Quantitative Analysis of Layer Relevance in Convolutional Neural Networks Using the Example of Artist and Counterfeit Detection

Wolfgang Reuter
Deggendorf Institute of Technology
Applied Research in Engineering Sciences (Master)
Dieter-Görlitz-Platz 1, 94469 Deggendorf
Email: wolfgang.reuter@stud.th-deg.de

***Abstract*—Artificial Intelligence (AI) is increasingly used in the authentication of artworks. Most current approaches rely on Convolutional Neural Networks (CNNs), which learn to recognize stylistic patterns specific to the artists represented in the training data. In the early layers of a CNN, simple features such as contrast and color variations are extracted, while deeper layers capture more complex patterns. However, early-layer style characteristics tend to be diluted as information propagates through the network. In this study, we dissect a standard ResNet50 architecture into three distinct block models, each trained separately. A full ResNet50 model, referred to as the benchmark, is also trained for comparison. Results show that certain artists are more accurately attributed using the block models. This method improves painter attribution accuracy from 85.6% to 93%, assuming a known candidate artist—a common scenario in art authentication. In open-set classification, where no artist is presumed, the approach does not significantly outperform the benchmark.**

I. Introduction

The present project builds on an algorithm developed and continuously refined by Art Intelligence GmbH since 2017. This algorithm is used commercially to verify the authenticity of artworks using Artificial Intelligence (AI) and Machine Learning (ML). The underlying model in this two-stage authentication process is a cross-validated ensemble of Convolutional Neural Networks (CNNs), trained on 596,322 patches of 24,429 artworks by 78 artists. Depending on the CNN architecture used, the model achieves a weighted overall accuracy of 85.6% (with ResNet50) or 87.1% (with ResNeXt101). This study examines the relevance of different blocks of layers within the CNNs. For this purpose, the model based on the ResNet50 architecture was segmented. The resulting block models and a full ResNet50, refrerred to as benchmark, were trained and evaluated separately. The aim was to improve the currently used approach of Art Intelligence GmbH.

II. Conceptual Foundation

The architecture of CNNs consists of a sequence of convolutional and pooling layers, with varying configurations. As a result, CNNs learn and recognize different types of patterns at different stages of their architecture. In the early layers, they tend to detect fine-grained structures such as edges at specific angles or basic color contrasts. In deeper layers, the networks are capable of identifying more complex features such as faces or specific shapes, which lead to stronger activations [1], [2], [3]. Examples are shown in Figure 1.

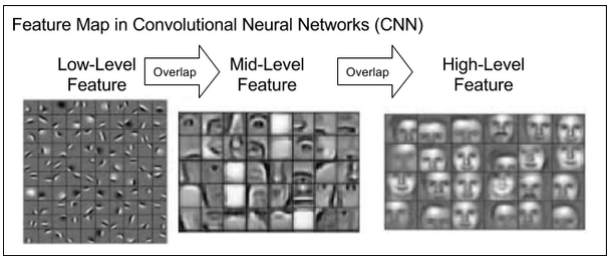

Figure 1: Visualization of filter outputs in a CNN. On the left: fine-grained structures detected in early layers; on the right: complex patterns and shapes that produce strong activations in deeper layers[4].

The hypothesis of this project is that the recognition of painting styles by individual artists is particularly achieved through fine-grained structures, while for other artists, more complex patterns enable better discrimination between different painting styles. Preliminary qualitative investigations on this topic have already been conducted [5]. The hypothesis that led to the setup of this project is that, in a CNN, learning success in the earlier layers is largely relativized (or blurred) when the entire network is trained from the input layer to the output layer. Artists who are more likely to be recognized based on fine-grained structures could be at a disadvantage and generally perform worse – which, if the hypothesis is correct, would naturally impact the overall performance of the model or ensemble.

The considerations presented here led to the project setup described in the following.

III. Project Structure

To quantitatively assess the relevance of different layers, the CNNs—based on the ResNet50 architecture in this case—were segmented, i.e., divided or truncated. The procedure is illustrated graphically in Figure 2.

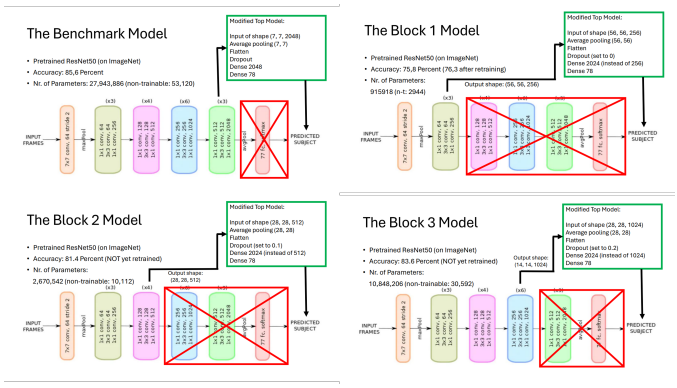

Figure 2: Schematic representation of the architecture of the CNNs used in this analysis. A global pooling layer after the last used block, as indicated in the green boxes, ensures that all models have the same output size ($2048 \times 7 \times 7$ pixels) before their output is further processed in the respective top models.

The model trained from the input to the output layer essentially corresponds to the classical ResNet50 architecture [6] and is hereafter referred to as the benchmark model. It is the model type currently used by Art Intelligence GmbH. Only after the last pooling layer has the architecture been slightly modified. The nomenclature of the other models is based on the respective block after which the CNNs were truncated. These are referred to as the Block-1, Block-2, and Block-3 models.

With one exception, all models were trained using the same configuration of hyperparameters, as shown in Table I. Only the dropout rate was varied between models. The reason for this is that the model previously trained by Art Intelligence uses a dropout rate of 0.3, which was determined empirically. However, the Block-1 through Block-3 models have (in some cases significantly) lower complexity, as indicated by the number of parameters, which can also be seen in Figure 2.

For this reason, the dropout rate was set to 0 for the Block-1 model, which has only about 1/30th of the parameters of the benchmark model, and then gradually increased in steps of 0.1 up to 0.3 in the benchmark model.

Hyperparameter	**Value**
Warm-up epochs	3
Total epochs	80
Initial learning rate	0.0005
Learning rate reduced after epoch	15
Reduction factor	0.95
Minimum learning rate	0.000007
Dropout rate	0.0, 0.1, 0.2, 0.3 (Block-1 to Benchmark)
Data augmentation	Vertical flip, horizontal flip, rotations (90°, 180°, 270°), color channel permutation

Table I. Training configuration (hyperparameters) used for the different model types.

All block models and the benchmark model were trained using cross-validation. The dataset was initially split into training, validation, and test sets in a 70:15:15 ratio. Afterwards, the training and validation sets were merged and then resplit for cross-validation using an 80:20 ratio. As a result, each model's validation set contains different data, while the training sets differ by 20% of the data. All results presented in this work are based exclusively on an evaluation of the artworks in the test set, which was not exposed to the models during training. The dataset split is illustrated in Figure 3. During training, the model with the best weighted accuracy across all painters was saved. The evaluation is based on this model, which may not necessarily have been trained for the full 80 epochs.

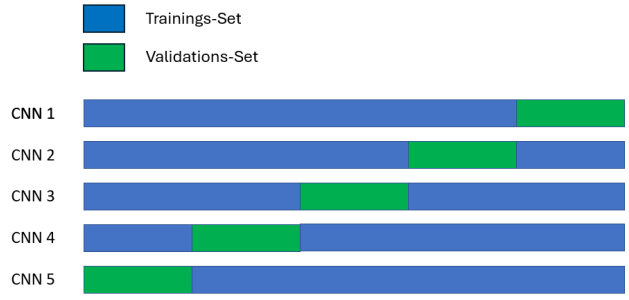

Figure 3: Schematic illustration of the cross-validated training and validation splits. The test set is separate from both and was not used during training.

IV. Results

The results are first of all analyzed for each individual model—both in terms of overall accuracy and the per-artist accuracy. Subsequently, the potential for improvement through the use of the block models is examined across the following levels of granularity:

1) Artist level
2) Image level
3) Crop level

A. *Overall Perfomance*

The weighted overall accuracy has already been shown in Figure 2 for each block model and the benchmark model. However, it is also listed here again in tabular form for various evaluation metrics.

Table II shows that all block models perform worse than the benchmark model, which was expected. However, in light of the considerations outlined above and the goals of the project, a more detailed examination is warranted. It becomes evident that each block model performs better—sometimes significantly better—for certain artists than the benchmark model.

Model Type	Accuracy	Precision	F1-Score
Benchmark	85.7	85.5	85.3
Block-1	76.6	75.8	75.8
Block-2	81.5	81.4	81.2
Block-3	83.5	83.6	83.2

Table II: Weighted average for accuracy, precision, and F1-score for the benchmark model and the three block models.

B. Painter Specific Results

All painters that are recognized better in at least one of the block models are listed in Table III. Values are averaged across all cross-validation models.

Model	Painter	Difference	Class Size (%)
Block-1	Carel Fabritius	0.333	0.001
	Edvard Munch	0.034	0.008
	Wolfgang Beltracchi	0.056	0.010
	Heinrich Campendonk	0.021	0.013
Block-2	Carel Fabritius	0.333	0.001
	Govert Flinck	0.042	0.006
	Edward Hopper	0.037	0.007
	Andy Warhol	0.069	0.008
	Wolfgang Beltracchi	0.028	0.010
	Georges Braque	0.028	0.010
	Heinrich Campendonk	0.021	0.013
	A. R. Penck	0.017	0.016
	Edouard Manet	0.032	0.017
	Max Ernst	0.029	0.019
Block-3	Govert Flinck	0.042	0.006
	Edward Hopper	0.037	0.007
	Andy Warhol	0.069	0.008
	Eugene Delacroix	0.057	0.009
	Georges Braque	0.028	0.010
	Heinrich Campendonk	0.042	0.013
	Henri Matisse	0.014	0.020
	Pablo Picasso	0.013	0.021

Table III: Improvement in recall for painters better recognized by the respective block models compared to the benchmark model.

These results are also visualized in Figure 4, which reveals a slight inverse relationship between class size and the degree to which the block models outperform the benchmark model.

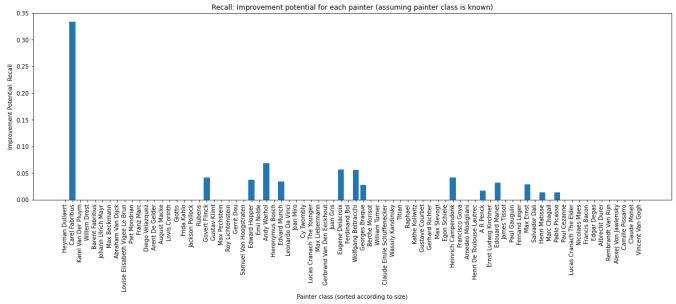

Figure 4: Accuracy improvements for painters better recognized by at least one block model over the benchmark model.

C. Improvement Potential Through Block Model Training

The following discussion assumes that inference does not involve a general classification task. This is because the estimation of the improvement potential compared to the benchmark model presupposes that the class of a test image is known—or that an image is being tested specifically for one class. In forgery detection, unlike in standard image recognition applications, this assumption holds true in most cases. Applied to painter recognition, this method cannot be used to determine which of Rembrandt's students might be the author of a given work. It can only be used to determine whether a particular artist is the author—or not. Additional use cases arise, for example, in the medical field, such as when a chest X-ray is evaluated for a specific diagnosis using block models. For the estimation of this potential, all 92,475 images were first tested on all models. The approach used during training—selecting as many patches from each image as needed to ensure that all painters are represented by the same number of patches—was retained for this evaluation. The estimation was then performed on three levels.

1) Painter Level: In this analysis, all images of a painter were evaluated using the model whose so-called confidence for that painter—averaged across all images and patches—was the highest. As a confidence metric, categorical crossentropy was used, i.e., the same metric applied in the loss function during training. This is a so-called voting approach, meaning that the predictions of the remaining models were not considered; they were neither averaged nor weighted in the final decision.

2) Image Level: In a second step, this procedure was applied at the level of entire images. That is, all patches extracted from a test image were evaluated using the model for which the confidence—averaged across all patches of that image—was highest for the actual artist.

3) Crop Level: Finally, the procedure was also applied at the level of individual patches, where each patch was classified using the model that showed the highest confidence for the actual artist with respect to that specific patch. This approach yielded the best results among all levels, as shown in Table IV.

Metric	Benchmark Model	Best Model per Artist	Best Model per Image	Best Model per Patch
Recall, w	85.6	87.8	89.9	93.0
Recall, u	78.0	80.6	83.7	86.9
Precision, w	85.7	87.8	90.0	93.0
Precision, u	79.2	81.9	85.0	86.9
F1-Score, w	85.3	87.6	89.8	92.8
F1-Score, u	77.9	80.8	83.8	87.7

Table IV: Overall results, weighted (w) and unweighted (u), across different aggregations (artist, image, and patch), if the model with the highest confidence for the true class is selected.

The results of this procedure improve with the granularity of the aggregation. At the artist level, the results are consistently about two percentage points better than those of the benchmark model, at the image level about 4-5 percentage points, and at the crop level between 7 and 10 percentage points, depending

on the metric. If the value of 100 percent minus the respective metric of the corresponding model is considered as the theoretical potential, this is roughly half-exploited by the procedure, as can be seen in Figure 5. It shows that the accuracy increases for almost all artists when the correct model is used for each crop.

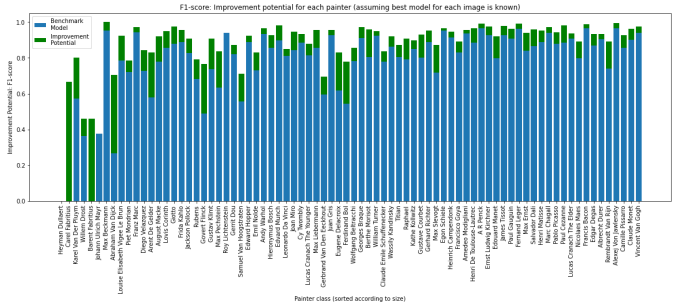

Figure 5: Accuracy improvements for painters better recognized by at least one block model over the benchmark model.

V. Interpretation of Results

The performance improvements achieved by splitting a CNN into block models are impressive, always assuming that the respective class—here, the presumed artist—is known when selecting the appropriate block model. It is therefore worthwhile to consider the results in a broader context.

The presented method is an ensemble approach. Ensemble methods lead to improved results in the vast majority of cases—see, for example, hundreds of studies on the topic [7]. They are often described as expert systems, drawing comparisons with human experts. For instance, the accuracy of medical diagnoses increases when multiple physicians from the relevant specialties discuss a patient's symptoms—an effect that is well-documented in numerous studies. One such example is provided in [8]. However, what generally applies to the use of different algorithms does not, or only partially, apply to deep convolutional neural networks. While, as discussed earlier in the section *Conceptual Considerations*, different "experts" are also involved here, the situation differs significantly from that of classical ensemble approaches.

The reason is that these "experts"—that is, the different blocks within a convolutional neural network—are connected in series and are therefore not independent. Specifically, the second block in a ResNet50 model does not receive the full data input, but only the already filtered information from the previous expert, namely the output of the first block. The expertise of the second block is subsequently further diluted in the deeper layers.

Although the issue of pre-filtered information in the Block-2 and Block-3 models is not resolved with the approach taken here, splitting the network into block models at least allows the expertise of each individual model to be fully utilized via the ensemble method, without requiring any additional modifications in downstream layers or blocks. However, in order to leverage this improvement in the general case—i.e., when no class is presumed—a dispatcher model is required. This is a model that predicts which of the four models should be used for a given image crop.

To generate the dataset required for such a dispatcher approach, features will be extracted from the image crops. The procedure for this is described in the following chapter.

VI. Feature Generation

To predict which block model should be used for a given image crop, meta-features must first be extracted. This process is guided by three high-level visual characteristics: color distribution, contrast, and flatness. The following subsections describe these properties and the derived features.

A. Color Distribution

To characterize the color distribution of each image crop, nine statistical metrics were computed for each color channel (RGB): minimum, maximum, mean, median, standard deviation, interquartile range (IQR), skewness, kurtosis, and entropy. The same metrics were also extracted from a grayscale version of the crop. In total, 36 features were derived to capture the overall color distribution.

B. Contrast

To capture contrast, each grayscale image crop was processed using Sobel, Scharr, and Laplace filters—yielding first-order (Sobel, Scharr) and second-order (Laplace) gradient maps. For each filter output, gradient magnitude and angle were computed per pixel, both per color channel and across aggregated channels.

As with the color distribution features, standard statistical metrics (e.g., min, max, mean, etc.) were applied to both magnitude and angle images. Additionally, both magnitude and angle values were linearly binned into 18 intervals, and bin counts were used as further features. In total, this resulted in 648 features describing the contrast characteristics of each crop.

C. Flatness (Flächigkeit)

To capture the flatness of an image—interpreted as the spatial distribution of similar color regions—each crop was color-quantized using K-Means clustering with 8, 4, and 2 colors. This reduction simplifies the image, making it amenable to region-based analysis.

For each quantization level, features were extracted from:

- **Cluster sizes:** The pixel counts per cluster, sorted by size, yielding 14 direct features and 24 statistical descriptors (e.g., mean, median, IQR, etc.).
- **Entropy:** Computed from the clustered image rather than from cluster sizes, yielding 3 features.
- **Connected components:** For each color, the number and size of contiguous regions were analyzed, producing 14 count-based features and 112 statistical descriptors across all color levels.

In total, 167 features were extracted to describe flatness, resulting in a final 851-dimensional feature vector when combined with color and contrast features.

VII. Dispatcher Model Training

The goal of training the dispatcher model is to predict, for each image crop, which of the four models (Block-1, Block-2, Block-3, or the Benchmark Model) is expected to perform best. The softmax output from the selected model is then used during final evaluation.

Before training, 25 features were removed because they consistently had a value of zero across all image crops. These features corresponded to the minimum values of the color gradients.

Labels (0, 1, 2, 3) were assigned based on the validation dataset by selecting, for each crop, the model that produced the highest softmax probability for the correct class.

The architecture of the dispatcher model is detailed in Table V. It consists of a fully connected neural network with alternating dense, dropout, and batch normalization layers. The model was designed to handle the reduced feature set of 826 input dimensions and outputs one of four class labels.

Table V: Dispatcher model architecture

Layer	Output Shape	# Parameters
Batch Normalization	(None, 826)	3304
Dense	(None, 826)	683,102
Dropout	(None, 826)	0
Batch Normalization	(None, 826)	3304
Dense	(None, 1652)	1,366,204
Dropout	(None, 1652)	0
Batch Normalization	(None, 1652)	6608
Dense	(None, 1652)	2,730,756
Dropout	(None, 1652)	0
Batch Normalization	(None, 1652)	6608
Dense	(None, 826)	1,366,204
Dropout	(None, 826)	0
Batch Normalization	(None, 826)	3304
Dense	(None, 826)	1,366,204
Dropout	(None, 826)	0
Batch Normalization	(None, 826)	3304
Dense (Output)	(None, 4)	3308

Table VI: Architecture of the Dispatcher Model.

VIII. Dispatcher Ensemble Results

The learning curve from training, as well as the distribution of predicted model types on the test set, is shown in Figure 6. The classification accuracies on the test set across the five cross-validated dispatcher models range from 42.57% to 47.68%. While these results are clearly above random guessing, they offer limited promise for a significant performance improvement in the painter classification task.

The ensemble approach—i.e., predicting the model type using cross-validated dispatcher models—achieves an accuracy of only 85.3%, representing a decrease of 2.3 percentage points compared to the benchmark model. While the dispatcher models do learn meaningful patterns, they fall short of outperforming the benchmark, especially considering that randomly selecting a model type yields a baseline accuracy of 83.2%. The results for each painter are visualized in Figure 7. As shown in Figure 6, the predicted distribution of model types is not uniform. This imbalance makes it particularly

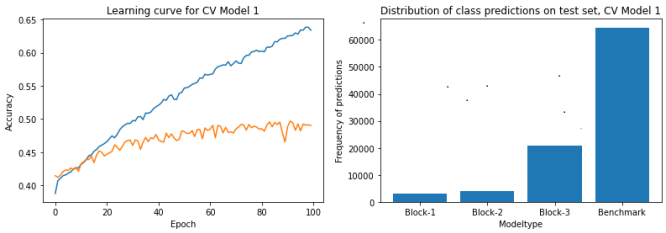

Figure 6: Training and validation result of the dispatcher model (left). Distribution of predictions across the individual blocks and the benchmark model on the test set (right).Both plots refer to the first cross-validated split

challenging for the dispatcher models to accurately predict the smaller classes, i.e., the block-specific models. To mitigate this issue, the dispatcher models were retrained with class weights inversely proportional to the class sizes.

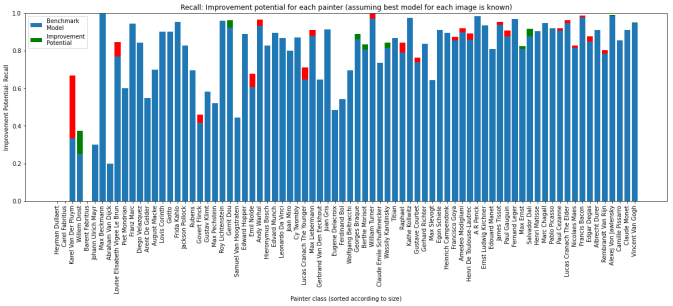

Figure 7: Dispatcher ensemble in painter classification. The visualization is based on model-type predictions at the image patch level; painters are sorted by the number of images in the dataset. Green bars indicate improvements over the benchmark model, red bars indicate performance drops.

However, this approach did not lead to any improvement in the overall accuracy on the test set. The weighted ensemble accuracy reached only 85.2%. Even stronger emphasis on the underrepresented classes or model types—by adjusting class weights—did not improve the accuracy or other metrics such as precision and F1-score. When class weights were squared, the weighted accuracy dropped slightly to 85.1%, and with weights raised to the power of three, it returned to 85.2%.

As an additional approach, a rule-based selection of model types was implemented. Here, the dispatcher model was only used if its Softmax output for a specific model type exceeded 0.85. In other words, the painter prediction of the model type suggested by the dispatcher was only adopted if the dispatcher was highly confident in its prediction. Otherwise, the prediction from the benchmark model was used. This restriction yields a minor, though negligible, improvement: the weighted accuracy across all painters increases from 85.0% to 85.8%. However, under this condition, the dispatcher model is only applied to 8.46% of all image patches.

IX. Summary

This project has demonstrated that segmenting CNNs into separate block models holds potential for improving classification performance. In the case at hand—the application of these models for detecting painting styles—this improvement is evident across all levels of aggregation:

1) When the best model is selected for each patch of each image, averaged over all patches and images of the corresponding painter,
2) When the best model is selected for each patch, averaged over all patches of the corresponding image,
3) When the best model is selected for each patch based on the corresponding painter.

However, the gains vary substantially depending on the granularity of aggregation: the finer the granularity, the greater the improvement compared to the benchmark model. On the basis of patch-level aggregation, the weighted overall accuracy increases from 85.7% to 92.8%.

While this does not refute the hypothesis that certain painters can be recognized primarily by fine-grained features, it significantly qualifies it. According to the current analysis, the decisive factor in choosing the right or best model appears to be characteristics inherent in the specific input image patches—rather than features associated predominantly with a painter's style.

The unresolved problem remains the selection of the correct block or benchmark model. An improvement in the overall result only occurs when a specific painter (or, more generally, a class) is hypothesized in advance—and then, based on this assumption, the best model for each patch is chosen. The actual classification then proceeds by ignoring this assumption and simply selecting the painter or class with the highest Softmax output.

As long as it is not possible to accurately predict or otherwise determine the correct or best model type for a given input image—without prior knowledge of the correct class—this approach remains of limited use. It can only be applied in scenarios where the input image is evaluated for membership to a specific, hypothesized class or painter. General classification without such prior assumptions is currently not feasible.

X. Future Work

The goal of this project is to enhance the dispatcher model to significantly improve painter classification accuracy. While current approaches are promising, they have not yet yielded fully satisfactory results. The following enhancements are proposed:

1. Extended Low-Level Feature Extraction

Further extraction of low-level features. Possible directions include:

(a) Categorizing color channel values into intervals.
(b) Extracting cross-channel "color contrasts" using techniques such as Sobel filters.
(c) Estimating pseudo-magnitude and angle values by interpreting the difference to the previous color channel as the x-, and to the next channel as the y-component.

2. Feature Normalization

So far, only min-max scaling was used to maintain compatibility with statistical tests. However, due to large value variations, other normalization strategies may improve classification:

(a) Log-based normalization for features with maximum values over 1000.
(b) Min-max scaling for values in the range of 10 to 1000.
(c) Gaussian normalization for smaller values.

4. Dispatcher Model Architecture

The dispatcher model architecture has not yet undergone systematic optimization. Potential improvements include:

(a) Adjusting the number of layers,
(c) Testing different loss functions, such as binary crossentropy in combination with sigmoid output, following a multi-class, multi-label paradigm,
(d) Developing a regression-based dispatcher model that predicts Softmax scores for each block model instead of performing classification,

5. Training on Raw Image Patches

Training the dispatcher model as CNN directly on raw image patches, while previously avoided due to limited interpretability, may yield improved performance and should be reconsidered.

These suggestions are not exhaustive. However, they should be systematically prioritized based on initial experimental results.

References

[1] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. European Conf. on Computer Vision (ECCV)*, Zurich, Switzerland, 2014, pp. 818–833.

[2] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, "Understanding neural networks through deep visualization," in *Proc. Deep Learning Workshop, Int. Conf. on Machine Learning (ICML)*, 2015.

[3] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K.-R. Müller, "Evaluating the visualization of what a deep neural network has learned," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 11, pp. 2660–2673, Nov. 2017, doi: 10.1109/TNNLS.2016.2599820.

[4] F. Borchert, "Feature extraction from event logs for predictive monitoring of business processes," Master's thesis, 2017, doi: 10.31237/osf.io/y6px4.

[5] S. J. Frank and A. M. Frank, "Analysis of Dutch Master Paintings with Convolutional Neural Networks," in *arXiv*, 2020. [Online]. Available: https://arxiv.org/abs/2002.05107.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Microsoft Research*, 2015. [Online]. Available: https://arxiv.org/pdf/1512.03385.pdf

[7] A. Stanescu and D. Caragea, "An empirical study of ensemble-based semi-supervised learning approaches for imbalanced splice site datasets," *BMC Systems Biology*, vol. 9, suppl. 5, p. S1, 2015. [Online]. Available: https://doi.org/10.1186/1752-0509-9-S5-S1

[8] E. C. Khoong, S. S. Nouri, D. S. Tuot, S. Nundy, V. Fontil, and U. Sarkar, "Comparison of diagnostic recommendations from individual physicians versus the collective intelligence of multiple physicians in ambulatory cases referred for specialist consultation," *Medical Decision Making*, vol. 42, no. 3, pp. 293–302, Apr. 2022. [Online]. Available: https://doi.org/10.1177/0272989X211031209