COMP5329 Assignment 2

Group Members	SID	Unikey
Cheng Lu	480286592	chlu8208
Haeri Min	480401391	hmin5434
Albert Huang	530215563	qhua0468

1 Introduction

1.1 Motivation of study

Abundance in data has led to implementation of deep learning techniques using various forms of input data. Images are one of the rising area of interest in applying deep learning techniques as there are many real-world applications that rely on image recognition such as face recognition on mobile phones, autonomous driving, and diagnosis based on scanned images in healthcare. Images can be used to segment, detect, track, and classify objects in a single image. To be able to correctly use image recognition tools in the real-world, it is imperative to improve and maximise model performance (i.e. accuracy) of image classifiers. Failure to detect images or incorrect labelling information may lead to high false negatives and/or false positives that incorrectly diagnose patients' disease in medicine or failure to detect other cars or pedestrians when driving which threaten safety of people.

1.2 Aim of study

Past literature covers how deep learning techniques could be used to classify single-label images (i.e. images with one object to identify such as dog or cat) but there is a lack of evidence in using deep learning to classify multi-label images, that is, images that contain multiple objects, not limited to one object in a single image data. Realistically, it is more common to find images that contain multiple objects, hence it is important to train classifiers so that they accurately classify multiple objects in one image. Therefore, the aim of this study is to train a multi-label image classifier that improves prediction accuracy of multi-label images.

1.3 Introduction & Motivation of Methods

In this project, we will integrate Residual Networks (ResNet) in image data for multi-label image classification to construct algorithm that enhances model performance. In order to improve classification performance, we will also utilise available text captions given in the input image dataset. Classification involving image datasets will be done using ResNet50 whereas captions (i.e. text data) will be processed using Long Short-Term Memory which is a type of recurrent neural network architecture that is ideal for long-term dependencies and sequence prediction tasks.

The remainder of this project is organised as follows: section 2 summarises related works, section 3 explains methods that were used to generate classification model, section 4 presents experimental results, section 5 discusses pros and cons of the proposed method, and summarises findings from the study. Finally, section 6 illustrates hardware specifications, and thoroughly explains steps to run our code.

2 Related Work

Multi-label image classification can be achieved by implementing deep learning techniques such as Convolutional Neural Networks (CNN) or Support Vector Machines (SVM). Many literatures introduce methodologies that lead to improvements in classification performance and some papers critically point out limitations of image classifications and propose a new approach that can detect 3-dimensional objects such as human actions. In this literature review, we aim to provide an overview of past research that looks at using deep learning techniques to predict labels for a particular image training sample.

CNNs, introduced by LeCun in 1989, are widely used across many studies to classify, detect, and segment images. Some of the winning CNN architectures in ImageNet Large Scale Visual Recognition Challenge (ILSVRC) include AlexNet (2012), ZF Net (2013), GoogleNet (2014), and Resnet (2015) [1]. In particular, AlexNet consists of 5 convolutional neural networks and 3 fully connected layers. Prawira et al. (2021) showed that AlexNet outperformed VGG16 architecture when detecting more than one disease in the human eye. The authors note that there may exist several diseases in one fundus image hence applied both AlexNet and VGG16 with Adam optimiser to compare model performance. As part of the data pre-processing step, they reduced the size of pixel images to 227 x 227 and split training and test data with a ratio of 70:30 [2]. AlexNet model was run using 200 epochs, and 100 epochs were used for the VGG16 model (VGG16 is a development of AlexNet but replaces large kernel size to 3 x 3 instead). After training both classifiers, the accuracy was 95% and 89% for AlexNet and VGG16, respectively. Despite the relatively high performance of AlexNet, its complex structure, and intensive computational cost when using a large amount of training image data require appropriate methods to be implemented. Sun et al. (2016) note that using a Rectified Linear Unit (ReLU) in the AlexNet structure enhances training speed, Local Response Normalisation (LRN) improves the generalisation of the model, and employing Dropout improves robustness in the fully-connected layers [3]. Although AlexNet has been known to produce high-performance results, appropriate methods need to be applied when constructing the architecture to ensure that the model is not computationally costly and has larger generalisability in other datasets.

Traditionally, many studies explore the classification of multi-label images using Convolutional nets. However, to further improve efficiency and accuracy, Tan & Le (2019) proposed a new neural architecture, called EfficientNets, that achieved state-of-the-art 84.3% top-1 accuracy on ImageNet, while being 8.4 times smaller and 6.1 times faster compared to the existing ConvNet [4]. In contrast to existing CNN architecture, EfficientNet adopts a sigmoid activation function instead of softmax as probabilities of output images are dependent in the softmax activation function (that is, if the probability of one class increases, then the probability of the other class decreases). However, in multi-label image classification tasks, we want probabilities to be independent of each other as there is more than one label in a single image and this is achieved by the sigmoid activation function as it turns a multi-label problem into an n-binary classification [5]. Similar to Prawira et al. (2021), Wang et al. (2020) also used fundus images to screen for retinal diseases. The authors highlight that the best training method is when data is resized to 448 x 448, use the parameters of the pre-trained EfficientNet into the transferred neural network, and integrate two independently trained weak classifiers [6]. Moreover, the authors acknowledge that when training using small datasets, fine-tuning the pre-trained CNN such as EfficientNet can bring good results [6].

Image classification has been widely adopted in medicine as a tool for early diagnosis and detection of cancer. Breast cancer is the second leading cause of cancer deaths among women in the United States. Shen et al. (2019) applied both Resnet50 and VGG16 to assess classification performance in detecting lesions in available mammography databases. The study is motivated by the fact that region of interest (ROI) annotations are limited in large mammography databases making the region-based convolutional neural network (R-CNN) infeasible. As an alternative approach and overcoming image classification where annotations are not available, the authors pre-trained a model to classify local image patches using a fully annotated dataset with ROI and then used the weight parameters of the classifier to fine-tune datasets without ROI annotations [6]. They implemented this methodology on two datasets: S1 and S10 where S1 is based on images on the region of interest with random background and S10 consists of images around the region of interest (overlapping ratio is 0.9) with random background. This paper highlights that the best classification performance was achieved using the S10 dataset and Resnet50 (AUC = 0.87 [0.84, 0.90]) as it allows a patch classifier (i.e. a sliding grid that generates a probability of outputs) to extract more features that can be important for whole image classification [6].

There are many analytic studies for Long Short-Term Memory (LSTM), for example, the one conducted by Yu et al. (2019) [7]. They aimed to thoroughly examine and understand the capabilities and applications of LSTM[7]. Moreover, Staudemeyer, C., & Rothstein, E. conducted a study that focuses on the evolution and impressive functionality of LSTM-RNNs through a detailed analysis of early groundbreaking publications[8].

Although many studies implement deep learning techniques to classify and detect images, not a lot incorporate text captions when building classification models. In this project, we aim to utilise both text captions and images to construct an algorithm that achieves high performance rate compared to the existing literature. This will be done by using LSTM for prediction of captions and ResNet50 for image classification.

3 Techniques

The aim of the study is to construct a multi-label image classifier that produces the highest performance. Deep learning techniques as well as data pre-processing techniques that were applied to boost the performance of the classifier are outlined in this section.

3.1 Data Pre-Processing Techniques

3.1.1 Balancing the data

Initially, to solve dataset imbalance, weighting methods like **Inverse Class Frequency Weighting** and **Dynamic Weighting** are considered as possible solutions. However, some significant drawbacks of these methods cannot be ignored. For example, inverse class frequency weighting would make the weights remain the same throughout the process, lead to the lack of adaptation to changes in performance or other factors. For dynamic weighting, it would increase the computational complexity, consume more resource, and lengthen running time of the code. What's more, the implementation of F1 Score as evaluation metric guarantee the possible of high score when the model is performing well on both positive and negative instances even with data imbalance. Therefore, with all considered, data balancing is unnecessary for this task.

3.1.2 Image augmentation in Keras

Image augmentation is a deep learning technique that is commonly applied in image processing. This technique creates variations (i.e. rotated, flipped, shifted etc.) of raw images. This not only increases the size of training image dataset, but also improves generalisability of deep learning models on unseen data.

In this study, we applied ImageDataGenerator from Keras in Python to transform original images to improve generalisaibility and enhance model performance. Most notably, ImageDataGenerator provides real-time data augmentation, meaning that it augments the original images while the model is being trained. Additionally, it requires lower memory usage as it allows all the images to be loaded to be in batches instead of loading all at once.

Table 1 summarises parameters used in the ImageDataGenerator.

Layer	Parameter	Value
ImageDataGenerator	rescale	1./255
	rotation_range	20
	width_shift_range	0.2
	height_shift_range	0.2
	shear_range	0.2
	zoom_range	0.2
	horizontal_flip	True
	fill_mode	'nearest'

Table 1: ImageDataGenerator Parameters

3.1.3 Natural Language Toolkit (NLTK)

Natural Language Toolkit (NLTK) is a leading platform that specialises in natural language processing [9]. It has a set of libraries that help with "classification, tokenization, stemming, tagging, parsing, and semantic reasoning" [9].

In the early stage of developing our classification model, the performance did not meet satisfactory level. In order to resolve poor performance issue, we decided to include captions in the input of our classifier. NLTK was selected as it is available in Python, simple to implement, relatively new, and suitable for the task of extracting usable information from captions. It is expected to preparing textual data and enhance the quality and suitability of it for the LSTM model. More accurate and relevance the extracted data, the better performance of the model will be.

For more detailed information, NLTK in TensorFlow.keras package was used to perform text pre-processing tasks as shown below:

1. **Tokenization:** The caption is a long raw text, and this process break the text into smaller tokens like words or punctuation. It is an essential first step and preparation for later process in transformation caption into usable input data [10].
2. **Stop Word Removal:** Stop words refer to those words who don't have a specific meaning in context, like "and", "the", "are", "is". These words will be removed to reduce possible noise, easing computational effort and improve overall data quality.

3. **Stemming:** This process change words to their root form. For example, ”played” to ”play”, ”playing” to ”play”. Ideally, this approach will contribute greatly to the generalization of the words, discovering and building connection or relationship will be easier.

3.1.4 MultiLabelBinarizer

MultiLabelBinarizer is a class from scikit-learn package. It is a transformer that converts a list of sets or tuples into the format of a binary matrix [11]. The matrix is constructed by samples x classes, in which every row represents a sample, and each column represents a label. If a label is occurred in one sample, then the corresponding cell in the matrix will be set to 1, otherwise 0. This output matrix is in supported multi-label format [11].

There are step we adopted to complete this task. First, we prepare the labels by converting the training data into a list of sets. Each set contains the all the labels shown for an image. Then, this list of sets will be fitted accordingly for the later process of MultiLabelBinarizer. In this step, all unique labels in the dataset will be identified. At last, labels will be transformed for each batch of data.

Using MultiLabelBinarizer simplifies the process of encoding or transforming labels. It provides a consistent solution on this matter while guarantee the quality of result. What’s more, the binary matrix output is suitable for our task here as each label is treated independently, allowing the model to gain information of the status of each label for every sample without causing interference or missed data.

3.2 ResNet50

Residual Neural Network (ResNet) was first introduced by He et al. (2015) [12] and it is a type of Convolutional Neural Network architecture. Unlike traditional deep learning models such as AlexNet, and VGG16 where networks directly learn output functions, ResNet learns output layers by approximating a residual function which is a difference between the input and output of the residual block, $F(x) = H(x) - x$ (as shown in Figure 3).

In ResNet, there are two pathways for the gradients to pass through to the input layer: Identity Mapping and Residual Mapping. Residual Mapping consists of two weight layers, W_1 and W_2 where the weights are updated and new gradient values are calculated (see Equation 2). On the other hand, identity mapping does not have any associated weight layers, which indicates that there won’t be any changes to the value of computed gradients. This improves the performance of the network as gradients transit back to the initial layer with unchanged gradient values.

ResNet takes in input image size of 224 x 224 x 3 and performs initial convolution and max-pooling using 7 x 7 and 3 x 3 kernel size respectively [13]. In particular, for deeper ResNet networks like ResNet50 or ResNet152, bottleneck design is used where 3 layers (1x1, 3x3, 1x1 convolutions) are stacked over each other. The 1x1 convolution layers reduce and then restore dimensions, whereas the 3x3 layer is left as a bottleneck with a smaller input/output dimensions. [13]

$$F(x, W_i) = w_w \sigma(W_1 x) \tag{1}$$

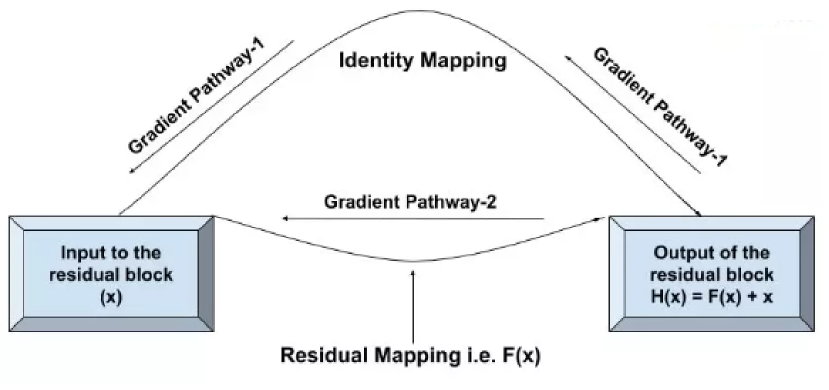

Identity Mapping
Gradient Pathway-1
Gradient Pathway-1
Gradient Pathway-2
Input to the residual block (x)
Output of the residual block H(x) = F(x) + x
Residual Mapping i.e. F(x)

Figure 1: Gradient Pathways in ResNet [13]

3.3 Long Short-Term Memory (LSTM)

LSTM is a type of recurrent neural network that was introduced to eliminate short-term memory of traditional neural networks. Hence, LSTM has long dependencies and has long-term memory which enables the model to improve performance by memorising and finding pattern in the data. Also, unlike other non-neural network classification techniques, where each word is classified to a category, LSTM learns from multiple words and classify into which group it belongs to. There are three main gates in LSTM: Forget, Input and Output gates. The forget gate decides which information is to be kept and discarded, the input gate helps identify important information and store certain data in the memory [14]. By combining all the information, the cell state is updated to new values. Finally, the output gate decides what information the hidden state should carry. The architecture of LSTM is shown in Figure 2.

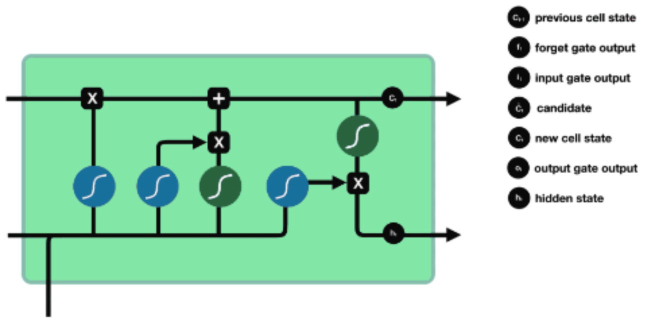


Figure 2: LSTM [14]

3.4 Advantages

Solving more complicated tasks involve using complex models that have larger computational cost compared to simple deep learning architectures. However, literature shows that increase in model complexity is not necessarily associated with better performance but actually increases training error due to vanishing gradient problem. This is mainly due to the backpropagation in neural networks which adjusts weights of neurons by travelling through the network from the back to the initial layer. As this process approaches the initial layers, the gradient can become smaller hence there may be no or very slight adjustment of neuron weights which increases training error. ResNet50 mitigates gradient descent problem by introducing skip connections in the design. As shown in Figure 4, the idea is to add the activation layer with the output of a later layer and allow the network to skip certain layers if they do not contribute anything to a better result. Not only ResNet eliminates gradient descent problem, but it has relatively lower computational cost compared to other deep neural networks such as DenseNet. Lastly, it uses Batch Normalisation which adjusts the input layer hence increasing the performance of the network and mitigating the problem of covariate shift. [13]

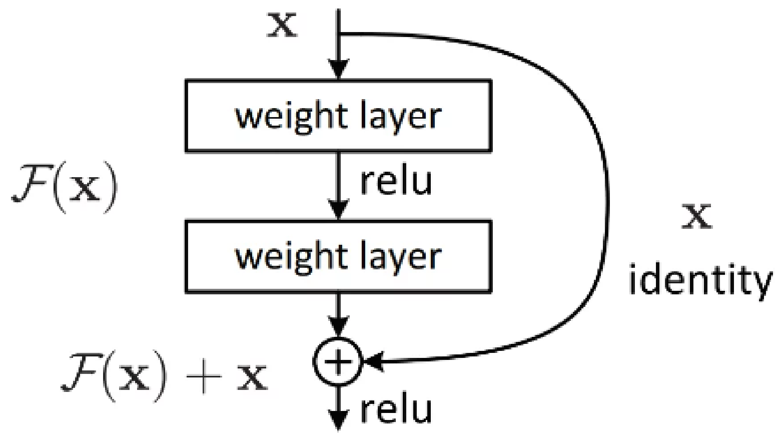

x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 3: ResNet Architecture

Similarily, LSTM alleviates gradient descent problem by using a unique additive gradient structure that includes direct access to the forget gate's activation. Additionally, LSTM is effective in capturing long-term dependencies hence it is useful in classification of texts, specifically, image captions given as part of input data in this project. Incorporating text data using LSTM in image classification is highly likely to improve accuracy of model as it provides extra level of information during training. Since LSTM can memorise long term dependencies and given that we have sentence type captions in the input data, it is reasonable to implement this technique in our project to improve classification performance.

4 Experiments and Results

4.1 Model Architecture

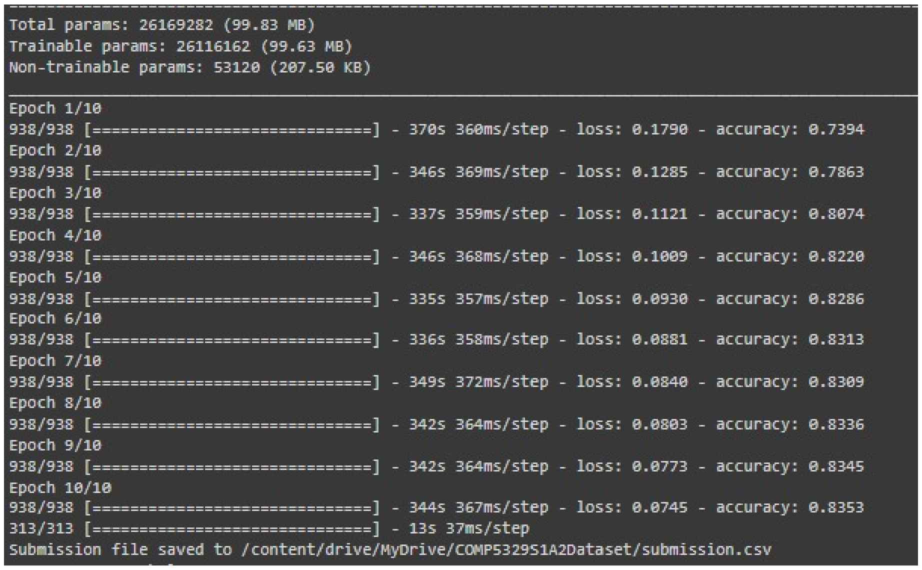
In general, our multimodal model is based on pre-trained Resnet50 and processes both image and captions. Captions are handled by Bidrectional LSTM. Specifically, the architecture is based on the pre-trained result on the ImageNet dataset. As the project requires the size of the model under 100MB, we have made several changes on neurons sizes on several layers. For instance, the neurons for dense layers and LSTM layer are reduced to 128. We managed the final model size to be 99.83MB.

4.2 Model Performance

F1 Score is the harmonic mean of precision and recall [15]. It provides a balanced assessment of the model's performance on positive and negative instances. In certain conditions, F1 score could be a good choice of evaluation metric for imbalanced datasets. And it is used in many areas, such as in medical diagnosis [16], fraud detection [17], or astronomical sourcing using machine learning [18]. The formula of F1 Score is shown as below:

$$\begin{aligned} F1\ Score &= \frac{2 \times precision \times recall}{precision + recall}, \\ precision &= \frac{true\ positives}{true\ positives + false\ positives}, \\ recall &= \frac{true\ positives}{true\ positives + false\ negatives} \end{aligned} \tag{2}$$

Though F1 score is chosen to be the final evaluation metric for assessing our model's performance, we chose 'Accuracy' as our metrics within the process of training as the training with F1-score requires more time. Also the main purpose for using loss and accuracy is to monitor the training process. According to our training history, the loss is decreasing gradually with the accuracy increasing gradually for each epoch. We believe it suggests our model is effective in capturing and learning the information in our dataset.



```
Total params: 26169282 (99.83 MB)
Trainable params: 26116162 (99.63 MB)
Non-trainable params: 53120 (207.50 KB)

Epoch 1/10
938/938 [==============================] - 370s 360ms/step - loss: 0.1790 - accuracy: 0.7394
Epoch 2/10
938/938 [==============================] - 346s 369ms/step - loss: 0.1285 - accuracy: 0.7863
Epoch 3/10
938/938 [==============================] - 337s 359ms/step - loss: 0.1121 - accuracy: 0.8074
Epoch 4/10
938/938 [==============================] - 346s 368ms/step - loss: 0.1009 - accuracy: 0.8220
Epoch 5/10
938/938 [==============================] - 335s 357ms/step - loss: 0.0930 - accuracy: 0.8286
Epoch 6/10
938/938 [==============================] - 336s 358ms/step - loss: 0.0881 - accuracy: 0.8313
Epoch 7/10
938/938 [==============================] - 349s 372ms/step - loss: 0.0840 - accuracy: 0.8309
Epoch 8/10
938/938 [==============================] - 342s 364ms/step - loss: 0.0803 - accuracy: 0.8336
Epoch 9/10
938/938 [==============================] - 342s 364ms/step - loss: 0.0773 - accuracy: 0.8345
Epoch 10/10
938/938 [==============================] - 344s 367ms/step - loss: 0.0745 - accuracy: 0.8353
313/313 [==============================] - 13s 37ms/step
Submission file saved to /content/drive/MyDrive/COMP5329S1A2Dataset/submission.csv
```

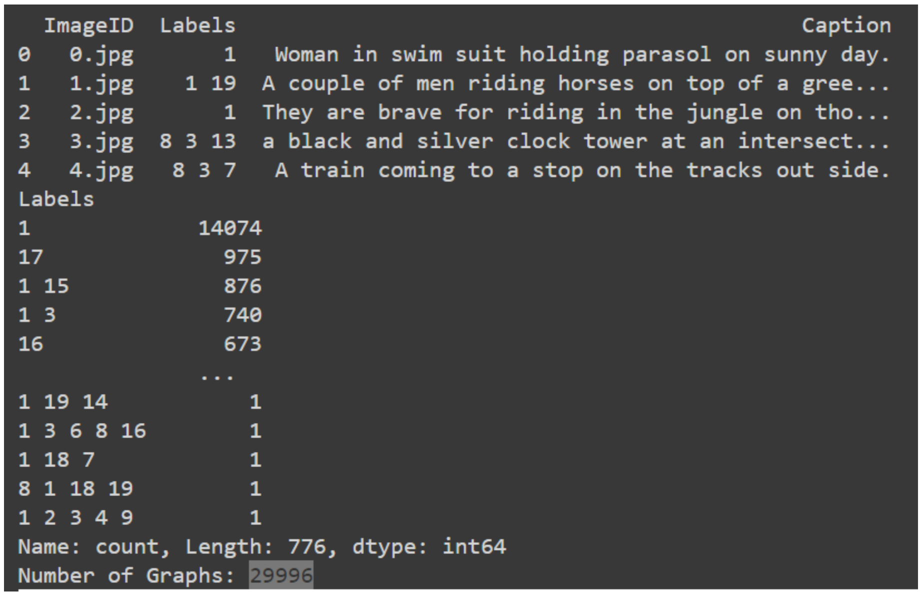
Figure 4: Accuracy Improvement

4.3 Extensive Analysis

4.3.1 Experimental Steps

Dataset is the source and the beginning point of this competition track. Therefore, it is vital to comprehensively understand the given dataset as it would provide guideline on the proceeding procedure.

Our dataset has three main parts: one training dataset (train.csv), one testing dataset (test.csv) and one folder (data) that contains all the images. There are 40,000 *.jpg* format images in the "data" folder, each is named by an unique number as their ID. The training dataset has 29996 image ID, each is correspond to one image in the folder. Every item (image) also has attributes of "Labels" and "Captions". "Labels" is integer from 1 to 19, and one image could have multiple labels. "Captions" is a series of text describing the context of the image. The rest of the images in "data" folder belongs to testing dataset which shares a similar structure as training dataset.



```
  ImageID  Labels                                           Caption
0   0.jpg       1   Woman in swim suit holding parasol on sunny day.
1   1.jpg    1 19  A couple of men riding horses on top of a gree...
2   2.jpg       1  They are brave for riding in the jungle on tho...
3   3.jpg  8 3 13  a black and silver clock tower at an intersect...
4   4.jpg   8 3 7   A train coming to a stop on the tracks out side.
Labels
1              14074
17               975
1 15             876
1 3              740
16               673
               ...
1 19 14            1
1 3 6 8 16         1
1 18 7             1
8 1 18 19          1
1 2 3 4 9          1
Name: count, Length: 776, dtype: int64
Number of Graphs: 29996
```

Figure 5: First five records in the *train.csv* data

As Figure 6 shows, there is imbalance between classes. Of 776 classes in total, 46.9% of images are classified as label 1, 22.8% are assigned to 12 different classes, and 30.3% of classes are assigned to labels that occur with less than 1%.

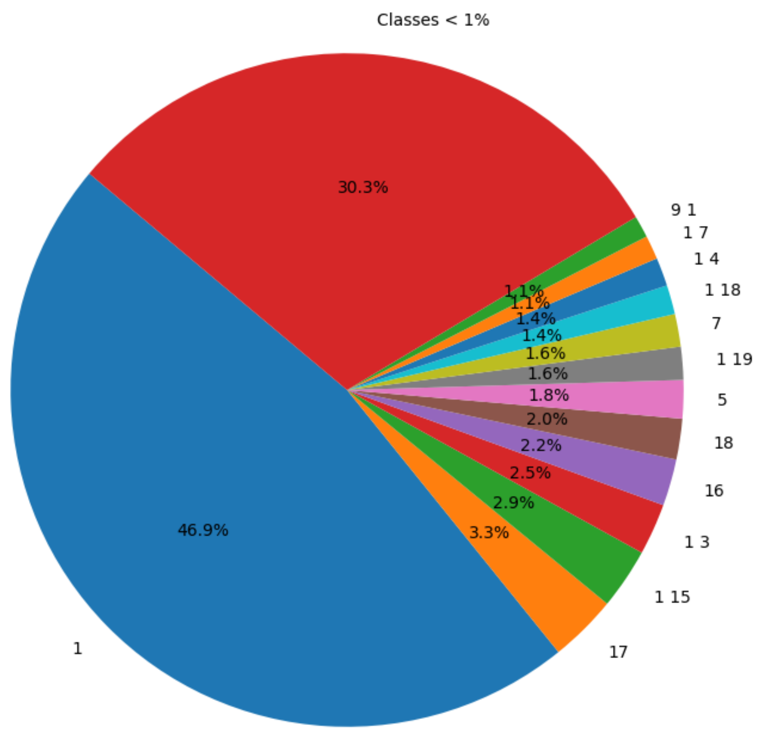


Figure 6: Distribution of image labels in the training data

Our rationale for this project is to choose a relative complicated CNN architecture as the project requirement only suggests the model size to be 100 MB and the feasible time limit for A2 is under 24 GPU hours on a single 4090. Also, we believe the captions should be handled in appropriate way since extra features might improve the overall performance for the model. Our mean f1-score for our first entry on Kaggle is 0.89359 with the model size 101.38 MB. As we used pre-trained Resnet-50 as our based model, we do not have many parameters to tune and we reduce the layer sizes to make it under the model size requirement. The team then put more time on using more pre-processing techniques like image augmentation to improve our mean f1-score. Team managed to achieve 0.91866 by tuning hyper-parameters and using more advanced pre-processing techniques, with the model size of 99.83 MB.

4.3.2 Ablation Study of Caption

In our model, we utilize captions for more related data input to the layer. It is expected to bring great performance improvement. To further understand the impact of including captions as one of the input source, we conduct ablation study on this matter by disabling related features like LSTM, NLTK and MultiLabelBinarizer. Without the input extracted from caption, the performance of the model was approximately dropped by 20%. It is certain that input data from captions contributes greatly to the performance of the model, and provides an insight on preparing training materials for multi-label classification tasks.

Hyperparameter	**Value**
num_words	10000
oov_token	OOV
output_dim	195
batch_size	32
epochs	10
learning_rate	1e-4
loss	'binary_crossentropy'
metrics	['accuracy']

Table 2: Hyperparameters of Best Model

5 Conclusion and Discussion

5.1 Outcome of Study

ResNet50 with LSTM which receive input from both images and caption has proved to be an ideal model architecture for multi-label image classification task with mid scale of data (40,000 images). Our model successfully achieved the aim of building a multi-label image classifier with desirable accuracy and mean F1-score. Through researching related works, exploring methods and techniques, and analyzing models, we gain deeper understanding of the application of deep learning in image classification and insights of principle of architecture, important factors like text data input for model performance.

5.2 Future Works

There is a relatively high performance gap between our model (F1-score = 0.91866) and the best model from the Kaggle Competition Leaderboard (F1-score = 0.96641). We believe the strategy of adopting NLP model like CLIP will better handle the features for captions. Also, it is possible that the simpler CNN architecture can have better performance on these specific dataset.

6 Appendix

6.1 Link to our code

`https://colab.research.google.com/drive/1jwu-AfZQ5cQqJUKVv4qkWDkMas54-Ya3?usp=sharing`

6.2 Link to our results (Predicted Labels)

`https://drive.google.com/file/d/1Rw4dbHmlDOe_cTuGr_8gdotM1M8swNnY/view?usp=sharing`

6.3 Specifications

6.3.1 How to run our code

1. Download the zipped file ”A2.zip” from the link: https://drive.google.com/file/d/1m6WlJe4kyIyTwirwZrw1-fEljxfVvTN0/view?usp=sharing
2. Upload the ”A2.zip” file to your own drive under ’My Drive’.

6.3.2 Software

1. Coding Language: Python
2. Web-interface: Google Colaboratory

6.3.3 Hardware

1. Operating System: Windows 11
2. RAM: System RAM (83.5 GB), GPU RAM (40 GB)
3. GPU: A100 from Google Colab
4. Runtime: 3600.392 seconds

References

[1] Chang Xu. *Neural Network Architectures*. COMP5329 Deep Learning, 2024.

[2] Reyhansyah Prawira, Alhadi Bustamam, and Prasnurzaki Anki. Multi label classification of retinal disease on fundus images using alexnet and vgg16 architectures. In *2021 4th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, pages 464–468, 2021.

[3] Jing Sun, Xibiao Cai, Fuming Sun, and Jianguo Zhang. Scene image classification method based on alex-net model. In *2016 3rd International Conference on Informative and Cybernetics for Computational Social Systems (ICCSS)*, pages 363–367, 2016.

[4] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 09–15 Jun 2019.

[5] Jing Wang, Liu Yang, Zhanqiang Huo, Weifeng He, and Junwei Luo. Multi-label classification of fundus images with efficientnet. *IEEE Access*, 8:212499–212508, 2020.

[6] Margolies L.R. Rothstein J.H. et al. Shen, L. Deep learning to improve breast cancer detection on screening mammography. *Sci Rep*, 9, 2019.

[7] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation*, 31(7):1235–1270, 2019.

[8] Ralf C Staudemeyer and Eric Rothstein Morris. Understanding lstm–a tutorial into long short-term memory recurrent neural networks. *arXiv preprint arXiv:1909.09586*, 2019.

[9] NLTK Project. Documentation natural language toolkit, 2023. Available online: `https://www.nltk.org/`, Accessed: May 17, 2024.

[10] Robert Friedman. Tokenization in the theory of knowledge. *Encyclopedia*, 3(1):380–386, 2023.

[11] scikit-learn developers. sklearn.preprocessing.multilabelbinarizer, 2024. with BSD liscense, Available online: `https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MultiLabelBinarizer.html`, Accessed: May 17, 2024.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[13] A. Sachan. Detailed guide to understand and implement resnets. *CV-Tricks.com: Learn Machine Learning, AI Computer Vision*.

[14] S. Shekhar. What is lstm for text classification? *Analytics Vidhya*, 2024.

[15] Yutaka Sasaki et al. The truth of the f-measure. *Teach tutor mater*, 1(5):1–5, 2007.

[16] Zachary DeVries, Eric Locke, Mohamad Hoda, Dita Moravek, Kim Phan, Alexandra Stratton, Stephen Kingwell, Eugene K Wai, and Philippe Phan. Using a national surgical database to predict complications following posterior lumbar surgery and comparing the area under the curve and f1-score for the assessment of prognostic capability. *The spine journal*, 21(7):1135–1142, 2021.

[17] Dang Ngoc Hung, Hoang Thi Viet Ha, and Dang Thai Binh. Application of f-score in predicting fraud, errors: Experimental research in vietnam. *International Journal of Accounting and Financial Reporting*, 7(2):303–322, 2017.

[18] A Humphrey, W Kuberski, J Bialek, N Perrakis, W Cools, N Nuyttens, H Elakhrass, and PAC Cunha. Machine-learning classification of astronomical sources: estimating f1-score in the absence of ground truth. *Monthly Notices of the Royal Astronomical Society: Letters*, 517(1):L116–L120, 2022.