Multipurpose Image Colorization: A Novel Pipeline Using Convolutional Neural Networks

Ivannia Gomez Moreno[a,b], Ulises Orozco-Rosas[a,*], Kenia Picos[a], and Tajana Rosing[b]

[a]CETYS Universidad, Ave. CETYS Universidad No. 4, Fracc. El Lago, C.P. 22210, Tijuana, Baja California, Mexico

[b]University of California San Diego, 9500 Gilman Dr, La Jolla, CA 92093, United States

ABSTRACT

The colorization of monochromatic images has demonstrated utility in enhancing human comprehension of images and boosting the accuracy of succeeding image-processing tasks. Nonetheless, current fully automated colorization methodologies often exhibit optimal performance based on the input image's nature and the employed algorithms' architectural specifics. In response to this challenge, this paper introduces a novel methodology aimed at effectively predicting the most suitable colorization model for a given input image. This comprehensive approach is characterized by exceptional accuracy across diverse datasets.

Keywords: Convolutional neural networks, Image processing, Deep learning, Colorization algorithms

1. INTRODUCTION

The colorization of monochromatic images represents a significant challenge within the domains of computer vision and image processing. Numerous applications, including but not limited to historical image restoration[1] medical imaging,[2,3] photography of astronomical objects[3] and CCTV surveillance,[3] stand to derive considerable benefits from the refinement of colorization techniques. Moreover, improved outcomes are anticipated across various subsequent tasks upon the integration of accurately colorized input data.[1]

Colorization algorithms have undergone significant advancements in recent years. Initially, manual colorization methods necessitated human intervention, where individuals applied color to each image by hand.[4] Subsequently, artificial intelligence (AI) facilitated partial automation of this process, enabling colorization based on rudimentary inputs such as scribbles or from a reference image.[5] Progressing further, the advent of Deep Neural Networks (DNNs) and Convolutional Neural Networks (CNNs) has revolutionized colorization capabilities. These sophisticated algorithms now possess the capacity to colorize grayscale images autonomously, leveraging DNNs and CNNs to infer and approximate the RGB channels, thereby "hallucinating" plausible colorizations.[5,6]

In recent years, learning-based approaches employing DNNs have demonstrated remarkable success, attributed to their adeptness in extracting pertinent features from images. Noteworthy architectures encompass Convolutional Neural Networks (CNNs),[3,5,7] Variational Autoencoders (VAEs),[8] and Generative Adversarial Networks (GANs).[1,9] These methodologies have significantly advanced the field of image processing and have been instrumental in enhancing colorization capabilities. In contrast, classification algorithms have gained widespread adoption, ranging from conventional Convolutional Neural Networks (CNNs)[10,11] to the latest advancements employing transformers.[12] The latter is characterized by their computational efficiency. Both, exploit feature extraction mechanisms akin to those utilized in colorization algorithms. However, they incorporate a classification head at the end of the network.

The diverse architecures among colorization models, within DNNs, have yielded varied accuracies across specific datasets, with certain models excelling in human-centric images, landscapes, or objects.[4] While this variability is inherent to their design and learning methodologies, it presents an opportunity for a more generalized approach. Hence, this paper introduces a comprehensive framework aimed at optimizing image colorization without regard to its content. By integrating a set of pre-trained models with diverse architectures and employing

*Further author information:
U. Orozco-Rosas: E-mail: ulises.orozco@cetys.mx

a classifier to intelligently assign uncolored images to the most suitable colorizer for maximal accuracy, this framework reduces limitations imposed by content specificity in the input images.

The main contributions of this paper are:

- An exhaustive exploration of various automatic colorization baselines across diverse models, aimed at discerning optimal algorithm-dataset pairings.
- Initial steps of a comprehensive framework integrating a classification model to facilitate the selection of the most suitable colorization model from a pool of alternatives, resulting in an overall reduction of the colorization error of black and white images.

2. FOUNDATION

Grayscale images are also called 2D images because they are composed of a 2D matrix where each element is a single number from 0 to 255 to determine the shade of gray. Compared to colored images, also called 3D images because they need at least 3 different matrices (RGB).[13] The size of the image is equal to the size of the matrix, and just like in black and white images, each channel (RGB) also saves a number from 0-255 which determines the intensity of that color. Visual data such as images is much harder for an AI to recognize than numerical values, this is where Deep Learning comes along.

Deep Learning is a subset of machine learning algorithms that "extracts features and attributes from raw data by using a neural network".[13] Many areas are taking advantage of the new world of data understanding like computer vision (in self-driving cars), natural language processing, recommendation engines, automatic colorization, and advertising,[13] just to name a few.

When mentioning neural networks, people associate them with how neurons work in human brains by creating connections between them,[13] and that's what artificial neural networks (ANN) are trying to simulate. In a mathematical sense, a neuron, also called a perceptron, receives several inputs and returns a single output.[14] The inputs are denoted by x_i, and each input is multiplied by a weight w_i, this is a number that grants certain inputs more importance than others. In the body, there is a compilation (sum) of all the inputs times the weights ($w_i x_i$).[14] And lastly, this is passed through an activation function, this function translates the result to either an input for another function or the classification directly. This can be seen visually in Fig. 1. The action of the activation function can vary depending on the type of neuron that it is.

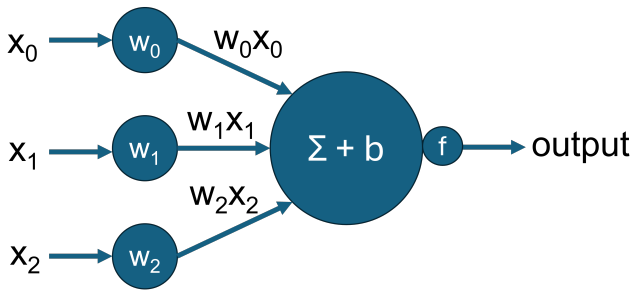

Figure 1: Representation of an artificial neuron with its components.[13]

There are 3 main types of neurons, depending on the layer (block of nodes) they are located in:

- Input layer: Just passing information.
- Hidden layer: Perform the calculations by passing the weights.
- Output layer: Uses the activation function to classify the number into an actual class prediction.

A complete neural network is a combination of many neurons, working together and passing information from layer to layer, just like a real neural network works in brains. As shown in Fig. 2, each circle is a node, the red box is the input layer, the last neuron is the output layer, the middle layers are the hidden layers and the lines imply the passing of information from the output of a node to the input of another.

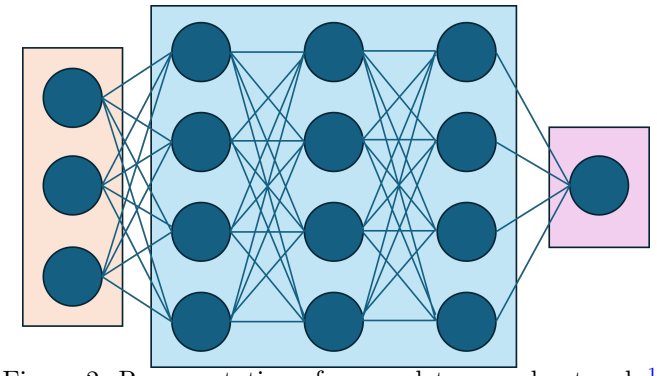

Figure 2: Representation of a complete neural network.[13]

The downside of this model is found inside its hidden layers, where the model can create these connections on its own, and the programmer will not know how and why these connections were made.[13] That is not exactly a bad thing, because the data scientists only care about the result of the model and its accuracy, the methodology to get a result is a "black box". It needs a lot more data than other machine learning algorithms,[13] so the training dataset must be specifically designed.

A similar adaptation of Artificial Neural Networks is the Convoluted Neural Networks (CNN), which specializes in recognizing patterns. It keeps the same architecture as ANN but adds some *convoluted layers* before it enters the *fully-connected layers* to reduce the number of parameters in this last layer.[15] For example, if there is an image, with thousands of raw pixels, in 3 layers (RGB), and each pixel is connected with a group of neurons, the number of connections in a pure ANN will grow exponentially. And that is not considering the different layers within the hidden layer. So the model becomes very computationally expensive. With CNN, we give a more filtered input before it reaches the fully connected part of the ANN.

These layers go from least abstract to more complex as the data goes from layer to layer as shown in Fig. 3. So it calculates a numerical value of how close is a specific section of the input to the pattern it is comparing it with.[16] These layers are also called filters, and they are usually just a numerical matrix that is multiplied by the input (similar to the weights),[13,15,16] and create a different representation of the original image. This is what the convoluted part of CNN refers to, "The convolution of a temporal or spatial signal with another signal produces a modified version of the initial signal"[13] in this case the image pixels.

CNN also replicates the biological behavior of the receptive field. El-Amir and Hamdy explain it as "Individual neurons respond to stimuli only in a restricted region of the visual field known as the receptive field. A collection of such fields overlaps to cover the entire visual area".[13] This translates in the neural network sense as every single neuron looks at only a certain part of the image and not the whole picture.[15] As the inputs travel deeper through the layers of the model, they start combining with the adjacent sections again.

This can lead to a problem, where the middle pixels are treated with a higher priority due to them being in more overlapping areas than the pixels in the edges. To fix this, CNN adds to the input a *padding,* which are rows of 0s in the edges of the input.[15] There are different sizes of the padding which affect the size of the resulting matrix:

- Padding Full: Each pixel is treated with the same priority, and the output matrix is bigger than the input matrix.
- Padding Same: The output of the convoluted operation is the same size as the input.

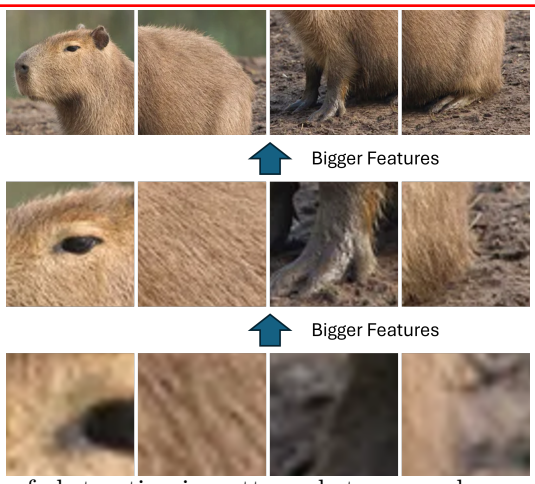

Figure 3: Development of abstraction in patterns between each convoluted layer in CNN.[15]

- Padding Valid: The same as no padding, and the output size is much smaller than the input.[13]

To calculate the size of the resulting matrix (O), the Eq. 1 occurs.

$$O = 1 + \frac{N - F}{S} \tag{1}$$

Where F is the dimension of the filter, N is the size of the whole image and S is the stride size, which refers to the distance between any side of the beginning of this filter and the edge of the adjacent one.

In Fig. 4 there is a complete representation of a CNN, the first 4 layers are the convoluted section of the CNN, and the last layers are the normal ANN with a big set of neurons.

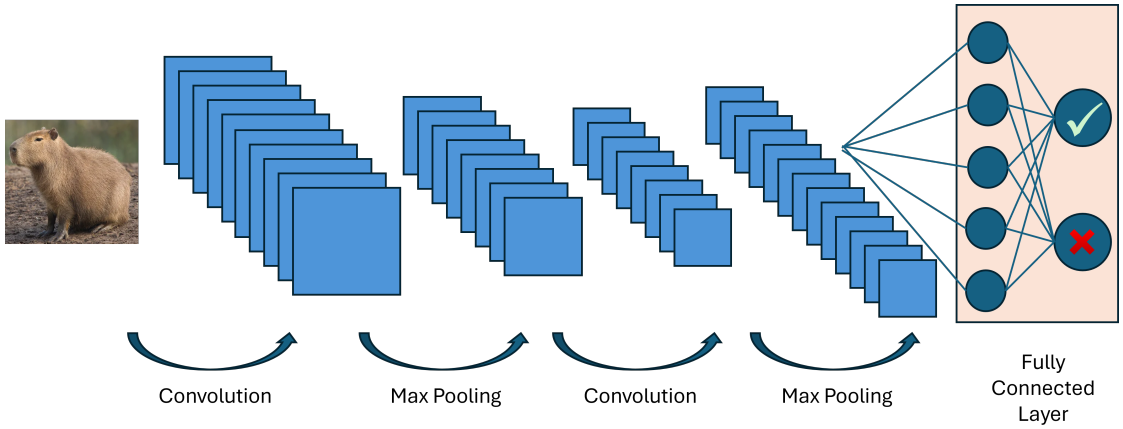

Figure 4: Representation of the complete CNN.[13]

Using CNN, was how the first colorization algorithms were created. Image colorization is the process of estimating RGB colors for grayscale images to improve their aesthetic and perceptual quality. This is known to be a complex job that often requires prior knowledge of image content and manual adjustments to achieve results. Also, since objects can be of different colors, there are many possible ways to assign colors to pixels in an image, which means there is no single solution to this problem.[4]

2.1 Classification Algorithms

Image classification is a modeling approach capable of assigning a class label to an input image.[17] This process involves the extraction of learned features from the image, as explained in Section 2, utilizing convolutional operations. Subsequently, these extracted features are processed to yield an output corresponding to a predefined set of classes.

Recent advancements in vision-processing algorithms have experienced a surge, facilitated by the ability to expand network architectures without constraints imposed by hardware limitations. Consequently, Convolutional Neural Networks (CNNs) have garnered significant attention in research,[18,19] alongside the emergence of vision-transformers[20] and other models.[21] These models have demonstrated superior classification accuracy and underscored the advantages of transferability in the realm of vision processing.

Transfer Learning has emerged as a prevalent strategy to address the substantial demands for task-specific data and computational resources inherent in training deep neural networks. This approach entails training a model initially on a large, generic dataset in a relevant task. Subsequently, the learned weights are transferred to a new model, which incorporates the pre-trained weights in its initial layers while fine-tuning the latter layers to accommodate the nuances of a distinct dataset or similar task.[22]

2.2 Colorization Algorithms

There are two main approaches to image colorization: one that requires the user to assign colors to some regions and extend that information to the entire image, and another that tries to learn the color of each pixel of a color image with similar content. With the rapid development of deep learning techniques, a variety of image colorization models have been introduced, and various deep learning models, ranging from the first brute force networks, convolutional neural networks (CNN), to generative adversarial networks (GAN). These coloring networks differ in many important ways, including network architecture and depth, loss functions, learning strategies, etc. This paper will focus on the approaches that do not require human input, due to having an unfair advantage with our comparing metric.

Precisely, one of the best techniques that currently exist was developed by implementing a CNN. Colorful Image Colorization[6] takes the underlying uncertainty of the problem by posing it as a multinomial classification problem and uses class rebalancing at training time to increase color variation in the result. The system is implemented as feedback to the CNN at the time of testing and is trained on over a million color images. This is achieved by working through the three-dimensional CIE Lab color spectrum, which expresses color as three values: L for perceptual lightness, and a and b for the four unique colors of human vision: red, green, blue, and yellow. In this sense, I (Intensity) can be calculated by Eq. 2.

$$I = \frac{R + G + B}{3} \tag{2}$$

From there you can calculate a with Eq. 3, and b with Eq. 4.[23]

$$a = \frac{B}{I} - \frac{R + G}{2I} \tag{3}$$

$$b = \frac{R - G}{I} \tag{4}$$

Given the luminosity channel L, the system predicts the corresponding a and b color channels of the image. For an input, the system poses a mapping learned with a CNN to predict a probability distribution over possible a and b (coloring) values. Any color photo can be used as a training example, simply by taking the L channel of the image as input, and it is a and b channels as the accuracy monitoring signal. Afterward, the model reweights the loss of each pixel at train time based on the pixel color rarity. And finally, the final colorization is produced by taking the strengthened mean of the distribution.

Table 1: Different colorization algorithm baselines for an example of an image.

Original	B / W	Izuka 2016	Larsson 2016	Zhang 2016	Kang 2023
					

Subsequently, the second model taken into consideration is Real-Time User-Guided Image Colorization with Learned Deep Priors.[24] The system uses user interaction in the form of real-time distribution of colored points upon the grayscale image. Two variants of the colorization neural networks are trained: the local hints and the global hints network, local hints refer to the colored points entered by the user and the global hints involve factors such as global color distribution or average image saturation. The local hints network treats the user points and predicts the color distribution and the global hints network incorporates the global statistics provided by the user into the main system. The colorization is performed in the CIE Lab color spectrum mentioned before, in this context the grayscale image serves as the L or lightness in the color space. The result of the model is the approximation of the ab color channels corresponding to the image.

3. EFFICIENT AND ACCURATE COLORIZATION ON *MULTIPURPOSE* DATASET

The subsequent sections describe the dataset formation process employed for the classifier training, alongside an exposition of the classifier's architectural framework.

3.1 Problem Definition

The problem's input consists of a monochromatic image extracted from one of the three datasets outlined in Section 4.1. The objective is to generate a colorized version of the same image, facilitated by the most optimal model available. MobileNetV2[25] serves as the backbone model for classification within this framework, chosen for its modest parameter count, efficiency in inference, and transferability. This classification model is subsequently fine-tuned using the dataset curated within this study, as detailed in Section 3.2. Each image in the dataset underwent resizing to dimensions of 64x64 for normalization, employing Python 3.11 libraries such as Pillow for image manipulation. Subsequently, the images were passed through a monochromatic filter to produce black-and-white renditions.

3.2 Proposed Dataset: Best Colorizer

Each of the publicly available image datasets undergoes inference processing through the pre-trained models. Specifically, Izuka 2016[26] was trained to utilize the Places dataset,[27] rendering it more effective for outdoor scenarios. Larsson 2016,[28] on the other hand, was trained on ImageNet[29] and a Sun database,[30] making it well-suited for colorizing object images. Zhang 2016,[6] pre-trained on Imagenet, was also tailored for coloring specific object images. Lastly, Kang 2023,[31] a more recent approach, underwent training on ImageNet as well but has demonstrated notable success in historic black-and-white image colorization. This presents an advantage, particularly for human-centric images.

The outcomes of each model are preserved and compared with the original colored image, as illustrated in Table 1. To derive a singular metric for each image, we employ the Color Peak Signal-to-Noise Ratio (CPSNR) as utilized in a prior colorization algorithm.[32] CPSNR is calculated by applying a logarithmic function to the average Mean Squared Error (MSE) of each channel for every pixel. This approach intensifies the penalty for discrepancies between the colorized and the ground truth images.

Despite incorporating all available datasets, there remains a significant class imbalance, with the Larsson algorithm having a disproportionately higher number of samples. Particularly since the Larsson model was pre-trained on ImageNet—a dataset comprising a larger quantity of colored images. Consequently, this model

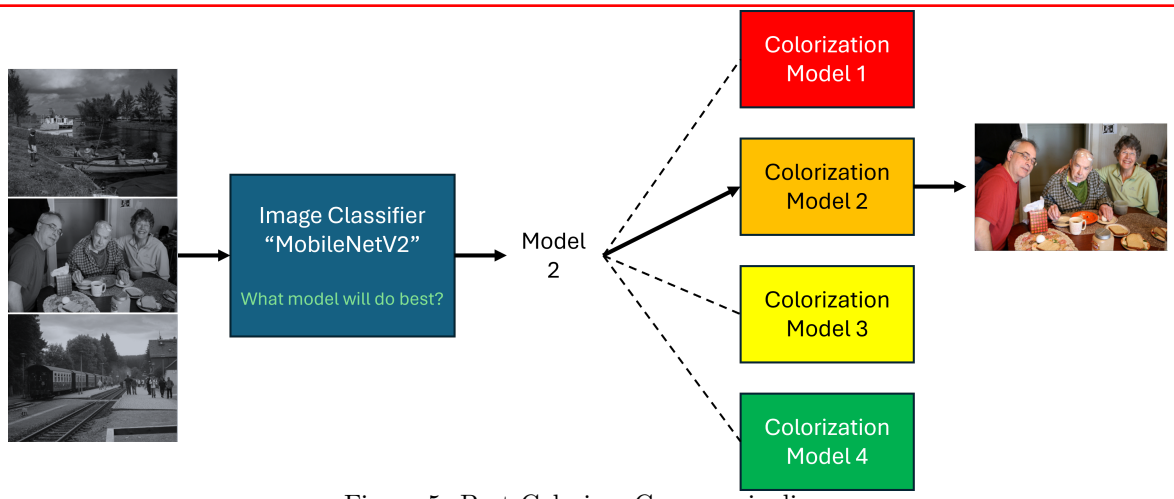

Figure 5: Best Colorizer Guesser pipeline.

naturally outperforms others and thus has a greater number of images associated with it. This imbalance affects the neural network's performance. To address this issue, we integrated the Synthetic Minority Over-sampling Technique (SMOTE)[33] into our pipeline. SMOTE mitigates the class imbalance by oversampling the minority class through the generation of synthetic examples. These synthetic samples are produced by interpolating between multiple instances of the minority class within a specified neighborhood.

The process of applying SMOTE and evaluating the dataset with four different algorithms to determine the optimal algorithm results in a refined dataset. In this dataset, each image is associated with the model that produced the most accurate colorization result.

3.3 Best Colorizer Guesser

The subsequent step involves devising an algorithm that takes the *Best Colorizer* as input and predicts the most suitable algorithm among the downloaded ones for a given image, excluding the need to execute an image on all algorithms. While no specific architecture is mandated, we opt for MobileNetV2[25] to conserve resources, particularly on low-powered devices. The classifier's outcome facilitates the prediction of the algorithm that will yield the optimal performance for each image, thereby executing the appropriate inference algorithm. The entirety of this pipeline is illustrated in Fig. 5. As a result, MobileNetV2 serves as the backbone of our model. By employing transfer learning, we incorporate ImageNet's pre-trained weights into the initial layers of the model and freeze these layers, but we only include the first 10 blocks of the model the rest are excluded to be retrained with our dataset. We subsequently train the final fully connected layers (FCL) of the model to optimize the prediction of the best-performing colorization model. Excluding the top layer, we introduce dense layers with 256, 100, and 50 neurons, respectively, followed by an output layer with 4 neurons. To mitigate overfitting, we incorporate a dropout rate of 0.2 between each layer.

4. RESULTS

In this section, the experimental setup for the proposed Best Colorizer Guesser and the overall results are described.

4.1 Experimental Setup

The pool of models chosen for our study represents the cutting edge in colorization, encompassing seminal works such as Izuka 2016,[26] Larsson 2016,[28] Zhang 2016,[6] and Kang 2023.[31] Utilizing these established models as distinct pretrained entities, we curate a tailored dataset comprising images sourced from TinyImageNet[34] for object colorization, and the Landscape Pictures dataset from Kaggle for landscape colorization. Subsequently, for each image within these datasets, color predictions are obtained from all baseline models, and the one most

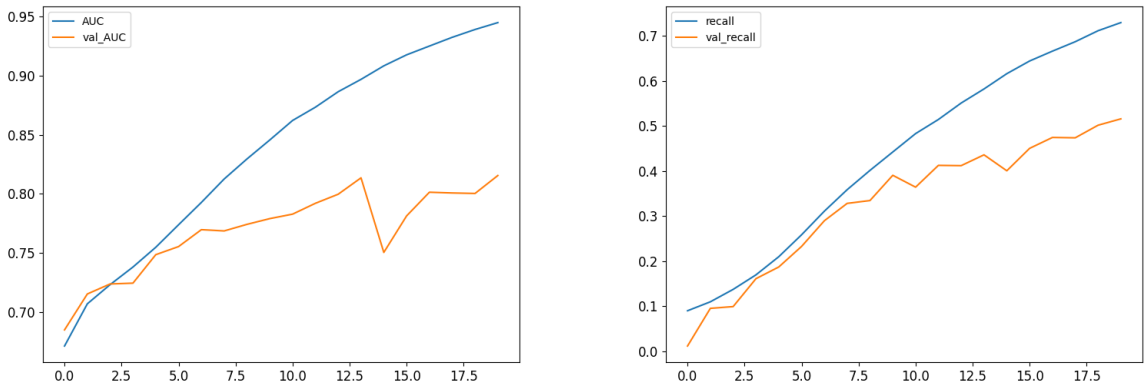

Figure 6: Learning Curve

Table 2: Colorization algorithm baselines in the testing section.

Izuka 2016	Larsson 2016	Zhang 2016	Kang 2023	Proposed Best Colorizer Guesser
30.16	29.03	29.66	29.59	29.36

closely aligned with ground truth is designated as the output label. The dataset was partitioned into 70% for training, 20% for testing, and 10% for validation.

We benchmark our model against each baseline model used in isolation. The metric employed to gauge the accuracy of our classifier encompasses overall accuracy across all classes is Area Under the Curve (to measure True Positive Rate over False Positive Rate) and recall for each respective class (in this instance, each baseline model). This facilitates analysis to ascertain if our model exhibits a bias towards specific models over others across different images. The model was trained for a total of 20 epochs.

4.2 Results

After training the model for 20 epochs, we achieved a 0.83 AUC and a 0.52 recall in estimating the correct colorizer. As shown in Fig. 6, the learning curves illustrate the model's performance metrics improving gradually with each epoch.

By comparing each of the models on the testing section of the dataset, we calculated the CPSNR for each image. These results are presented in Table 2. Based on the CPSNR of the testing images, our proposed model achieves better results than most of the models individually. Specifically, the CPSNR scores for the individual models are 30.16 for Izuka, 29.03 for Larsson, 29.66 for Zhang, and 29.59 for Kang. Our proposed model, by selecting the best model for each image, achieves an overall CPSNR of 29.36.

5. CONCLUSIONS

In numerous real-world scenarios, the utility of a multi-purpose colorization algorithm becomes evident, specifically when subsequent tasks can greatly benefit from the addition of color to black-and-white images. While multiple successful algorithms addressing this problem have emerged over the years, the architecture and dataset upon which they were trained can significantly influence the model's accuracy in diverse environments. Thus, we propose Best Colorizer Guesser, a multipurpose colorization algorithm capable of determining the most suitable colorization algorithm from a pool of candidates and delivering precise colorization results. Our results demonstrate that our approach has the potential to effectively address the individual weaknesses of each model while maintaining the lightweight nature of standard colorization algorithms.

ACKNOWLEDGMENTS

This work was supported by the Coordinación Institucional de Investigación of CETYS Universidad, and by the Mexican National Council of Science and Technology (Consejo Nacional de Humanidades, Ciencias y Tecnologías, CONAHCYT).

REFERENCES

[1] Poterek, Q., Herrault, P.-A., Skupinski, G., and Sheeren, D., "Deep learning for automatic colorization of legacy grayscale aerial photographs," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **13**, 2899–2915 (2020).

[2] Chen, S., Xiao, N., Shi, X., Yang, Y., Tan, H., Tian, J., and Quan, Y., "Colormedgan: A semantic colorization framework for medical images," *Applied Sciences* **13**(5), 3168 (2023).

[3] Shankar, R. S., Mahesh, G., Murthy, K., and Ravibabu, D., "A novel approach for gray scale image colorization using convolutional neural networks," in [*2020 International Conference on System, Computation, Automation and Networking (ICSCAN)*], 1–8, IEEE (2020).

[4] Žeger, I., Grgic, S., Vuković, J., and Šišul, G., "Grayscale image colorization methods: Overview and evaluation," *IEEE Access* (2021).

[5] An, J., Kpeyiton, K. G., and Shi, Q., "Grayscale images colorization with convolutional neural networks," *Soft Computing* **24**, 4751–4758 (2020).

[6] Zhang, R., Isola, P., and Efros, A. A., "Colorful image colorization," in [*ECCV*], (2016).

[7] Anitha, A., Shivakumara, P., Jain, S., and Agarwal, V., "Convolution neural network and auto-encoder hybrid scheme for automatic colorization of grayscale images," in [*Smart Computer Vision*], 253–271, Springer (2023).

[8] Deshpande, A., Lu, J., Yeh, M.-C., Jin Chong, M., and Forsyth, D., "Learning diverse image colorization," in [*Proceedings of the IEEE conference on computer vision and pattern recognition*], 6837–6845 (2017).

[9] Li, B., Lu, Y., Pang, W., and Xu, H., "Image colorization using cyclegan with semantic and spatial rationality," *Multimedia Tools and Applications* , 1–15 (2023).

[10] Tan, M. and Le, Q., "Efficientnet: Rethinking model scaling for convolutional neural networks," in [*International conference on machine learning*], 6105–6114, PMLR (2019).

[11] Cai, Y., Zhou, Y., Han, Q., Sun, J., Kong, X., Li, J., and Zhang, X., "Reversible column networks," *arXiv preprint arXiv:2212.11696* (2022).

[12] Chen, X., Wang, X., Changpinyo, S., Piergiovanni, A., Padlewski, P., Salz, D., Goodman, S., Grycner, A., Mustafa, B., Beyer, L., et al., "Pali: A jointly-scaled multilingual language-image model," *arXiv preprint arXiv:2209.06794* (2022).

[13] El-Amir, H. and Hamdy, M., [*Deep Learning Pipeline*], Apress Berkeley, CA, Jizah, Egypt, 1 ed. (2020).

[14] Bergel, A., [*Agile Artificial Intelligence in Pharo*], Apress Berkeley, CA, Santiago, Chile, 1 ed. (2020).

[15] Albawi, S., Mohammed, T. A., and Al-Zawi, S., "Understanding of a convolutional neural network," in [*2017 International Conference on Engineering and Technology (ICET)*], 1–6 (2017).

[16] Technology, I., "What are convolutional neural networks (cnns)?."

[17] Bird, J. J. and Lotfi, A., "Cifake: Image classification and explainable identification of ai-generated synthetic images," *IEEE Access* (2024).

[18] Mahajan, D., Girshick, R., Ramanathan, V., He, K., Paluri, M., Li, Y., Bharambe, A., and Van Der Maaten, L., "Exploring the limits of weakly supervised pretraining," in [*Proceedings of the European conference on computer vision (ECCV)*], 181–196 (2018).

[19] Huang, Y., Cheng, Y., Bapna, A., Firat, O., Chen, D., Chen, M., Lee, H., Ngiam, J., Le, Q. V., Wu, Y., et al., "Gpipe: Efficient training of giant neural networks using pipeline parallelism," *Advances in neural information processing systems* **32** (2019).

[20] 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).

[21] Tolstikhin, I. O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., et al., "Mlp-mixer: An all-mlp architecture for vision," *Advances in neural information processing systems* **34**, 24261–24272 (2021).

[22] Kolesnikov, A., Beyer, L., Zhai, X., Puigcerver, J., Yung, J., Gelly, S., and Houlsby, N., "Big transfer (bit): General visual representation learning," in [*Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*], 491–507, Springer (2020).

[23] Deshpande, A., Rock, J., and Forsyth, D., "Learning large-scale automatic image colorization," in [*2015 IEEE International Conference on Computer Vision (ICCV)*], 567–575 (2015).

[24] Zhang, R., Zhu, J.-Y., Isola, P., Geng, X., Lin, A. S., Yu, T., and Efros, A. A., "Real-time user-guided image colorization with learned deep priors," *ACM Trans. Graph.* **36** (jul 2017).

[25] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C., "Mobilenetv2: Inverted residuals and linear bottlenecks," in [*Proceedings of the IEEE conference on computer vision and pattern recognition*], 4510–4520 (2018).

[26] Iizuka, S., Simo-Serra, E., and Ishikawa, H., "Let there be color! joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification," *ACM Transactions on Graphics (ToG)* **35**(4), 1–11 (2016).

[27] Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., and Oliva, A., "Learning deep features for scene recognition using places database," *Advances in neural information processing systems* **27** (2014).

[28] Larsson, G., Maire, M., and Shakhnarovich, G., "Learning representations for automatic colorization," in [*Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*], 577–593, Springer (2016).

[29] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L., "Imagenet: A large-scale hierarchical image database," in [*2009 IEEE conference on computer vision and pattern recognition*], 248–255, Ieee (2009).

[30] Xiao, J., Hays, J., Ehinger, K. A., Oliva, A., and Torralba, A., "Sun database: Large-scale scene recognition from abbey to zoo," in [*2010 IEEE computer society conference on computer vision and pattern recognition*], 3485–3492, IEEE (2010).

[31] Kang, X., Yang, T., Ouyang, W., Ren, P., Li, L., and Xie, X., "Ddcolor: Towards photo-realistic image colorization via dual decoders," in [*Proceedings of the IEEE/CVF International Conference on Computer Vision*], 328–338 (2023).

[32] Pang, J., Au, O. C., Tang, K., and Guo, Y., "Image colorization using sparse representation," in [*2013 IEEE International Conference On Acoustics, Speech And Signal Processing*], 1578–1582, IEEE (2013).

[33] Fernández, A., Garcia, S., Herrera, F., and Chawla, N. V., "Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary," *Journal of artificial intelligence research* **61**, 863–905 (2018).

[34] Le, Y. and Yang, X., "Tiny imagenet visual recognition challenge," *CS 231N* **7**(7), 3 (2015).