Multimodal in Multi-Label Classification: A Report

Technical Report · June 2023

DOI: 10.13140/RG.2.2.29898.54722

3 authors:

Chongyangzi Teng
The University of Sydney
4 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Pengwei Yang
The University of Sydney
11 PUBLICATIONS **26** CITATIONS
SEE PROFILE

Mengshen Guo
The University of Sydney
2 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Multimodal in Multi-Label Classification: A Report

Chongyangzi Teng[1†], Pengwei Yang[1†*], and Mengshen Guo[1]

[1]*School of Computer Science, The University of Sydney, Sydney NSW 2000, Australia*
[†]*Equal contribution*
[*]*Corresponding author*

Abstract

Multi-label classification is a general task in the machine learning field. This paper proposes a combined framework that performs well for this type of tasks. In the study, images are concatenated with the captions as the input and a word embedding model is combined with an attention-based model or a CNN model. A set of experiments were conducted to compare the varied combination of well-known backbones. The result shows that our framework outperforms the single image-based model.

1 Introduction

Deep neural network models can be implemented in addressing various aspects of our daily lives [6], such as IoT energy services [12][13]. As a general task in the machine learning field, multi-label classification can also be well performed by deep neural networks [10]. For instance, it could be utilized to label multiple fashion items in an image; to diagnose multiple medical conditions of a patient from a medical image, and to classify movies into different topics and categories based on videos, posters, and or word descriptions. Due to its wide applications, it is important to continue developing techniques that can contribute to this area. In addition, information is usually distributed across various formats, and the need to build a model that can effectively process multiple forms of information such as images, videos, and text has also become increasingly crucial.

Traditionally, the development in convolutional neural networks (CNN) has been driving the success of computer vision, from AlexNet to ResNet and continues to develop [4][5]. More recently, a range of transformer models has emerged and has achieved competitive performance in both natural language processing (NLP) and computer vision, examples including Vision Transformer (ViT), Data-efficient Transformer (DeiT), and Swin Transformer [3][7][9].

This study will provide an overview of related works in the images and text multi-label classification field in section 2 and then will review computer vision and NL techniques BERT, ResNet-50, ViT, DeiT and Swin Transformer in more detail in section 3. The study will then apply these techniques to build a multi-label classifier using images and captions data. The experiment set-up and results will be discussed in section 4. Finally, the study will be concluded with our learning in section 5.

2 Related Works

ResNet-50 is a traditional CNN backbone model as part of the Residual Network (ResNet) family [4]. The primary innovation in ResNet is the residual block, which facilitates skip connections, allowing the network to be much deeper while still optimizing effectively. With 50 layers, ResNet-50 is widely used in various computer vision tasks.

Transformers have led to significant advancements in deep learning, primarily for NLP tasks. More recently, since the introduction of ViT model, transformer-based models have also been successfully applied to computer vision tasks [3]. Below is a high-level overview of popular transformer-based models and their timelines.

The ViT model applies the transformer architecture directly to sequences of image patches [3]. It divides the input image into fixed-size non-overlapping smaller patches, linearly embeds them into flat vectors, and then feeds them into a standard transformer model.

DeiT is a more data-efficient version of the Vision Transformer [9]. By using distillation with a strong convolutional neural network teacher and knowledge distillation during pre-training, DeiT achieves competitive performance even when trained on smaller datasets. Additionally, it introduces several training techniques to further improve the Transformer model's performance.

Swin Transformer introduces a shifted windowing mechanism for local self-attention in the transformer model [7]. This design enables better hierarchical feature learning and reduces computational complexity. Swin Transformer excels in image classification, object detection, and semantic segmentation [7].

3 Methodology

3.1 Pre-processing

In this paper, we applies data normalization and data augmentation techniques for image data, and word embedding for captions. The details can be seen below.

3.1.1 Data Augmentation and Data Normalization

Random horizontal flips and vertical flips to augment the image can be used for data augmentation. The main purpose of it is to increase the training data size by creating transformed versions of images in the dataset. These techniques enable the model to be more robust and less sensitive to the object orientation in the images. Another benefit is to prevent overfitting since the model is less likely to memorize the training data and instead learn more patterns.

Regarding data normalization, we use the mean and standardization of ImageNet to normalize our data. ImageNet is a large-scale and diverse dataset used to pre-trained many models in computer vision, hence it is highly efficient to use the above values to normalize our small dataset.

3.1.2 Tokenizer

We use BertTokenizer to process the raw caption data prior to use as input into a BERT model [2]. BertTokenizer breaks down the caption sentences into tokens, including the classifier token at the front of the caption sentence and the separator token at the end of the caption sentence. It can also allow padding to treat input text with different lengths [11].

3.2 Transfer Learning and Pre-trained Model

Deep neural networks rely heavily on availability of sufficient training data due to its poor generalization ability [1] and large amounts of parameters in the model architecture. Transfer learning [8] [9], a concept inspired by how human brains can use previous experience to solve new unseen problems, is introduced to tackle this limitation. Using the concept of transfer learning, models can be trained on large-scale dataset to capture knowledge. It can then be fine-tuned to transfer the learned knowledge to a different domain where sample size is limited. In our experiment, we utilised pre-trained models that are available on HuggingFace and then fine-tuned using our image and caption data [11].

3.2.1 BERT Word Embedding

Pre-trained Bidirectional Encoder Representations from Transformers (BERT) model [2] will be used for caption information and then embed into the final model with image information. Unlike traditional word embedding techniques such as Word2Vec and GloVe which can only provide fixed embedding, BERT can provide a contextual embedding. This can help the model to understand the meaning our our caption data in the study's context. In addition, the pre-trained BERT model can help to leverage on the knowledge that learned from larger training dataset and then fine-tune on smaller caption dataset for the study.

3.3 Backbone Models

3.3.1 ResNet-50

ResNet-50 is designed to address the challenge of training Deep Residual Learning for Image Recognition for large-scale image recognition tasks [4], which includes residual connections that allow gradients to propagate more efficiently during backpropagation, alleviating the problem of gradient disappearance. ResNet-50 includes residual connections that allow gradients to propagate more efficiently during backpropagation [4], alleviating the problem of gradient disappearance. At the same time, ResNet-50 uses a bottleneck structure which consists of three convolutional layers (1x1, 3x3 and 1x1) in each residual block [4]. This design reduces the number of parameters and computational complexity, while maintaining the expressive power of the network. The bottleneck structure helps the model scale to a depth of 50 layers [4], which is the reason why the model is called ResNet-50. It also employs global average pooling, which reduces the parameter amount and the computation resources required, thereby reducing the risk of overfitting. ResNet-50 utilizes batch normalization after each convolutional layer [4], this can help to accelerate training time and improve the stability of the model. ResNet-50's novel architecture and innovations have made it a widely adopted model for a variety of computer vision tasks, including image classification, object detection and semantic segmentation [4].

3.3.2 Vision Transformer (ViT)

Transformer model is traditionally used for NLP problems. However, since ViT was introduced in 2020 [3], the transformer model has now become a very popular model architecture for image classification. The model works by first splitting the input images into smaller patches and then flattening them to produce a linear embedding. A positional embedding will also be added to feed into the standard transformer encoder in a similar way as NLP. An architecture of a ViT is shown in Fig.1.

This multi-head attentional layer makes ViT process different parts of the input images feature well. Compared to CNN, the ViT model requires less training time and is more computationally efficient. It has competitive accuracy performance especially when used with models pre-trained on large datasets. However, supervised training on large datasets also has its limitations in areas where labeling data is costly and time-consuming[3].

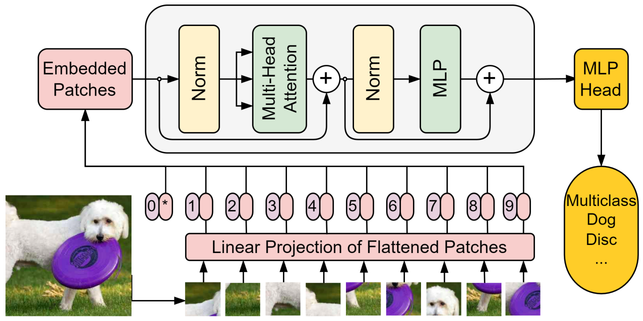


Figure 1: Vision Transformer (ViT) main architecture

3.3.3 Data-efficient Transformer (DeiT)

Compared to ViT, DeiT is a transformer model that can be more efficiently trained. It requires less data and computational resources for similar performance [9]. Different from ViT, DeiT has an additional distillation token using a teacher-student strategy to reduce reliance on data. This distillation token interacts with the class and patch tokens and its objective is to enable the model to extract learning from the output of the teacher instead of a true label. An architecture of a DeiT model is shown in Fig.2.

There are different versions of DeiT models, in this study we used a smaller variant, DeiT-Small model. The model is pre-trained on ImageNet-1k at a resolution of 244x244 pixels. This pre-trained model divides input images into patches of 16x16 resolution and then linearly embedded into a transformer encoder. The reason for using this model for our study is due to its advantages in a smaller number of parameters and model size given the constraint of our computational power.

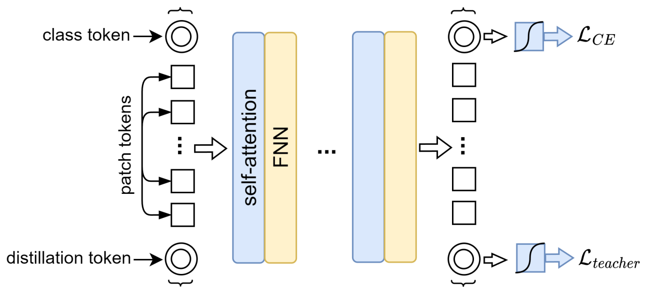


Figure 2: DeiT main architecture

3.3.4 Swin Transformer

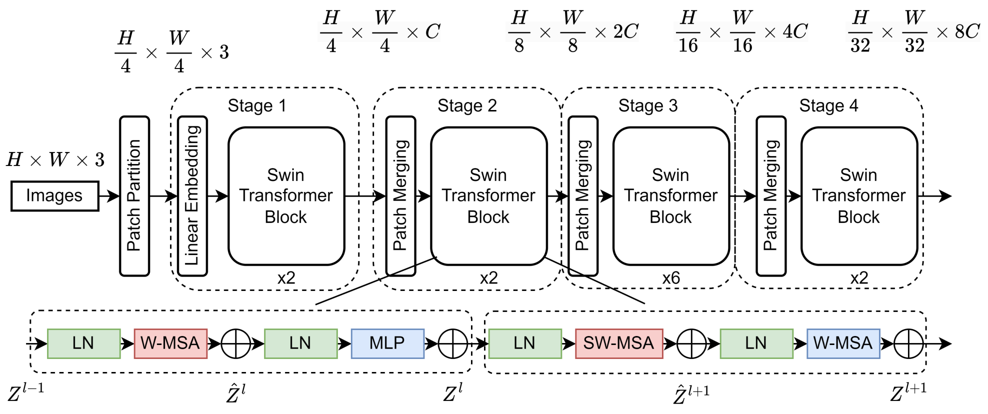


Figure 3: Swin Transformer main architecture

Swin Transformer is proposed to address the difficulty of scaling to high-resolution images with regard to the traditional Vision Transformers [3] [7]. Swin transformer introduces a novel hierarchical architecture including a shift-based window mechanism. Swin Transformer divides the input image into smaller non-overlapped patches instead of using a fixed patch size and processes patches in a hierarchical manner. The details of the Swin Transformer are as follows.

Patch embedding includes the patch partition and linear embedding as shown in Fig.3. This step makes use of a convolutional layer to do the downsampling task. Given the input size $H \times W \times 3$, the output size is $\frac{H}{4} \times \frac{W}{4} \times C$, where C is the embedding dimension. For instance, given the color picture with input size (224, 224, 3), the output of this step is (3136, 96) when using a 2D convolutional layer with a kernel size of (4, 4) as well as the stride, and the embedding dimension is set to 96. It can reduce the token size and thus the computation.

Regarding window partition, in the aforementioned instance, given the input feature map (56, 56, 96), the window partition step is to split the feature map into 64 windows where each window size of 7×7. The output of this step is (64, 7, 7, 96).

Window Multi-head Self Attention (W-MSA) aims to compute the self-attention within each window (See Fig.3). W^Q, W^K and W^V are computed parallelly, the size is (3, 64, 3, 49, 32) respectively stands for 3 matrices, 64 amount of windows, 3 heads, 49 tokens, and each token with 32 dimensions in 1 head. Thus, the output size of W-MSA is (64, 49, 96).

Shifted Window Multi-head Self Attention (SW-MSA) is the key idea of the Swin Transformer (See Fig.3). In this step, the windows have been shifted to compute the overlapped attention among windows. To enable the computation among shifted windows, a Mask mechanism has been used. The motivation is to enable the same computation of the shifted windows.

The output of the abovementioned two window mechanisms is 7×7 windows with an amount of 64. However, it cannot work as the input to the next Swin Transformer block since the input is supposed to be 56×56. Therefore, the window reverse mechanism reshapes the size from (64, 49, 96) to (56, 56, 96) which is the desirable input of the next Swin Transformer block.

3.3.5 Proposed Model Architecture

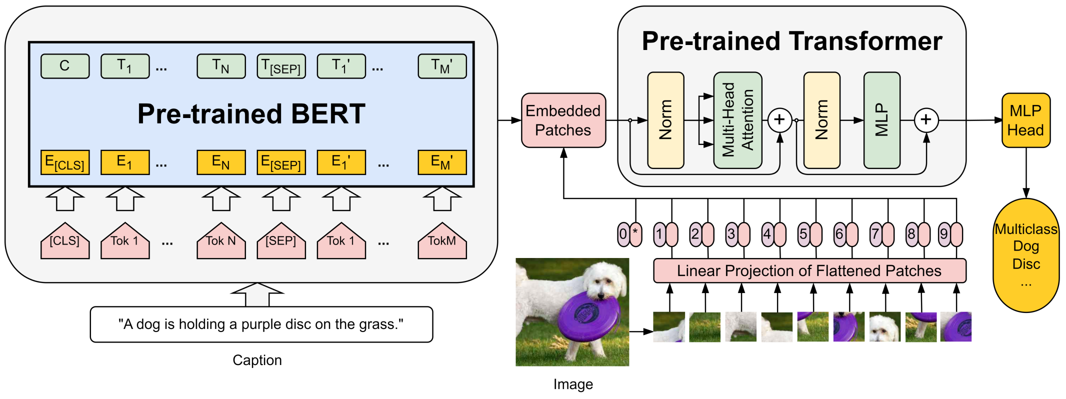


Figure 4: Our model architecture

In this paper, we take an image with a caption as the input and make use of two pre-trained models, i.e., a pre-trained BERT model and a pre-trained Transformer model (See Fig.4). We use pre-trained BERT to output the word embeddings. Note we do not consider backpropagation through the BERT model in this paper. Our Transformer models take the combination of word vectors and images as the inputs and output a probability distribution regarding each class. In addition, we set a threshold probability, the input data are assigned to a defined class when its output conditional probability is higher than the predefined threshold. In contrast to models that exclusively process either images or textual data as independent inputs, our proposed model uniquely incorporates both visual and linguistic information, fostering a more comprehensive understanding of the input data.

4 Experiment and Results

4.1 Dataset Description and Experiment Setup

In this study, we will be utilizing different techniques to build deep-learning models for multi-label classification problems. We have training and testing sample sets of 30,000 and 10,00 images respectively. Each image also has a caption that describes the image in text format. The samples have 19 labels from 1 to 19, each image can have multiple labels.

The experiment compares different pre-trained models including ResNet-50, ViT, DeiT-Small, and Swin Transformer. It also compares how the use of caption information can help to improve classifier performance by the use of the pre-trained BERT model to output as word embedding. The findings from the comparative analysis and ablation analysis will guide us to select a final model considering the trade-off between model performance and model size. A hyperparameter tuning will then be performed on the chosen model architecture to fine-tune the model.

4.2 Performance metrics

In the experiment, the performance of the models will be evaluated with the following metrics:

4.2.1 F1 Score

F1 score is the harmonic mean of precision and recall. It is a performance measure that considers the balance between precision and recall. The formula for the F1 score is:

$$F1 = 2 * \frac{precision * recall}{precision + recall}$$

where the Precision stands for the number of positive predictions divided by the total number of positive class values in the predicted dataset. And Recall denotes the number of positive predictions divided by the number of positive class values in the test dataset.

4.2.2 Multi-label Soft Margin Loss

In this paper, we use multi-label soft margin loss, which is designed for multi-label classification tasks. For instance, an image could be simultaneously assigned to a cat, a dog, and a person class. Each class is considered independent, so, compared with CrossEntropyLoss, the sum of predicted probabilities is not forced to be 1. Given the following preliminaries, $y \in 0, 1^N$ is ground truth, where N is the number of classes.$x \in R^N$ as the raw predicted output from your model, i.e., logits. $l(x, y)$ be the "MultiLabelSoftMarginLoss" for one instance in the batch.

The mathematical formula can be seen below.

$$l(x, y) = -\frac{1}{N}\sum_{i=1}^{N}[y_i * log(\sigma(x_i)) + (1 - y_i) * log(1 - \sigma(x_i))]$$

where $l(x, y)$ is the MultiLabelSoftMarginLoss for one instance in the batch. x are the raw predicted outputs from your model, i.e., logits. y is the ground truth. N is the number of classes. $\sigma(x_i)$ is the sigmoid function applied to x_i, which is $\frac{1}{1+e^{x_i}}$. The sum is over all classes.

4.3 Experiment results

Table 1 shows the results of using the same architecture as described in section 3.3.5 but with a different pre-trained model backbone. Due to constraints in time and computation resources, we only train the model using 1000 images to compare the performance. We find all three transformer-based models ViT, DeiT small, and Swin are a lot faster to score compared to the convolutional model ResNet-50. This also confirms our literacy research that the transformer models are more computationally efficient compared to convolutional models. In terms of F1 Score, the Swin model performs the best followed by ViT and ResNet-50. Although the performance of DeiT Small is the weakest among the four, considering its small size and faster fine-tuning time, we will choose this model as our model backbone. In addition, the performance results are indicative only due to the small 1000 training set used, the results can vary. For a more rigorous comparison, larger training data should be used.

Table 1: Comparison of different backbone model

Model Backbone	Fine-tune time	Model size	Number of parameters	Test - F1 score
ResNet-50	448s	98MB	24.37M	0.6610
ViT	16s	346MB	82.56M	0.6954
DeiT-Small[†]	37s	88MB	21.03M	0.6260
Swin	43s	353MB	83.70M	**0.7673**

[†] Final Model Choice

Further ablation analysis is performed to understand the effect of different model components. The results are shown in Table 2. When training data increased from 1000 images to 27000 images (90 percent of training data). The F1 score improved by 0.22 from 0.6260 to 0.8474. In addition, caption also adds valuable information to improve the model performance, without it the model performance will deteriorate by 0.19 in the F1 score. Considering the benefit of utilizing more training data and captions, we will be utilizing 27000 training images and caption information for our final model.

Table 2: Ablation analysis results

Model element	Model 1	Model 2[†]	Model 3
Use of Captions	Yes	Yes	No
Model Backbone	DeiT Small	DeiT Small	DeiT Small
Number of images used	1000	27000	27000
Test data F1 score	0.626	**0.8474**	0.6612
F1 Score changes to Model 1	Baseline	+0.22	-0.19

[†] Final Model Choice

4.3.1 Hyperparameter Analysis

We performed the hyperparameter analysis on the proposed model using DeiT-small as the backbone model.

Table 3: Final Model - Threshold Analysis

Threshold	0.1	0.2	0.3	0.4	0.5
Testing data F1 score	0.7485	0.8225	0.8388	**0.8474**	0.8361

We have tuned on a set of thresholds from 0.1 to 0.5 using F1 score as evaluation criteria. The model will predict a probability of label membership and if the probability is greater than a threshold, then the image will be assigned with the label. A global threshold has been applied to all labels; this means only one threshold is applied. The results of the threshold tuning are shown in table 3 below and we found 0.4 gives the best F1 score.

Table 4: Final Model - Batch size and learning rate

Batch size	64	64	16	32
Learning Rate	0.001	0.005	0.001	0.001
Threshold	0.4	0.4	0.4	0.4
Testing data F1 score	**0.8474**	0.8410	0.8263	0.8354

In addition, we also tried a combination of batch size and learning rate and found a learning rate of 0.001 and a batch size of 64 provides the best performance for our chosen model. The parameter analysis result is shown in Table 4.

4.3.2 Final model

The details of our final model modules are shown in table 5

5 Discussion and Conclusion

This paper first reviews related works in the images and text neural network model field. It then conducts an experiment on a multi-label classification problem to compare different model architectures. After model comparison and ablation analysis, the chosen final model utilises a pre-trained BERT model for caption information and combined it with a pre-trained DeiT small transformer for images. The model is fully trained on Google Colab, considering the computational resources constraint, the model has achieved sound performance with an F1 score of 84.7% on the testing dataset.

Table 5: Final Model Details

Model parameters	Final Model
Use of Captions	Y - BERT embedding
Model Backbone	DeiT Small
Number of images used	27000
Batch size & Learning rate	BS = 64, LR = 0.001
Threshold	0.4
Epoch	10
Test data F1 score	0.8474
Model Size	86M
Fine-tuning time	4hr20mins
Testing time	1hr10mins

For the model comparison analysis, We only trained on 1000 images to compare the performance between different backbone models. In future, to improve the rigorousness of the experiments, more training data should be used and hyper-parameter tuning should also be performed for each model before comparing against each other.

The paper used DeiT-small as the backbone model for the final model after consider trade-off between performance and efficiency. To improve the model performance, more complicated pre-trained models such as the larger DeiT model, Swin, or BEIT model could be used. In addition, the model performance can also be improved by using a larger training data dataset to fine-tune the model. We have demonstrated the benefit of a larger training dataset for fine-tuning for specific use in our ablation analysis. In terms of hyper-parameter tuning, in the study we have only tested on a global threshold, however, the dataset is imbalanced as such using multi-threshold for different labels could further improve the model performance.

In the future, we may try a larger modal with potential trustworthy machine learning techniques [14]. For the contextualized embedding technique, the BERT, we may fine-tune the BERT model by freezing and unfreezing some of the BERT parameters.

References

[1] Belkin, M., Hsu, D., Ma, S., Mandal, S.: Reconciling modern machine-learning practice and the classical bias–variance trade-off. Proceedings of the National Academy of Sciences **116**(32), 15849–15854 (2019)

[2] Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding (2019)

[3] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)

[4] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)

[5] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Pereira, F., Burges, C., Bottou, L., Weinberger, K. (eds.) Advances in Neural Information Processing Systems. vol. 25. Curran Associates, Inc. (2012)

[6] LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. nature **521**(7553), 436–444 (2015)

[7] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. pp. 10012–10022 (2021)

[8] Thrun, S., Pratt, L.: Learning to learn. Springer Science & Business Media (2012)

[9] Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jégou, H.: Training data-efficient image transformers & distillation through attention. In: International conference on machine learning. pp. 10347–10357. PMLR (2021)

[10] Tsoumakas, G., Katakis, I.: Multi-label classification: An overview. International Journal of Data Warehousing and Mining (IJDWM) **3**(3), 1–13 (2007)

[11] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al.: Huggingface's transformers: State-of-the-art natural language processing. arXiv preprint arXiv:1910.03771 (2019)

[12] Yang, P., Abusafia, A., Lakhdari, A., Bouguettaya, A.: Energy loss prediction in iot energy services. In: Proceedings of the IEEE international conference on web services. IEEE (2023)

[13] Yang, P., Abusafia, A., Lakhdari, A., Bouguettaya, A.: Monitoring efficiency of iot wireless charging. In: Proceedings of the IEEE international conference on pervasive computing and communications. IEEE (2023)

[14] Yang, P., Teng, C., Mangos, J.G.: Establishment of neural networks robust to label noise. arXiv preprint arXiv:2211.15279 (2022)

6 Appendix

6.1 Code and Model

Folder link:

`https://drive.google.com/drive/folders/14zh7qWebkhUnwaodWgrvQz_C_IukygO_?usp=sharing`

Code File:

There are four .ipyb files in the folder. **Main Code.ipyb** is the main model code for the report. The other three Appendix code files is used for comparison analysis purpose only.

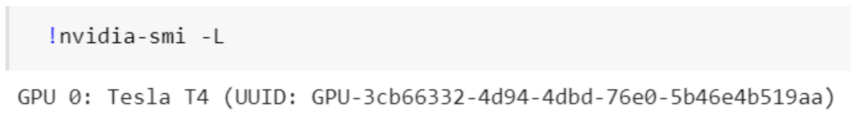
The code can be run in Google Colab. The input and output directory path should be updated to user's own directory. More detailed instructions can be found in the code.

Model File: finetunemodel.pt The final model size is 86M as demonstrated in image below.

finetunemodel.pt 9/05/2023 4:05 PM PT File 86,061 KB

6.2 Hardware and Software:

- Python version: Python 3.9.16
- Packages used:
  os 3.10.11
  re 2.2.1
  csv 1.0
  timm 0.9.2
  string 3.10.11
  pandas 1.5.3
  numpy 1.22.4
  statistics 3.10.11
  itertools 3.10.11
  PIL 8.4.0
  time 3.10.11
  matplotlib 3.7.1
  torch 2.0.1+cu118
  torchvision 0.15.2+cu118
  scikit-learn 1.2.2
  transformers 4.29.2
- Google Colab is used to create the model and details are shown below:



```
!nvidia-smi -L
```

```
GPU 0: Tesla T4 (UUID: GPU-3cb66332-4d94-4dbd-76e0-5b46e4b519aa)
```