Multi-Label Image Classification

Course: COMP5329 Deep Learning Assignment2
Group members: Young Jun Cho (490455715), Ruyi Lu (440509448)

1 Introduction

1.1 Aim of the study

The aim of the study is to utilise various deep learning techniques on a provided dataset containing images and captions, with the objective of addressing a multi-label classification problem. A series of experiments were conducted on the dataset to analyse the architecture of the model affects the performance of the model and explore potential ways to enhance the accuracy of the model.

1.2 Importance of the study

The importance of the study is to explore how the combination of natural language processing (NLP) and convolutional neural networks (CNNs) for both textual and visual information can improve the accuracy of the classification models. NLP excels at processing textual information while CNN excels at extracting features from image data. By leveraging the advantages of both applications, this integration could lead to better performance.
In real life, data originates from different sources and may contain noise, so identifying relevant and important information from the data has the potential to improve the model's accuracy by excluding non-key features.

1.3 General introduction of used method and motivation for a solution

The deep learning model used in this assignment is a hybrid model incorporating CNNs, LSTMs, and Transformers. This is expected to leverage the strengths of each component to handle the complexities of multi-label image classification. By merging diverse architectures, our framework is well-positioned to tackle intricate label dependencies, making it a robust solution for real-world MLC challenges. The motivation for applying a hybrid model is due to the different data modalities and the motivation to combine the advantages of different models to achieve better performance in the multi-label classification task.

2 Literature Review (Related works)

2.1 Literature Review on Deep Learning for Multi-Label Image Classification

1. **Introduction of multi-label classification problem**
   Multi-label classification is a challenging domain in computer vision. Our task is a one-to-many scenario and we aim to predict one or more than one label from a single input, where an image with captions belongs to different labels. Given the difficulties inherent in multi-label classification, such as label dependencies and large scale of the image data, we chose deep learning techniques as they are more effective in addressing those challenges faced in multi-label learning in comparison to the conventional methods (Zhang et al., 2020).

2. **Multi-Modal Challenges in Deep Learning for Image Classification**
   Nowadays, data is in different representations such as voice data, text data, image data, etc. The provided dataset contains images with captions which contain two different modalities. This increases the difficulties of the models as they require learning complex data with multiple modalities.
   One of the challenges is to join two or more modalities on a classification problem as each data type has its own unique structure and may not always be aligned with each other (Rusli, 2021). Although multi-modal challenges exist, this gives us opportunities to utilise deep learning techniques effectively to enhance the performance of the model in image classification tasks.

3. **Overview of different deep learning techniques**

   - **Convolutional neural networks (CNN)**
     CNN has become the state of the art for various computer vision tasks, especially image classification problems. The architecture of the networks consists of convolutional layers, pooling layers, and fully connected layers, allowing them to extract complex features from the raw image.
     Innovations such as ResNet and Inception models are advanced CNNs which mitigates the vanishing gradient problem. This improvement solves the issues with training of deeper neutral networks, enabling more complex and abstract representations of image content (Alzubaidi et al., 2021).
     Additionally, CNN models designed for single-label image classification can be extended to address multi-label problems. This extension aims to improve the structure of the CNN models by optimising the binary cross-entropy (BCE)(Tarekegn et al., 2024).
   - **Long short-term memory (LSTM)**
     Long short-term memory (LSTM) is good at learning sequences of data and reducing the problems of vanishing or exploding gradients. In our case, it can be applied to process captions associated with images. In terms of multi-label classification, each image is associated with one or more than one label. By incorporating the caption, it allows the model to learn from the additional textual information, which provides more context and additional semantic information, enabling it to learn different features and predict more accurate information for each label.
   - **Transformers**
     Transformers are initially designed for natural language processing tasks. Later on, they have been applied to vision-related tasks. For instance, the Vision Transformer (ViT) applies a self-attention mechanism to patches of images, treating them similarly to tokens in a sentence, which allows it to capture intricate dependencies between image regions and labels (Tarekegn et al., 2024). In terms of multi-modalities, transformers use cross-modal interactions to integrate data in different forms.

4. **Handling Label Dependencies**
   Label dependencies are common in multi-label classification tasks. Graph neural networks (GNNs) have been utilised to address the failure of capturing the dependencies of labels. GNNs enhance the accuracy of the prediction by capturing the semantic context among label relationships (Tarekegn et al., 2024).

5. **Training Strategies**
   Due to the imbalanced label distributions, training models for multi-label classification is challenging. To overcome this class imbalance problem, several techniques have been employed such as label smoothing and focal loss. Furthermore, novel approaches including partial labels or soft labels which are common in real-world datasets enable models to learn from incomplete or uncertain label data (Tarekegn et al., 2024).

6. **Datasets and Evaluation**
   For multi-label image classification, datasets must be carefully curated to reflect the diversity and complexity of real-world scenarios. This includes ensuring a balanced representation of labels and considering the specific characteristics of images that may affect classification, such as variations in lighting, pose, and background (Laith Alzubaidi et al.,

2023).
In our study, the provided dataset for multi-label classification tasks comprises two csv files: a training dataset and a testing data. Both files contain images of different sizes, each image has captions that describe the content of the image. Also, the training dataset includes labels for each image, which is a supervised learning process for training multi-label classification models.
Evaluation metrics for multi-label classification include precision, recall, F1-score, and the mean average precision (mAP), which consider the prediction accuracy across multiple labels (Papers with Code - Multi-Label Image Classification, n.d.).

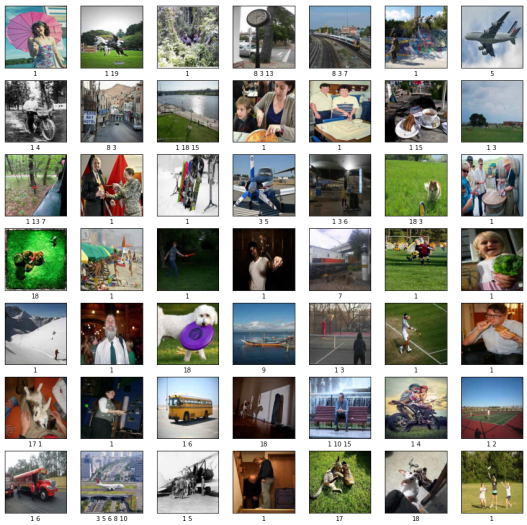

Figure 1: *Image with Labels*

7. **Applications and Future Directions**
   Multi-label classification (MLC) has a wide range of applications, from medical imaging, where multiple pathologies may need to be identified in a single image, to social media, where content needs to be tagged with multiple relevant tags. The future of MLC likely involves integrating multimodal data, as the combination of text, image, and possibly audio data could provide complementary information that enhances model performance.

2.2 Example of a Review of Specific Papers

A prominent research paper that utilises a hybrid model is CNN-RNN: A Unified Framework for Multi-label Image Classification. It combines CNNs and RNNs (specifically LSTM) to create a joint image-label embedding. This approach captures label dependencies, improving classification accuracy on benchmark datasets. The model demonstrates how recurrent layers can complement convolutional ones, making it an effective framework for multi-label classification (Wang et al., 2016).

2.3 Integration with Existing Research and Future Research Directions

In developing our hybrid model using CNN, LSTM, and Transformers, we build on existing research highlighting the strengths of attention mechanisms in modelling complex data relationships. Transformers, known for scalability and parallelism, will enhance CNN and LSTM capabilities. LSTMs excel at capturing sequential patterns in textual captions accompanying images, while Transformers, with their self-attention mechanisms, model long-range dependencies. Together, these architectures capture spatial, sequential, and context-dependent patterns, enabling more accurate multi-label classification through efficient integration of both visual and textual data.

3 Techniques

3.1 Data Loading and Preprocessing

Principle
The principle involves parsing CSV files to load image file paths and label data. During the preprocessing phase, text manipulations resolve any format issues, and transformations of images and labels adjust the data to be suitable for model training.

Implementation
Images are loaded from directories, and depending on the requirements, image transformations (image_transform) are applied. Label transformations (label_transform) convert categorical labels into a one-hot encoding format.

Justification
Data loading and preprocessing are essential steps in any machine learning pipeline. Efficient and accurate data loading mechanisms reduce training time and ensure data consistency and accuracy. One-hot encoding of labels allows each category to be treated independently in multi-label classification tasks.

Advantage/Novelty
This method adeptly handles complex text and errors that may occur during data loading through the implementation of one-hot encoding for labels, minimising potential errors. It facilitates independent processing of each category in multi-label classification, enhancing the model's ability to accurately categorise multiple labels simultaneously.

3.2 Image Transformations

Principle
The principle involves applying various image transformation techniques for data augmentation. This ensures that the model can robustly handle a variety of visual alterations.

Implementation
To enhance the image data for training, the Compose function is used to apply several transformations including 'ToTensor', 'Normalize', 'Resize', 'RandomHorizontalFlip', 'RandomVerticalFlip', and 'RandomCrop'. These transformations are intended to improve the model's generalisation capabilities.

Justification
Image datasets are often limited, so data augmentation effectively increases the quantity of data and prevents overfitting. Transformations allow the model to reflect various real-world conditions, thus enhancing its ability to generalise.

Advantage/Novelty
These image transformation techniques are particularly beneficial in a multi-label environment, where they help the model better recognize different objects contained within each image. Additionally, the transformed images enable the model to operate robustly across a wider range of scenarios.

3.3 Dataset Splitting

Principle
Splitting the dataset into training and validation sets allows for the evaluation of the model's generalisation capabilities. This method ensures that the model does not merely overfit to the training data but performs well across a variety of real-world data.

Implementation
The 'random_split' function is used to divide the entire dataset into training and validation sets. This function splits the data randomly, ensuring that each data point is exclusively assigned to either the training set or the validation set. 80% of the total data is allocated for training, while the remaining

20% is used for validation to monitor the model's performance during training.

Justification
Splitting the dataset is a standard approach to prevent overfitting and objectively assess the performance of the model. Using a validation dataset to periodically evaluate the model during training allows for necessary adjustments in the model development process (Raschka, 2018).

Advantage/Novelty
Random splitting ensures that all data points are included in the training or validation dataset without bias, allowing fair representation of all data and enabling an accurate assessment of the model's generalisation ability. Using a manual seed for reproducible splitting maintains consistency in experiments, providing a foundation for comparing model performance under the same conditions across different settings or at different times.

3.4 Model Training and Evaluation

Principle
The model is provided with batches of images and labels, and errors are calculated using a loss function, followed by weight adjustments through backpropagation. This process ensures that the model optimises itself for the task at hand.

Implementation

- The 'training_cycle' function handles batch processing, loss calculation, and backpropagation, with optional evaluation of the validation dataset. Validation performance is measured using the F1 score.

- **Caption Encoding**
  - This process numerically encodes text data and combines it with image data for multi-labeled learning, allowing the model to process related textual information alongside images for more accurate predictions (Seo et al., 2022) . According to J (2021), recognizing captions with trigram methodology can better capture the rich information of text data and learn more diverse optional information in the text.
  - Implementation: Trigrams from the captions are calculated and converted into a frequency vector mapped to unique indices. This vector is used as an input during training, processed together with the image data (J, 2021) .

Justification
Backpropagation and loss minimization are fundamental components of deep learning model training, helping the model to optimise for the specified task. Periodic evaluation with a validation dataset provides opportunities to monitor and adjust the model's performance during the development process.

Advantage/Novelty
The ability to optionally process caption data allows the model to learn complex patterns, enhancing the model's accuracy. Experimenting with various combinations of hyperparameters helps identify the most effective model configuration for specific tasks, and this flexibility extends the practical application range of the model.

3.5 Model Architecture and Pipeline Design

Principle
This hybrid model utilises a combination of CNN (Convolutional Neural Network), LSTM (Long Short-Term Memory), and Transformer architectures. CNNs are used to extract features from images, LSTMs are apt for learning temporal characteristics of sequence data, and Transformers excel in understanding complex relationships and patterns. This structure is designed to effectively learn features from both images and associated caption data.

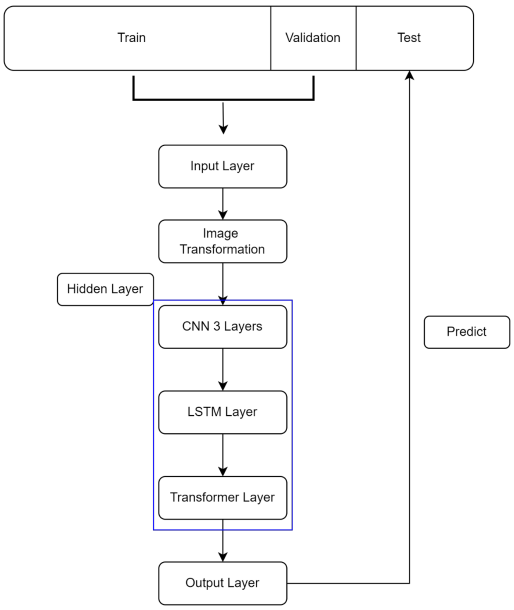

Figure 2: *Model Architecture*

Implementation

- **CNN Layer Configuration**
  - The CNN component begins with the first layer utilising 128 convolution filters (nn.Conv2d) with a 3x3 kernel size, stride of 1, and padding of 1. This layer generates 128 feature maps from the input images.
  - Subsequent layers reduce the feature maps to 64 using the same kernel size and padding settings through two more convolution layers. After each CNN layer, max pooling (F.max_pool2d) is applied with a 2x2 window to reduce the spatial dimensions while retaining critical spatial features.
- **LSTM Layer Configuration**
  - The transformed feature maps from the CNN layers are restructured and serve as inputs to the LSTM layer. The LSTM configuration includes one layer with 128 units, optimised for capturing temporal dynamics in the data. It processes the temporal sequence of features, retaining only the output from the last time step to be forwarded to the next processing stage.
- **Transformer Layer Configuration**
  - The LSTM outputs are then channelled into the Transformer encoder, which consists of two layers equipped with an attention-based mechanism (TransformerEncoderLayer with a model size of 128 and 4 attention heads). This setup enables the model to discern and learn complex dependencies and patterns across different segments of the sequence, focusing particularly on salient features.
- **Fully Connected Layer Configuration**
  - Post-transformer processing, the data is passed through a dense network layer where the output is transformed from 17704 units down to 1024 by an initial fully connected layer (nn.Linear), followed by a dropout layer (nn.Dropout) with a dropout rate of 0.7 to mitigate overfitting. The process concludes with a final dense layer that maps these features into the required number of categories, facilitating class prediction.
- **Activation Function**

* ReLU activation function has been employed allowing flexibility in the activation dynamics during training. Furthermore, the architecture supports the integration of image features with optional caption data, enhancing the model's ability to handle multi-labeled inputs and provide enriched context for more accurate predictions.

Justification
CNNs demonstrate strong performance in image processing by efficiently extracting key visual features. LSTMs have strengths in handling sequence data, making them useful for learning the visual flow in images. Transformers focus on important aspects of data through their attention mechanism, allowing for deeper understanding of relationships.

Advantage/Novelty
This hybrid model combines the strengths of each architecture to provide a robust learning capability specialised for multi-label image classification. Particularly, the ability to process image features alongside caption information enables the model to learn complex patterns of multi-labeled data, which significantly enhances the novelty and effectiveness of the model.

3.6 Training and Evaluation Pipeline

Principle
The training of the model is conducted by setting various hyperparameters such as epochs, batch size, learning rate, etc. Different experiments are performed according to each setting to find the optimal combination. During training, prediction errors are calculated using a loss function, and the model is optimised through backpropagation.

Implementation

- **Hyperparameter Settings**
  * Various dropout rates, batch sizes, learning rates, and various number of epochs are tested to optimise the model's performance. Each combination is evaluated and compared under specific experimental conditions.
- **Data Loader Setup**
  * A 'DataLoader' is used to supply training, validation, and testing data to the model. Data is processed in batches, and shuffling ensures diversity in the data throughout the training process.
- **Training Loop**
  * During each epoch, the model receives batches from the training data, performs predictions, and calculates loss using the 'nn.MultiLabelSoftMarginLoss' function. The Adam optimizer is used to quickly adjust the learning rate.
- **Performance Evaluation**
  * The performance of the validation dataset is periodically assessed during training, and metrics such as the F1 score are used to measure the model's predictive performance.
- **Prediction**
  * Predictions are carried out on the test dataset, with the results being converted into a NumPy array. In cases where label data is absent, the value is set to "3" to maintain consistency in the data. This is because in the labelling analysis, "3" is identified as the category "Others," which encompasses a variety of elements without a specific theme.

Justification
Experimenting with various training conditions is crucial for maximising the model's performance and assessing its utility in real-world environments. In the case of multi-label classification, accurate label assignment is essential, necessitating meticulous adjustments and evaluations.

Advantage/Novelty
The complex training pipeline evaluates the model's performance across a wide range of scenarios

through hyperparameter settings. This allows for the identification of the most effective model configurations for specific tasks, and such flexibility extends the model's application range in practical implementations.

4 Experiments and Results

4.1 Ablation studies

4.1.1 Step 1: CNN Model Only

Description
In this phase, we evaluated the performance of the CNN model in multi-label image classification. This was not the main focus of our research, so we concentrated on basic performance evaluation without conducting experiments with various combinations of hyperparameters.

Result
The test results showed that the CNN model achieved an approximate F1 score of 0.758. Adjusting various parameters did not significantly alter the performance.

Epochs	Dropout	Batch Size	Learning Rate	Learning Time	Train Loss	Validation Loss	Test F1-score
1	0.5	100	0.001	29 min.	0.66167	*	0.75800

Figure 3: *Step1 table*

4.1.2 Step 2: Hybrid Model

Description
This stage involved evaluating the performance of a hybrid model combining CNN, LSTM, and Transformer. The goal was to assess the model's ability to effectively learn the temporal and contextual characteristics of complex data.

Result Testing the hybrid model, which was built on the existing CNN model with added LSTM and Transformer, revealed significant performance improvements. Even a simple test showed enhanced performance, indicating that the hybrid model combining CNN, LSTM, and Transformer can deliver superior performance in multi-label image classification challenges. As part of the research, hyperparameter tuning was applied to enhance the model's performance and identify the optimal combination of hyperparameters. Consequently, various settings such as dropout rate, batch size, number of epochs, and learning rate were applied to conduct a more detailed comparison and analysis of performance.

Epochs	Dropout	Batch Size	Learning Rate	Learning Time	Train Loss	Validation Loss	Test F1-score
1	0.5	50	0.001	77 min.	0.07767	0.00298	0.91891
1	0.5	30	0.001	103 min.	0.08758	0.00069	0.91754
3	0.5	50	0.001	295 min.	0.08599	0.00057	0.91575
3	0.5	30	0.001	409 min.	0.09396	0.00094	0.91512
3	0.7	30	0.001	370 min.	0.10864	0.00046	0.91225
1	0.7	30	0.001	102 min.	0.14768	0.00092	0.90766
3	0.5	50	0.01	291 min.	0.06266	0.00108	0.90645
1	0.5	50	0.01	77 min.	0.12200	0.00298	0.90370
3	0.5	30	0.01	402 min.	0.09027	0.00258	0.90241
3	0.7	50	0.001	260 min.	0.11368	0.00236	0.89850
1	0.5	30	0.01	102 min.	0.15769	0.00184	0.89558
3	0.7	30	0.01	374 min.	0.12618	0.00182	0.89433
1	0.7	50	0.001	73 min.	0.11559	0.00212	0.89283
1	0.7	30	0.01	104 min.	0.15169	0.00137	0.88766
3	0.7	50	0.01	260 min.	0.13142	0.00494	0.88600
1	0.7	50	0.01	73 min.	0.12677	0.00151	0.87308

*default : Optimisation = Adam, Activation = ReLU, Threshold = 0.5, CNN layer = 3

Figure 4: *Step2 table*

4.2 Hyperparameter Analysis

Hyperparameter	Range	Final Model
Epochs	1,3	1
Dropout Rate	0.5,0.7	0.5
Batch Size	30,50	50
Learning Rate	0.01,0.001	0.001

*default : Optimisation = Adam, Activation = ReLU, Threshold = 0.5, CNN layer = 3

Figure 5: *Hyperparameters*

The following hyperparameters are used during the training and validating phases.

4.2.1 Dropout Rate

We chose dropout rates of 0.5 or 0.7 to understand whether the dropout rate affects the generalisation of the model. It is a regularisation method to prevent overfitting and improve the generalisation ability of the model by randomly dropping the number of nodes during the training phase. We aim to select the optimal hyperparameter that best suits our model (Brownlee, 2019).

4.2.2 Batch Size

It is set at 30 or 50 to check how the convergence of the model changes when the batch size varies.

4.2.3 Number of Epochs

It is set at either 1 or 3. The number of epochs refers to a whole complete process through the entire training dataset. Higher epochs may result in overfitting whereas lower epochs may lead to underfitting. As a result, it's important to find the optimal number of epochs to determine when the performance of the model stops improving.

4.2.4 Learning Rate

It is set at 0.001 or 0.01 to observe how different learning rates impact the accuracy and convergence of the model. The learning rates determine how the parameters are updated during the training process. A smaller learning rate may take longer for the model to reach the global optimal point. Larger learning rate can quickly cause the model to reach sub-optimal point rather than the global optimal point. Therefore, we aim to examine different learning rates to find the most optimal one for our model.

4.2.5 Default values

- **Optimization Method: Adam**
  The Adam optimizer applies an adaptive learning rate to the updates of each parameter, allowing for learning adjustments at appropriate speeds based on individual characteristics (Fedorenko, 2019). This dynamic adjustment is crucial in multi-label problems where each label may have different characteristics. Moreover, Adam facilitates smooth optimization through the estimation of the first and second moments, which is effective in complex situations such as label interdependencies (Fedorenko, 2019).

- **Activation Function: ReLU**
  ReLU (Rectified Linear Unit) outputs zero for negative inputs and maintains positive inputs unchanged. This provides non-linearity in the neural network while keeping computational costs low. Using ReLU in multi-label classification allows neurons to be activated, emphasising important features for label detection while reducing unnecessary interactions between labels when not activated. This helps prevent overfitting and enhances the generalisation ability of the model (Song et al., 2021).

- **Threshold: 0.5**
  In binary classification problems, a threshold setting determines that an output probability above 0.5 indicates belonging to a certain class. In a multi-label context, an independent judgement for each label is necessary, and 0.5 provides a balanced cut-off point (Tang et al., 2009). This consistent criterion aids in clearly distinguishing the model's predictions.

- **CNN Layer: 3 Layers**
  The number of CNN layers is determined to ensure the model can adequately extract necessary information from images. Three layers offer depth sufficient to learn from basic to more complex features without excessively increasing computational complexity. In multi-label problems, where various labels may have different characteristics, the use of multiple layers is appropriate to extract features at various levels (Wang et al., 2016).

4.3 Performance Analysis with best model

Epochs	Dropout	Batch Size	Learning Rate	Learning Time	Train Loss	Validation Loss	Test F1-score
1	0.5	50	0.001	77 min.	0.07767	0.00298	0.91891
1	0.5	30	0.001	103 min.	0.08758	0.00069	0.91754
3	0.5	50	0.001	295 min.	0.08599	0.00057	0.91575
3	0.5	30	0.001	409 min.	0.09396	0.00094	0.91512
3	0.7	30	0.001	370 min.	0.10864	0.00046	0.91225
1	0.7	30	0.001	102 min.	0.14768	0.00092	0.90766
3	0.5	50	0.01	291 min.	0.06266	0.00108	0.90645
1	0.5	50	0.01	77 min.	0.12200	0.00298	0.90370
3	0.5	30	0.01	402 min.	0.09027	0.00258	0.90241
3	0.7	50	0.001	260 min.	0.11368	0.00236	0.89850
1	0.5	30	0.01	102 min.	0.15769	0.00184	0.89558
3	0.7	30	0.01	374 min.	0.12618	0.00182	0.89433
1	0.7	50	0.001	73 min.	0.11559	0.00212	0.89283
1	0.7	30	0.01	104 min.	0.15169	0.00137	0.88766
3	0.7	50	0.01	260 min.	0.13142	0.00494	0.88600
1	0.7	50	0.01	73 min.	0.12677	0.00151	0.87308

*default : Optimisation = Adam, Activation = ReLU, Threshold = 0.5, CNN layer = 3

Figure 6: *Performance Table*

The results are listed in the order that showed the highest performance among all 16 hyperparameter combinations.

4.3.1 Performance Analysis

Based on this data, the combination of 1 Epoch, 0.5 Dropout rate, Batch Size of 50, and a Learning Rate of 0.001 shows the most balanced performance, achieving the highest F1-score of 0.91891. This configuration also provides a relatively quick training time (77 minutes) and low training and validation losses, demonstrating a good balance between efficiency and performance. Looking at the overall trends, the number of epochs and batch sizes do not significantly impact performance. In contrast, the dropout rate and learning rate turn out to be crucial factors, especially notable is that lower dropout rates and learning rates lead to marked performance improvements.

4.3.2 Efficiency Analysis

There is a notable difference in training times depending on batch size and learning rate. Larger batch sizes and lower learning rates (0.001) tend to shorten the training time. Particularly, the fact that training time does not increase significantly from Epoch 1 to Epoch 3 suggests efficient learning processes are in place.

4.4 Comparison Analysis

Based on overall trends and results, a detailed analysis was conducted by comparing the top three and bottom three hyperparameter combinations.

Epochs	Dropout	Batch Size	Learning Rate	Learning Time	Train Loss	Validation Loss	Test F1-score
1	0.5	50	0.001	77 min.	0.07767	0.00298	0.91891
1	0.5	30	0.001	103 min.	0.08758	0.00069	0.91754
3	0.5	50	0.001	295 min.	0.08599	0.00057	0.91575

Figure 7: *Top3*

Epochs	Dropout	Batch Size	Learning Rate	Learning Time	Train Loss	Validation Loss	Test F1-score
1	0.7	30	0.01	104 min.	0.15169	0.00137	0.88766
3	0.7	50	0.01	260 min.	0.13142	0.00494	0.88600
1	0.7	50	0.01	73 min.	0.12677	0.00151	0.87308

Figure 8: *Bottom3*

By comparing the top three performances of the hyperparameter combinations, it is evident that the higher performances are highly driven by a learning rate of 0.001 and a dropout rate of 0.5, regardless of how number of epochs and batch size change. As shown in the table above, the top three models all have test F1 scores above 0.915. Therefore, a learning rate of 0.001 and a dropout rate of 0.5 are crucial factors in enhancing the performance of the models.

In contrast, by comparing the last three performances of the hyperparameter combinations, we can clearly see that they are largely impacted by a learning rate of 0.01 and a dropout rate of equal to 0.7 regardless of the variations of the numbers of epochs (either 1 or 3) and the batch size (either 30 or 50). The models consistently perform under these hyperparameter combinations with test F1 scores between 0.87308 and 0.88766. Therefore, the models' poor performance might be hugely impacted by a dropout rate of 0.7 and a learning rate of 0.01.

5 Discussion and Conclusion

5.1 Discussion

5.1.1 Superiority of the Hybrid Model Over the Standalone CNN Model

Shown performance that the hybrid model combining CNN, LSTM, and Transformer performs better in multi-label image classification tasks is supported by several rational reasons.

1. **Integrated Processing of Multidimensional Data**
   The hybrid model merges various architectures like CNN, LSTM, and Transformer, enabling simultaneous processing of visual and textual information in images. This model can extract crucial features from each data type and integrate them for more accurate label prediction (Jena et al., 2021).
2. **Understanding Relationships Between Labels**
   In multi-label classification, multiple labels can exist simultaneously on a single image, and these labels often have intrinsic relationships. Using Transformers, the attention mechanisms can effectively capture these complex inter-label relationships (Alfigi, 2023).
3. **Complex Pattern Recognition**
   The hybrid model combines the powerful image processing capabilities of CNN, the sequential data handling of LSTM, and the complex interdependence learning of Transformers. This allows for the recognition of more complex patterns and subtle differences, particularly useful in classifying the characteristics and relationships of various objects within an image accurately (Zhou et al., 2019).
4. **Enhanced Performance**
   By combining the strengths of different models, the hybrid approach can address challenges that are difficult for a single model, enhancing overall performance. For instance, while CNN excels at extracting visual features from images, LSTM and Transformers excel in interpreting these features within temporal or contextual frameworks.

5.1.2 Captioning Analysis

To facilitate the analysis of textual data, preprocessing steps such as converting text to lowercase, removing stopwords, and extracting lemmas were performed. Following this, the Term Frequency-Inverse Document Frequency (TF-IDF) method was used to numerically evaluate the importance of words corresponding to each class. Based on the TF-IDF scores, the most significant words for each class were identified and visualised. The top 10 words were extracted to rank their significance, and their visual representation was enhanced using WordCloud.

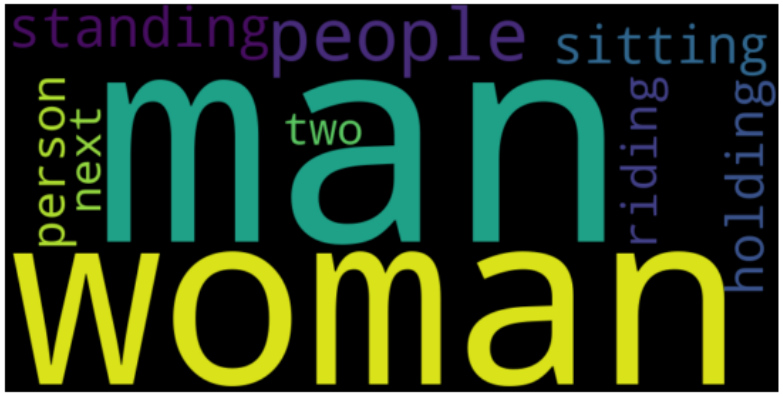

Figure 9: *wordcloud example: Class1 'Human'*

Figure 10: *wordcloud example: Class3 'Others'*

Class 1 - Human
This class includes words related to human activities or characteristics, thus it is designated as 'human'.

Class 2 - Bicycle
Words associated with bicycles dominate, including terms directly related to cycling such as pedals and helmets, therefore it is designated as 'bicycle'.

Class 3 - Others
This category encompasses a variety of elements with diverse specific themes, making it difficult to summarise with a specific keyword, hence it is designated as 'others'.

Class 4 - Motorcycle
Words related to motorcycles, including engines and wheels, are prominent, thus it is designated as 'motorcycle'.

Class 5 - Airplane
Words associated with airplanes, such as aviation and jet, are included, thus it is designated as 'airplane'.

Class 6 - Bus
Words related to bus travel, such as travel and bus stops, are typical, therefore it is designated as 'bus'.

Class 7 - Train
Words central to train travel, such as railway and carriages, are highlighted, thus it is designated as 'train'.

Class 8 - Truck
Words related to trucks, such as cargo and transport, are key, therefore it is designated as 'truck'.

Class 9 - Boat
Words associated with boating, such as sailing and sea, are included, therefore it is designated as 'boat'.

Class 10 - Street
Words related to the street, such as road and sidewalk, are included, therefore it is designated as 'street'.

Class 11 - Fire / Fire Related Materials
Words related to fire, such as blaze and smoke, are prominent, thus it is designated as 'fire'.

Class 13 - Sign (Stop Sign Related)
Words related to stop signs, such as signal and caution, are included, thus it is designated as 'sign'.

Class 14 - Parking
Words related to parking, such as parking lot and parking area, are chosen as key words, thus it is designated as 'parking'.

Class 15 - Bench
Words related to benches, such as rest and park, are included, therefore it is designated as 'bench'.

Class 16 - Bird
Words associated with birds, such as flight and feathers, are included, thus it is designated as 'bird'.

Class 17 - Cat
Words central to cats, such as pet and play, are highlighted, thus it is designated as 'cat'.

Class 18 - Dog
Words related to dogs, such as pet and bark, are included, thus it is designated as 'dog'.

Class 19 - Horse
Words related to horses, such as riding and racing, are key, therefore it is designated as 'horse'.

5.2 Conclusion

In conclusion, our study demonstrates that the hybrid model, consisting of convolutional neural networks (CNNs), long short-term memory (LSTM) and transformers has performed better compared to the use of CNN model alone in multi-label classification tasks. This outstanding performance can be explained by the unique strengths of each component: CNN excels well in image processing, LSTM handles effectively with the sequences of data and transformers specialise in extracting crucial data through attention mechanisms.
By integrating textual data captions and visual data images, our hybrid model significantly improves the accuracy of the model by leveraging information from different data modalities. This integration approach allows the model to learn the relationships between images and their related captions, resulting in more satisfied predictions. Ablation studies and hyperparameter tuning were conducted to optimise the accuracy of the model, identifying the best hyperparameter combinations for training the model.
The success of utilising hybrid models is evident that it is crucial to integrate different data modalities through the training process to improve the accuracy of the model, resulting in better predictions. In future, there's room for further improvements for refining the structure of the hybrid models and advancing the performance of the model for multi-label classification tasks and multi-modal data. Continued research and experiments are essential for handling more complex datasets through the use of hybrid models in real life.

6 Appendix

6.1 Runtime

We are aware of the long run times resulting from experimenting with lots of hyperparameter combinations. Long run times can lead to inefficiencies in resource use and increased costs. However, excessively reducing the range or number of hyperparameters, which could impair the model's learning capacity and lead to unexpected performance degradation, was not considered.

Parameter tuning part
Epoch 1 : 8 number of cases, 11 hour 51 minute
Epoch 3 : 8 number of cases , 44 hour 21 minute

6.2 Instructions on how to run the code

To run the code successfully, please follow these steps:

1. Access the link:
   `https://drive.google.com/file/d/1Azj_6MD8__lmV1S9w_4WGAXldwSWXLYf/view?usp=drive_link`

2. Open the file with Google Colaboratory.

3. Run the code in the following order:

   (a) Mount your Google Drive in the first cell (sign-in with your Google account will be required).

   (b) Subsequent cells should automatically handle data download and execution (note that the data size is around 438.39 MB, ensure you have at least this much free space on your drive).

4. The predicted result files that submitted in Kaggle are in this link:
   `https://drive.google.com/drive/folders/1NpKWnKWSqXDazpimGqcmhkJHRLtDHHi0?usp=sharing`

6.3 Specifications

Software

- **Language:** Python3.11
- **Web-interface:** Jupyter Notebook(Local environment), Google Colab

Hardware

- **Edition:** Windows 10 Pro
- **Processor:** Intel(R) Core(TM) i7-7700K CPU @ 4.20GHz 4.20 GHz
- **Installed memory (RAM):** 16.0 GB

- **System type:** 64-bit operating system, x64-based processor
- **Version:** 22H2

6.4 Evaluation Criteria

Evaluation Category	Evaluation Criteria	Report
Introduction	Aim of the study	p1
	Importance of the study	p1
	General introduction of used method and motivation for a solution	p1
Literature Review (Related works)	Literature Review on Deep Learning for Multi-Label Image Classification	p2-3
	Example of a Review of Specific Papers	p3
	Integration with Existing Research and Future Research Directions	p3
Techniques	Data Loading and Preprocessing	p4
	Image Transformations	p4
	Dataset Splitting	p4-5
	Model Training and Evaluation	p5
	Model Architecture and Pipeline Design	p5-7
	Training and Evaluation Pipeline	p7-8
Experiments and Results	Ablation studies	p8
	Hyperparameter Analysis	p9-10
	Performance Analysis with best model	p10-11
	Comparison Analysis	p11
Discussion and Conclusion	Discussion	p12-14
	Conclusion	p14
Appendix	Runtime	p15
	Instructions on how to run the code	p15
	Specifications	p15-16
	Evaluation Criteria	p16
	References	p17-18

Figure 11: *Evaluation Criteria*

7 Reference

Alfigi, H. (2023). Multi-label and single-label text classification using standard machine learning algorithms and pre-trained bert transformer. Cankaya.edu.tr. http://hdl.handle.net/20.500.12416/6414

Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. Journal of Big Data, 8(1). https://doi.org/10.1186/s40537-021-00444-8

Brownlee, J. (2018, December 3). A Gentle Introduction to Dropout for Regularizing Deep Neural Networks. Machine Learning Mastery. https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/

Fedorenko, Y. S. (2019). The Simple Approach to Multi-label Image Classification Using Transfer Learning. Studies in Computational Intelligence, 207–213. https://doi.org/10.1007/978-3-030-30425-6_24

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. ArXiv.org. https://arxiv.org/abs/1512.03385v1

J, B. (2021). Image Caption Generation Using CNN-LSTM Based Approach. Proceedings of the First International Conference on Combinatorial and Optimization, ICCAP 2021, December 7-8 2021, Chennai, India. https://doi.org/10.4108/eai.7-12-2021.2314958

Jena, B., Saxena, S., Nayak, G. K., Saba, L., Sharma, N., & Suri, J. S. (2021). Artificial intelligence-based hybrid deep learning models for image classification: The first narrative review. Computers in Biology and Medicine, 137, 104803. https://doi.org/10.1016/j.compbiomed.2021.104803

Laith Alzubaidi, Bai, J., Aiman Al-Sabaawi, Santamaría, J., Albahri, A. S., Bashar, Fadhel, M. A., Manoufali, M., Zhang, J., Al-Timemy, A. H., Duan, Y., Abdullah, A., Farhan, L., Lu, Y., Gupta, A., Albu, F., Amin Abbosh, & Gu, Y. T. (2023). A survey on deep learning tools dealing with data scarcity: definitions, challenges, solutions, tips, and applications. Journal of Big Data, 10(1). https://doi.org/10.1186/s40537-023-00727-2

Papers with Code - Multi-Label Image Classification. (n.d.). Paperswithcode.com. Retrieved May 14, 2024, from https://paperswithcode.com/task/multi-label-image-classification

Raschka, S. (2018). Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning. ArXiv.org. https://arxiv.org/abs/1811.12808

Rusli, A. (2021, June 24). 5 Core Challenges In Multimodal Machine Learning — Mercari Engineering. Engineering.mercari.com. https://engineering.mercari.com/en/blog/entry/20210623-5-core-challenges-in-multimodal-machine-learning/

Seo, P. H., Nagrani, A., Arnab, A., & Schmid, C. (2022). End-to-End Generative Pretraining for Multimodal Video Captioning. Openaccess.thecvf.com. https://openaccess.thecvf.com/content/CVPR2022/html/Seo$_{End - to - End_Generative_Pretraining_for_Multimodal_Video_Captioning_CVPR_2022_paper.html}$

Song, L., Wu, J., Yang, M., Zhang, Q., Li, Y., & Yuan, J. (2021). Handling Difficult Labels for Multi-label Image Classification via Uncertainty Distillation. https://doi.org/10.1145/3474085.3475406

Tang, L., Rajan, S., & Narayanan, V. (2009). Large scale multi-label classification via metalabeler. The Web Conference. https://doi.org/10.1145/1526709.1526738

Tarekegn, A. N., Ullah, M., & Cheikh, F. A. (2024, March 3). Deep Learning for Multi-Label Learning: A Comprehensive Survey. ArXiv.org. https://doi.org/10.48550/arXiv.2401.16549

Wang, J., Yang, Y., Mao, J., Huang, Z., Huang, C., & Xu, W. (2016). CNN-RNN: A Unified Framework for Multi-label Image Classification. ArXiv (Cornell University). https://doi.org/10.48550/arxiv.1604.04573

Zhang, Y., Wang, Y., Liu, X.-Y., Mi, S., & Zhang, M.-L. (2020). Large-scale multi-label classification using unknown streaming images. Pattern Recognition, 99, 107100. https://doi.org/10.1016/j.patcog.2019.107100

Zhou, T., Li, Z., Zhang, C., & Ma, H. (2019). Classify multi-label images via improved CNN model with adversarial network. Multimedia Tools and Applications, 79(9-10), 6871–6890. https://doi.org/10.1007/s11042-019-08568-z