A Comparison of Model Architectures for Missing Pixel Predictions

Pittawat Taveekitworachai
Graduate School of Information Science and Engineering
Ritsumeikan University
Shiga, Japan
gr0609fv@ed.ritsumei.ac.jp

***Abstract*—This study compares and analyzes model architectures for detecting missing pixels in images. The effectiveness of three models—the multi-layer perceptron (MLP), convolutional neural network (CNN), and MobileNetV2—is assessed, with separate training on two datasets. The first dataset consists of curated images provided by an instructor, while the second dataset is a 1K sampled subset of Tiny-ImageNet-2000. To evaluate the scalability of the MLP, a larger architecture is trained on the 1K-Tiny-ImageNet-2000 dataset. Additionally, the original MLP is trained on the entire Tiny-ImageNet-2000. Moreover, an open-sourced Stable Diffusion (SD) model, tailored for inpainting, is also investigated for a comprehensive comparison. The performance metric used is mean squared error. The results reveal that models developed using the larger dataset perform worse than those trained on the smaller dataset. In both datasets, the CNN and MobileNetV2, which are CNN-based architectures, consistently outperform the MLP, showcasing their efficiency in extracting useful features through convolutional layers. Unexpectedly, scaling the MLP to a bigger architecture does not enhance its performance. Contrary to general observations, where it excels in general inpainting tasks, the SD model exhibits inferior performance in this specific task. To improve performance for this task, it is recommended to consider diversifying datasets, exploring novel model architectures, and incorporating data augmentation and transfer learning approaches.**

***Index Terms*—missing pixels, cnn, mlp, inpainting, stable diffusion**

I. Introduction

Missing pixels prediction task, also known as image inpainting, is one of the computer vision tasks [1]. The main objective of this task is to predict missing, corrupted, or target pixels in an image. By doing so, a visually plausible content that blends with the surrounding context is generated. This task is crucial in various image processing tasks and applications, such as image restoration, object removal, and image synthesis [2]. Deep learning (DL) models have been integrated for this task in recent years [3].

Convolutional Neural Networks (CNNs) are a foundational architecture for numerous tasks in computer vision due to their ability to extract hierarchical and spatial features from images [4]. This property of convolutional filters makes them an ideal candidate for the tasks, allowing for the better capture of local and global features of the input image. Moreover, CNN-based architectures have evolved beyond simple CNNs, and more complex models have been introduced over the years. Among these, MobileNetV2 [5] was introduced by Sandler et al. back in 2018. This architecture has gained attention for its ability to perform efficiently in low-resource environments. Therefore, it is one of the models I chose for evaluating performance in this task.

Along with MobileNetV2, simple CNN, and multi-layer perceptron (MLP) architectures were developed to assess performance and compare them for a better perspective. These three models are trained on two datasets separately to determine how well they can handle different input sizes and variety. The first dataset consists of selected images provided by an instructor, allowing for a controlled examination of the models' performance. The second dataset is a 1K sampled subset of Tiny-ImageNet-2000, which is part of the larger ImageNet dataset [6]. This subset presents a more challenging real-world scenario with a wide variety of complex images.

In addition to comparing different model architectures, I am also investigating the scalability of the MLP by training a larger version of the model on the 1K-Tiny-ImageNet-2000 dataset. This will allow me to observe if a larger architecture helps the model to be properly fitted with an increasing amount of data. Additionally, I am training the original MLP on the full Tiny-ImageNet-2000 dataset to determine if increasing the amount of data leads to improved performance. This analysis provides valuable insights into the effects of model and dataset sizes for this task.

The diffusion model, such as Stable Diffusion (SD) [7], has recently gained popularity in various computer vision applications, including text-to-image generation [7], image-to-image translation [8], and super-resolution [7]. Therefore, I have included an open-sourced SD model, specifically tuned for general image inpainting tasks[1], in this evaluation. This presents a more comprehensive view of evaluation against the current state of the field. The mean squared error (MSE) metric is selected as a performance metric for this task due to its simplicity.

This study aims to offer insights into the comparative performance of various model architectures for missing pixels prediction tasks. The findings can serve as a guide for researchers and practitioners in selecting the most suitable model for this task, facilitating informed decisions for designing an architecture for the task. For the rest of the paper, I provide

[1]https://huggingface.co/runwayml/stable-diffusion-inpainting

a detailed description of the methods in Section II, including model architectures and datasets. Section III presents experimental results, followed by a discussion and an analysis of the findings. Finally, I conclude by summarizing the contributions of this research and outlining potential improvements for this task in Section IV.

II. Methods

This section provides details of the model architectures, dataset, and training and evaluation procedures. First, the details of the model architecture are discussed in Section II-A. Next, additional architectures explored are explained in Section II-B. Finally, the details of the dataset and the training and evaluation process are discussed in Sections II-C and II-D, respectively.

A. Model Architecture

MLP is a traditional neural network architecture consisting of multiple layers of fully interconnected nodes, i.e., hidden layers. For an architecture designed for this task, the input layer has 120 units with a rectified linear unit (ReLU) activation function, expecting an input shape of `(60)`, which allows the network to model complex non-linear relationships in the data efficiently. One hidden layer is utilized, containing 60 units with ReLU activation functions, to allow the model to learn the features. The output layer has four nodes with a Sigmoid activation function, as it is a regression task. I noted that this architecture choice was selected and provided by the instructor as a baseline model.

CNN is a DL architecture known for its effectiveness in computer vision tasks due to its ability to learn and extract useful features. The CNN used in this study consists of two convolutional layers, each followed by a max-pooling layer for downsampling. The first convolutional layer serves as an input layer, expecting an input image of shape `(8, 8, 1)` where the middle four pixels are removed. In this layer, 32 filters of size `(5, 5)` are utilized because of the limited size of the input. The ReLU activation function is used to introduce non-linearity, just like in MLP. The option of `same` padding is chosen to retain the spatial dimensions of the input.

The next convolutional layer has 64 filters of size `(3, 3)` with the same ReLU activation function and default padding. The final max-pooling layer has the same settings as the previous one. After the convolutional layers, the outputs are flattened to serve as inputs for an additional fully connected layer with 64 units and a ReLU activation function. The output layer is the same as the MLP, containing four nodes with a Sigmoid activation function.

MobileNetV2 is a lightweight CNN architecture designed for low-resource environments. This architecture is chosen because it offers several advantages, including faster training due to the low number of parameters and operations. It is also well-tested and optimized for various vision tasks, making it a reliable choice for the study. It employs depth-wise separable convolutions to achieve this reduction in computational complexity.

For this study, I use the MobileNetV2 architecture with its default configuration, except for the last layer of the original output. The output layer now contains four nodes using a Sigmoid activation function to match the other architectures. The implementation follows an open-source implementation[2].

B. Additional Exploration

In addition to training the three aforementioned models on both the original dataset and the 1K-Tiny-ImageNet-2000 dataset, I also investigated the scalability of the MLP model. Specifically, I scaled the MLP to a larger architecture called MLP-L with the following configuration: 256 units in the input layer and three additional hidden layers with 128, 128, and 64 units, respectively, all using ReLU activation functions. The output layer retains the same configuration as the MLP. I then proceeded to train this larger MLP-L model on the 1K-Tiny-ImageNet-2000 dataset. In addition, I explored the performance of the original MLP model when trained on the full Tiny-ImageNet-2000 dataset, without the 1K sample, aiming to observe any potential improvements with increased training data. Furthermore, I evaluated a pre-trained open-sourced SD model tuned for inpainting as a comparative baseline.

C. Datasets

Original Dataset is provided by an instructor. It consists of a diverse set of grayscale images, totaling 55 samples. Out of these, one image, `balloon.bmp`, is designated as the testing target for evaluation across all models. To ensure consistency, all images are resized to `(256, 256)` before being used during training and evaluation.

1K-Tiny-ImageNet-2000: Tiny-ImageNet-2000 is a subset of the ImageNet dataset originally designed as part of the Stanford CS231N class. For this study, I created a subset of 1,000 images by randomly selecting from training, validation, and testing images of the Tiny-ImageNet-2000 dataset. This was done to minimize the computation requirements and make training more manageable. To standardize the input data for the models, all images were pre-processed into grayscale before training. Additionally, it is worth noting that all images from the original Tiny-ImageNet-2000 dataset, across all sets, were also utilized in one of the experiments. Each image in this Tiny-ImageNet-200 dataset has a shape of `(64, 64)`. This dataset was selected because ImageNet is a well-established dataset used for many standard evaluations [9].

D. Training and Evaluation

Each model architecture is trained separately on each dataset to compare their performance. The training process is performed using the Adam optimizer [10] with a fixed learning rate. The models are trained for 450 epochs with a batch size of 512 for MLP and 256 for the rest. MSE is selected as the loss function during training, and we assign 10% of the training dataset for validation during training. All inputs are chosen to have a patch size of `(8, 8)`, where the four middle pixels

[2]https://github.com/Haikoitoh/paper-implementation

are removed (i.e., set to zero) or not supplied as a part of the input for MLP models. The training process also implements a function to save only the best model, i.e., the model with the lowest loss, and all evaluation processes are done using the saved version of the model.

For evaluation, the models are asked to predict the four missing pixels while all other pixels are automatically filled using the original image. The results from the model predictions are saved as an image file and later loaded to compare with the original image. The value of each predicted pixel is compared with the corresponding original pixel, and the sum of these differences is the final metric used for comparison. The chosen evaluation image is a file named `balloon.bmp`, which is part of the original dataset but was exempted from the training. This provides a fair comparison and ensures no data leakage during the training process. For the SD model, there is no additional training or fine-tuning performed. The evaluation process provides the model with the entire image as input while masking all target pixels that are supposed to be missing. These masked pixels serve as targets for the model to predict.

III. Results and Discussions

In this section, I present the results of the evaluation conducted on five model architectures for missing pixel predictions: MLP, CNN, MobileNetV2, MLP-L, and SD. The performance evaluation of each model is provided in Section III-A. Following the presentation of results, a discussion of the findings and their implications is outlined in Section III-B. This analysis aims to investigate some reasons behind the obtained results, allowing for a deeper understanding of their performance in the context of missing pixel predictions.

A. Model Performance

I assessed every model on the same target image, as described in the previous section. The results are MSE obtained from the predictions of each model. The lower the MSE, the better the result, indicating that it is closer to the ground truth image. Table I summarizes the results obtained from evaluating each model architecture on different training datasets, and Fig. 1 shows the original images and images obtained from model predictions. The results showed the superiority of CNN-based architectures over MLP architectures, while also indicating that scaling up (either architecture or data) worsens the performance for this task. Additionally, SD obtained the worst results across all of the models, as can be observed visually.

TABLE I: Model performance on missing pixels predictions using various architectures on different datasets.

Model	**Training Dataset**	**MSE↓**
MLP	Original	22.09
CNN	Original	**13.61**
MobileNetV2	Original	15.65
MLP	1K-TinyImageNet-2000	23.61
CNN	1K-TinyImageNet-2000	**13.90**
MobileNetV2	1K-TinyImageNet-2000	17.04
MLP-L	1K-TinyImageNet-2000	64.22
MLP	TinyImageNet-2000	446.78
SD	N/A	312.80

B. Discussions

The findings highlighted interesting observations on the performance of each model that may not depend on the size of the model but rather on the base architecture that the model uses to learn patterns of data for predictions. The comparative performance was discussed in Section III-B1. Next, the scaling of MLP on architecture size and training data was analyzed in Section III-B2. Finally, I provided some possible reasons for the poor performance of SD and suggested ways, in general, to investigate further to improve the models in Section III-B3.

1) Comparative Performance of Model Architectures: The evaluation results demonstrated that CNN and MobileNetV2, which are CNN-based architectures, consistently outperformed the traditional MLP, regardless of the training datasets. This outcome showcased the effectiveness of CNN-based architectures in extracting spatial features through convolution layers, which was particularly beneficial for this task. Surprisingly, the MobileNetV2 architecture, known for its efficiency and performance in resource-constrained environments, did not outperform a simple CNN architecture in this context. This could be due to the fact that the data was not sufficient for the model to learn the patterns, both in terms of the number of samples and the size of the input. It is worth noting that the original MobileNetV2 is generally designed to work with images larger than `(32, 32)`, which is four times the size of our current inputs.

As for the traditional MLP architecture, it struggled to match the performance of CNN-based architectures, highlighting its limitations in handling spatial relationships effectively. Despite having a single hidden layer and being capable of modeling complex non-linear relationships in data, the MLP's performance lagged behind the CNNs. This suggested that for the missing pixels task, where spatial information was crucial, CNN-based architectures are more suitable.

It could also be observed that more training data, such as using the 1K-TinyImageNet-2000 as a training dataset, did not increase performance and, in fact, worsened the performance of all models, although they did not suffer much decrease in MSE. However, this was a surprising result, as when we compared CNN and MobileNetV2, which is a much larger CNN-based architecture, we did not observe an increase in the performance of MobileNetV2 trained on a larger dataset.

The usual understanding is that with a larger architecture and a larger training dataset, the performance should increase in general. However, this was not the case, and it might be due to the fact that the input size with only `(8, 8)` image patches might not provide sufficient information for the model to learn the patterns. On the other hand, it might be more beneficial to utilize a smaller architecture, which results in fewer computational requirements while outperforming in this

(a) Original image

(b) A result image from the MLP model trained on the original dataset.

(c) A result image from the CNN model trained on the original dataset.

(d) A result image from the MobileNetV2 model trained on the original dataset.

(e) Original image

(f) A result image from the MLP model trained on the 1K-Tiny-ImageNet-2000 dataset.

(g) A result image from the CNN model trained on the 1K-Tiny-ImageNet-2000 dataset.

(h) A result image from the MobileNetV2 model trained on the 1K-Tiny-ImageNet-2000 dataset.

(i) Original image

(j) A result image from the MLP-L model trained on the 1K-Tiny-ImageNet-2000 dataset.

(k) A result image from the MLP model trained on full Tiny-ImageNet-2000 dataset.

(l) A result image from the Stable Diffusion Inpainting model.

Fig. 1: Comparison of prediction results from various models. *Top row*: Models trained on the original dataset. *Middle row*: Models trained on the 1K sampled subset of Tiny-ImageNet-2000. *Bottom row*: Results from extra models.

specific task.

2) Scalability of MLP: To investigate whether the trend of using larger datasets together with larger architectures results in better performance holds true, we developed MLP-L for comparison with the standard MLP in this task, similar to how CNN was compared with MobileNetV2. Interestingly, the analysis of MLP's scalability revealed surprising results. When the MLP was scaled up to a larger architecture, MLP-L, and trained on the 1K-Tiny-ImageNet-2000 dataset, its performance significantly degraded. The larger MLP-L achieved a higher MSE compared to the original MLP, indicating that increasing the model's number of trainable parameters did not lead to improved performance in this context.

Furthermore, training the original MLP on the full Tiny-ImageNet-2000 dataset also resulted in poor performance, with a significantly higher MSE than the MLP trained on the smaller original dataset. This suggests that increasing the training data does not necessarily lead to better results for the traditional MLP architecture in this task, especially when the size of the architecture does not scale up to match the size of the dataset.

The observations regarding the scalability of MLP raise

questions about the model's ability to effectively capture complex spatial patterns and relationships. It indicates that MLP's limitations may become more prominent when dealing with larger datasets and higher-dimensional data, making it less suitable for image inpainting tasks compared to CNN-based architectures.

3) Performance of SD Model: This study also included an evaluation of the SD model specifically tuned for inpainting tasks. The SD model showed the worst performance compared to the other architectures. This was unexpected, as SD has demonstrated impressive results in general image inpainting tasks and other applications. This may indicate that its architecture or training procedure is not well-suited for the nature of this task, with very limited inputs. It could also suggest that there might be a domain shift between the data used for tuning the SD model and the testing target used in this study, leading to poor performance. Further investigation and experimentation would be necessary to understand the specific reasons behind the SD model's poor performance in this scenario.

IV. Conclusions and Future Work

In this study, I conducted a comparative analysis of model architectures for predicting missing pixels in images. I evaluated three distinct baseline architectures: MLP, CNN, and MobileNetV2, trained using two datasets: an instructor-provided dataset and a 1K-Tiny-ImageNet-2000. Additionally, I explored the scalability of the MLP by training a larger architecture, MLP-L, on the 1K-Tiny-ImageNet-2000 dataset and evaluated the original MLP on the full Tiny-ImageNet-2000. Furthermore, I evaluated the SD model to provide a comprehensive comparison.

The results showed that CNN-based architectures, including CNN and MobileNetV2, consistently outperformed the traditional MLP, regardless of training datasets. Surprisingly, the MLP-L, a scaled-up version of MLP, exhibited degraded performance compared to the original MLP, indicating that increasing the model's size did not lead to improved results in this scenario. Moreover, training the original MLP on the full Tiny-ImageNet-2000 dataset also resulted in poor performance, suggesting that simply increasing the training data may not yield better results. The performance of the SD model was unexpectedly poor in this specific context, indicating potential challenges in its architecture or training process for handling limited inputs effectively.

For future work, several directions can be explored to improve the performance of models. First, diversifying the datasets used for training and evaluation may help gain insights into the generalizability and robustness of the models across various scenarios and lead to better performance for the task. Second, investigating novel model architectures, for example, Vision Transformers [11], could lead to more effective inpainting. These approaches also have the potential for leveraging pre-trained models. Third, utilizing larger input sizes could potentially enhance the model's performance. Moreover, it is essential to study the domain shift issue between the SD model's tuning data and the target dataset to understand the reasons behind its underperformance.

References

[1] Andreas Lugmayr, Martin Danelljan, Andres Romero, et al. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11461–11471, June 2022.

[2] Xiaobo Zhang, Donghai Zhai, Tianrui Li, et al. Image inpainting based on deep learning: A review. *Information Fusion*, 90:74–94, 2023.

[3] Hanyu Xiang, Qin Zou, Muhammad Ali Nawaz, Xianfeng Huang, Fan Zhang, and Hongkai Yu. Deep learning for image inpainting: A survey. *Pattern Recognition*, 134:109046, 2023.

[4] Jaya Gupta, Sunil Pathak, and Gireesh Kumar. Deep learning (cnn) and transfer learning: A review. *Journal of Physics: Conference Series*, 2273(1):012029, may 2022.

[5] Mark Sandler, Andrew Howard, Menglong Zhu, et al. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[6] Jia Deng, Wei Dong, Richard Socher, et al. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[7] Robin Rombach, Andreas Blattmann, Dominik Lorenz, et al. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, June 2022.

[8] Chitwan Saharia, William Chan, Huiwen Chang, et al. Palette: Image-to-image diffusion models. In *ACM SIGGRAPH 2022 Conference Proceedings*, SIGGRAPH '22, New York, NY, USA, 2022. Association for Computing Machinery.

[9] Olga Russakovsky, Jia Deng, Hao Su, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, Dec 2015.

[10] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, et al. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

Appendix

Datasets, trained models, and code are available at https://bit.ly/ru-kbs-2023.