian Noise

Observing Figure 1, the introduction of Gaussian noise in the implementation resulted in an increase in validation loss across all models, as compared to the original model. Notably, the application of random mean Gaussian noise yielded the largest validation loss for both the binary and categorical models. Moreover, a high mean Gaussian noise of 0.8 was applied led to the highest validation loss for both the binary and categorical models with class weights.

Finally, the inclusion of low mean Gaussian noise demonstrated the highest validation loss for the RMSprop models, both with and without class weights.

Resolution

The incorporation of low (16x16) and medium (32x32) resolutions in the training set resulted in a reduction of validation loss across all models, as illustrated in Figure 2. Notably, the categorical models with class weights experienced the most substantial benefits from this augmentation, with a decrease of 0.7503 when low resolution was added and a decrease of 1.0054 when medium resolution was utilized. Conversely, the introduction of random resolution into the training set led to an increase in validation loss, particularly noticeable in the binary model, with an increase of 0.0309.

Rotation

While the binary and RMSprop models showed an increase in validation loss with rotation, the categorical models, both with and without class weights, demonstrated a decrease in validation loss when low (90 degrees) and high (180 degrees) rotations were applied, as shown in Figure 3. Models with random image rotation experienced the highest validation loss across all models.

Table Summary of Validation Loss for Every CNN Model

		Binary	Binary w/class weights	Cateogorical	Categorical w/class weight	RMSprop	RMSprop w/class weight
		Val Loss	Val Loss	Val Loss	Val Loss	Val Loss	Val Loss
No Noise		0.2298	0.185	6.1524	5.5162	0.1718	0.1599
Gaussian Noise	0.2	0.2930	0.2029	6.2574	5.8756	0.2202	0.1821
	0.8	0.2607	0.2210	6.6276	6.3434	0.2156	0.1751
	Random	0.3078	0.2086	6.8412	6.4580	0.2149	0.1800
Resolution	16x16	0.2101	0.1770	5.6171	4.7255	0.1782	0.1563
	32x32	0.2145	0.1714	5.1790	4.4704	0.1652	0.1592
	Random	0.2607	0.1840	5.9652	5.1211	0.1813	0.1602
Rotation	90	0.2355	0.1878	5.800	5.2267	0.1823	0.1628
	180	0.2460	0.1836	5.63470	4.9477	0.1840	0.1585
	Random	0.2611	0.1818	6.1100	5.4714	0.1824	0.1629
Combined Model		0.2084	0.1769	4.8067	4.6177	0.1891	0.1601

Figure 2: Summary Table of Validation Loss for All CNN Models.

Combining Data Augmentations

The combination of all data augmentations resulted in a reduction of validation loss across all models, except for the RMSprop model without class weights, as indicated in the summary table presented in Figure 4. Additionally, Figure 5 highlights that the combined models achieved a flatter learning curve, indicating an improvement in mitigating overfitting of the dataset. Notably, the most significant improvement was observed in the models utilising categorical cross-entropy, both with and without class weights, with a respective decrease of 1.3457 (21.87%) and 0.8985 (16.29%) in validation loss.

As summarised in Figure 4, all models with class weights experiences a significant decrease in validation loss. This trend is particularly pronounced in the binary models, which experienced an average decrease of 0.05887 (23.10%) in validation loss. Similarly, models utilising the RMSprop optimizer without class weights display an average decrease of 0.05842 (23.04%) in validation loss. Moreover, models employing the RMSprop optimizer in conjunction with class weights demonstrate an average decrease of 0.0239 (12.45%) in validation loss.

Results - Deployment

All the plots above can be accessed through the **CCR Shiny App**. Thiss computer vision tool was development for the healthcare practitioners and academic researches in this field. It consists of six tabs: the introduction tab, three separate tabs for a brief description of each noise and interactive bar chart results, one tab for the interactive visualisation of learning curves for every model used for this project, and one tab that demonstrates an application of such tool. All interactive visualisation tools were created using plotly package in R (reference to plotly).

Introduction

The introduction tab provides an overview of on the usage of the application, designed as a navigation guide for first time users.

Choise of Noise

Within the noise description and results tabs, each augmentation was accompanied by examples showcasing its effect on a cell image, allowing users to compare it with the original image. This visual representation served to illustrate the impact of the augmentation for users who may be unfamiliar with the technique. Subsequently, a justification for the selection of each augmentation was provided, explaining the reasoning behind its inclusion. Additionally, an interactive bar chart was utilised to compare the validation loss of the models with the implemented augmentation. By hovering over each bar in the interactive plot, users can access the specific validation loss value.

Interactive Learning Curve Visualisation

This particular tab offers an interactive visualisation of the learning curves for all the models employed in this project. By default, the plot displays the validation loss over 100 epochs for models trained without any noise augmentation. However, users have the freedom to select and compare different models of their choice. This interactive feature enables users to gain insights into the impact of various augmentations, model optimisers, and loss functions on the learning process of CNN models.

Demonstration

The demonstration tab provides users with the opportunity to upload a cell image and experiment with various data augmentations, such as different levels of Gaussian noise, rotation, and resolution changes. By applying these augmentations simultaneously, users can observe the resulting predicted cluster in the output tab. This interactive feature enables users to gain a better understanding of how different data augmentations affect the model's predictions, showcasing one of the many practical applications of this tool. All predictions are generated using the Combined CNN model, utilising categorical cross-entropy as the loss function.

Discussion

How robust and generalizable is your finding or your product?

Discussion of potential shortcomings or issues associated with the development process or the product with reference to both disciplines.

Identification of future work or improvement in both disciplines.

Furthermore, as can been seen in the confusion matricies in Figure (ref) ... (Talk about Combined model not overfitting)

Why the use of this app can help our stakeholders

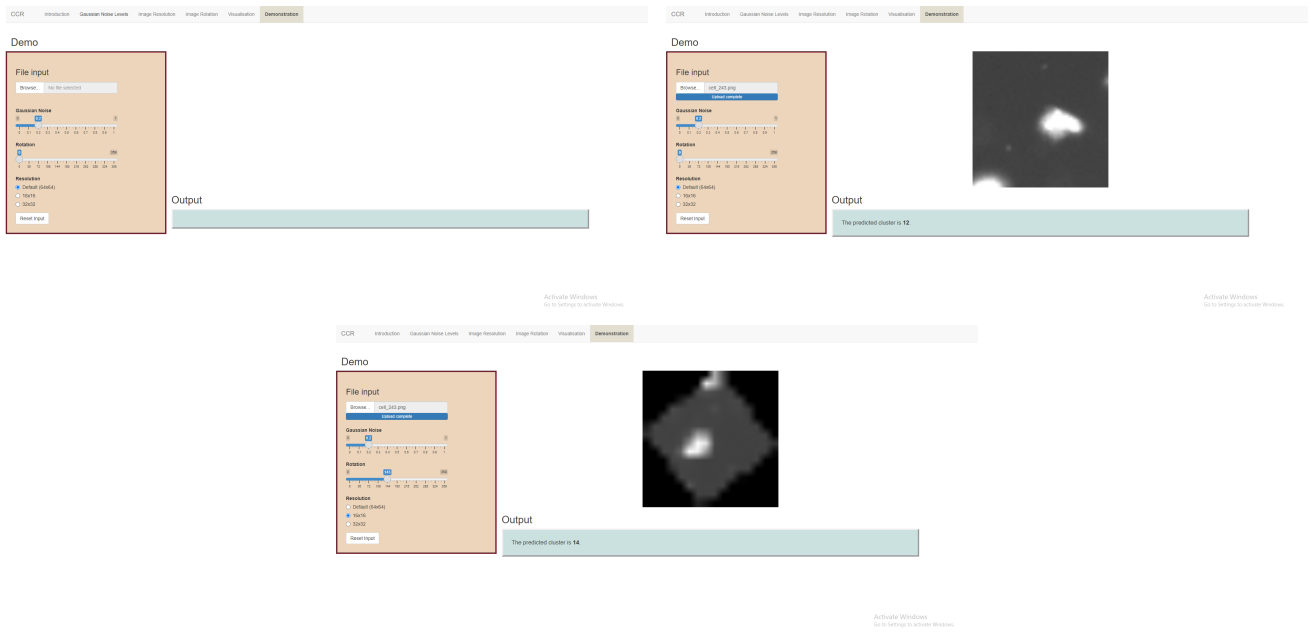

Figure 3: Screenshots of the Demonstration Tab in the CCR Shiny App. Default (Top Left), Image with 0.2 mean Gaussian Noise (Top Right), and Combination of Different Augmentations (Bottom).

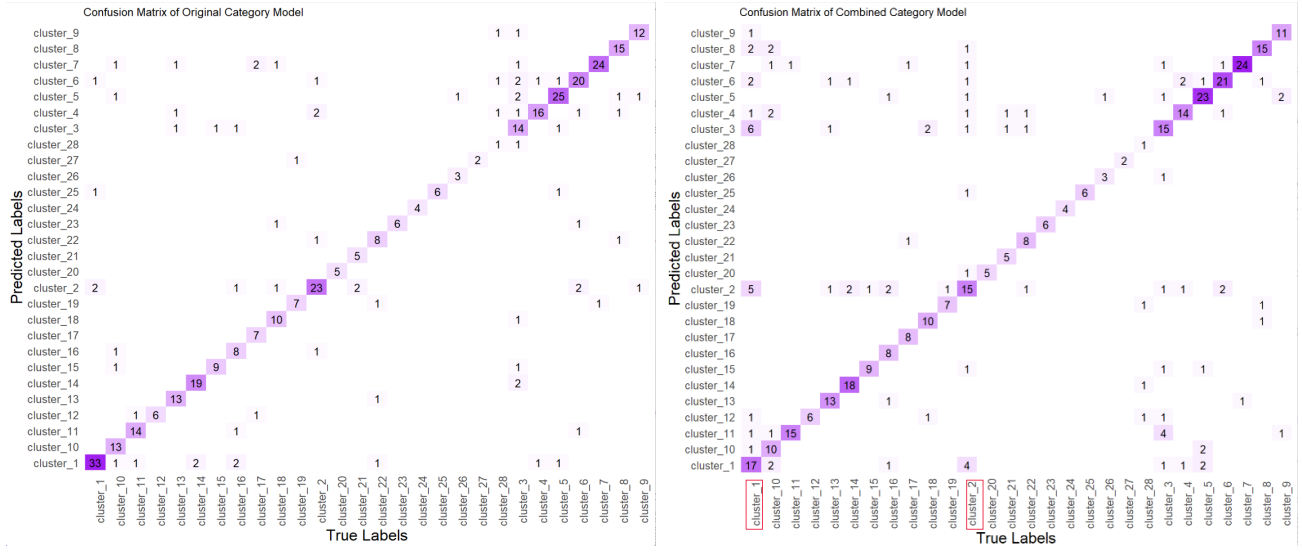

Figure 4: Confusion Matrix on Validation Set for Original CNN model (Left) and Combined CNN model (Right). Both models used Categorical Cross-Entropy as the loss function.

Conclusion

Conclusion adequately summarises the project and identification of future work.

Student Contributions

References

Appendix

Main code & technical details of your approach