UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e ingenierías

Very Deep Convolutional Neural Networks for Image Colorization

Stefano Camilo Cevallos Aguirre

Ingeniería en Ciencias de la Computación

Trabajo de fin de carrera presentado como requisito
para la obtención del título de
Ingeniero en Ciencias de la Computación

Quito, 6 de mayo de 2021

UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e ingeniería

HOJA DE CALIFICACIÓN
DE TRABAJO DE FIN DE CARRERA

Very Deep Convolutional Neural Networks for Image Colorization

Stefano Camilo Cevallos Aguirre

Nombre del profesor, Título académico	**Noel Pérez, Ph.D.**

Quito, 6 de mayo de 2021

© DERECHOS DE AUTOR

Por medio del presente documento certifico que he leído todas las Políticas y Manuales de la Universidad San Francisco de Quito USFQ, incluyendo la Política de Propiedad Intelectual USFQ, y estoy de acuerdo con su contenido, por lo que los derechos de propiedad intelectual del presente trabajo quedan sujetos a lo dispuesto en esas Políticas.

Asimismo, autorizo a la USFQ para que realice la digitalización y publicación de este trabajo en el repositorio virtual, de conformidad a lo dispuesto en la Ley Orgánica de Educación Superior del Ecuador.

Nombres y apellidos:	Stefano Camilo Cevallos Aguirre
Código:	00139936
Cédula de identidad:	1719313585
Lugar y fecha:	Quito, 6 de mayo de 2021

ACLARACIÓN PARA PUBLICACIÓN

Nota: El presente trabajo, en su totalidad o cualquiera de sus partes, no debe ser considerado como una publicación, incluso a pesar de estar disponible sin restricciones a través de un repositorio institucional. Esta declaración se alinea con las prácticas y recomendaciones presentadas por el Committee on Publication Ethics COPE descritas por Barbour et al. (2017) Discussion document on best practice for issues around theses publishing, disponible en http://bit.ly/COPETheses.

UNPUBLISHED DOCUMENT

Note: The following capstone project is available through Universidad San Francisco de Quito USFQ institutional repository. Nonetheless, this project – in whole or in part – should not be considered a publication. This statement follows the recommendations presented by the Committee on Publication Ethics COPE described by Barbour et al. (2017) Discussion document on best practice for issues around theses publishing available on http://bit.ly/COPETheses.

RESUMEN

La dificultad de la colorización de imágenes ha incentivado a investigadores en el campo de las ciencias de la computación hacia el desarrollo de algoritmos capaces de facilitar esta tarea. A pesar de que sus esfuerzos han dado como fruto numerosos avances; los algoritmos desarrollados aun presentan varias limitaciones tanto en términos de la calidad de las imágenes generadas, como de la cercanía de estas colorizaciones a los colores reales representados en las fotografías. Por estas razones, en este trabajo proponemos un método automático de colorización de imágenes basado en modelos de aprendizaje profundo. Dicho modelo logra resultados de error absoluto medio y error cuadrático medio que demuestran que las colorizaciones producidas presentan colores bastante similares a los originales; adicionalmente los resultados de Proporción Máxima de Señal a Ruido indican que las imágenes generadas por este método exhiben una pérdida de calidad de imagen relativamente baja. Finalmente, la validación de las imágenes mediante percepción humana respalda los buenos resultados numéricos, mostrando imágenes con colorizaciones y saturación subjetivamente buenos.

Palabras clave: Automatic colorization, autoencoders, CNNs, VGG, image-to-image translation.

ABSTRACT

The difficulty behind image colorization has incentivized computer scientist towards the development of algorithms capable of simplifying the colorization process. Despite the numerous advancements yielded by these efforts; the developed algorithms are still limited in terms of the quality of the generated images, as well as the similarity of the generated colors to ground truth. For these reasons, in this work we propose an automatic image colorization method based on deep learning models. Said model achieves mean absolute error and mean quadratic error results that demonstrate the similarity between the generated colors and the ones present in the ground truth images; additionally, the peak signal-to-noise ratio results indicate that the colorized images generated by the proposed method exhibit a relatively low drop in image quality. Finally, the validation of the images by means of expert criterion validation supports the good numeric results, showing subjectively good colorization and saturation in the generated samples.

Key words: Automatic colorization, autoencoders, CNNs, VGG, image-to-image translation.

TABLA DE CONTENIDO

Introduction 10

Materials and methods 13

  Database 13

  Autoencoders 13

  Very Deep Convolutional Networks (VGG) 15

  Proposed Model 16

    Encoder Architecture. 17

    Decoder Architecture. 18

  Experimental setup 20

    Data processing. 20

    Training and testing partitions. 20

    Model configuration. 20

    Assessment metrics. 20

Results and discussion 22

    Quantitative Results 22

    Qualitative Results 22

    Limitations 24

Conclusions and future work 26

References 27

ÍNDICE DE TABLAS

Table 1: Average Metrics for the Selected Models 22

ÍNDICE DE FIGURAS

Figure 1: Traditional autoencoder model architecture 14
Figure 2: Workflow of a Convolutional Autoencoder 15
Figure 3: Original VGG16 architecture 15
Figure 4: General colorization process for a real-life use case workflow 17
Figure 5: Detailed encoder model and workflow 17
Figure 6: Detailed decoder model and workflow 19
Figure 7: Successful performance examples of colorization for the Fruits-360 dataset 23
Figure 8: Successful performance examples of colorization for the FFHQ dataset 23
Figure 9: Examples of unexpected green image colorization 24
Figure 10: Example of colorization under different lightning conditions 25

INTRODUCTION

The process of image colorization consists of painting colors over monochrome images (Cohen-Or, et al., 2007). At first glance this task may seem trivial; after all, it is easy for humans to imagine the colors that correspond to the objects portrayed within a picture (even if some of them can have more than one correct color). Nevertheless, the colorization process gets much more difficult when the task is not only imagining the colors but painting them to turn the monochrome picture into a fully colored one. This is very time consuming and usually requires a skilled artist.

For these reasons, colorization has become a very researched topic amongst computer scientists. Hence numerous advancements have been made in the form of colorization algorithms. These algorithms can be divided into two groups based on how they correlate grayscale to color (Efros et al., 2016). The first group can be considered a semi-automatic approach as it needs some form of user input. The most basic version of such an algorithm uses "scribbles" or dots of color made to the grayscale image by the user, this was originally proposed Levin et al. (2004) and later improved by Chen et al. (2005) and Efros et al. (2017). Other semi-automatic techniques were proposed by: Ashikhmin et al. (2002) where reference images are used for colorizing the grayscale pictures, yielding mostly unsatisfactory results; and Chang et al. (2015) where a reference color palette is required as an input.

Although being simpler than manual colorization, user guided techniques still present several difficulties for the users. This caused researchers to turn their efforts towards developing fully automatic colorization techniques. These are mostly based on deep learning and try to take advantage of the large image datasets available to the public by turning them into color-grayscale pairs and using them to train neural networks. Cheng et al. (2015) proposed using Deep neural networks (DNNs) to approach the colorization task as a least squares minimization problem, yielding subjectively good results. Despite this, most modern colorizers replace DNNs with Convolutional Neural Networks (CNNs). For example, In, Colorful Image Colorization (Efros et al., 2016), the task is proposed as a classification problem, and a CNN is trained using the ImageNet large scale visual recognition challenge (ILSVRC) dataset (Berg, et al., 2015). Although this approach results in colorized images portraying vibrant colors, they are not a good representation of the tonalities depicted in the ground truth images, resulting on a peak-signal-to-noise-ratio (PSNR) of 22.04 dB. Another attempt at using CNNs has been proposed by Larsson et al. (2017) where a model is trained to predict hue and chroma distributions on a per pixel basis based on spatially localized multilayer slices; this model was trained on SUN-A and SUN-6 subsets of the SUN dataset (Ehinger et al., 2010), and achieves a PSNR result of 24.93 dB.

Finally, after the use of simple CNNs and autoencoders, Generative Adversarial Neural Networks (GANs) were proposed for the task. Ebrahimi et al. (2018) conducted a research to determine how effective these networks are as well as the best training strategies used for speeding up the training process and encouraging convergence. Another example of GANS used in this context was proposed by Ballester et al. (2020) in the form of a GAN model, where a two part generator is used, the first part which outputs the chrominance information while the second outputs a class distribution vector for adding a semantic representation of the picture; this method was trained over ILSVRC (Berg, et al., 2015) and achieved a PSNR of 25.57 dB. Finally, another important contribution made in their work was the addition of a non-human algorithm evaluation system in the form of the PSNR values used to rank both their model as well as some of the state-of-the-art methods.

Despite the advancements made in the recent years, the process of automatic colorization still presents various limitations such as high losses in image quality, low color saturation (Ballester et al., 2020) and difficulty generating colors similar to the original picture

(Efros et al., 2016). Therefore, in this work we propose an image colorization method based on deep learning and the most recent developments in the field.

MATERIALS AND METHODS

Database

In order to test the different capabilities of the proposed method, two datasets were selected for both training and testing the model, the Fruits-360 dataset (Muresan & Oltean, 2017) and the Flickr-Faces-HQ dataset (Aila et al., 2018).

First, the Fruits-360 dataset is comprised of over 90 000 images of distinct varieties of 70 different fruits. A total of 131 classes conform the dataset, with an average of 688 images per class. The large number of classes and the little color variation between images of the same class is useful for determining ability of the model to recognize patterns in the form of fruits. For example, painting apples red and bananas yellow means being able to recognize between them.

Secondly, the Flickr- Faces-HQ dataset consists of 70 000 portraits of people of various complexions and other physical characteristics (e.g., hair color, eye color, etc.) with different backgrounds and in different lightning settings. For this reason, FFHQ serves as a good benchmark of how well the network performs on the colorization process when confronted with a task of a higher complexity.

Autoencoders

Autoencoders (AEs), originally introduced by Hinton et al. (1985), are neural networks in which the input is the same as the output. The original AE architecture uses fully connected layers and is composed of two parts: the encoder and decoder (see Figure 1). The objective of the encoder is to be able to create a compressed (latent) representation of its inputs. Meanwhile, the goal of the decoder is to recreate these inputs by using only the latent representation previously generated and adhering to the restriction that the dimensionality of the outputs must be identical to that of the inputs.

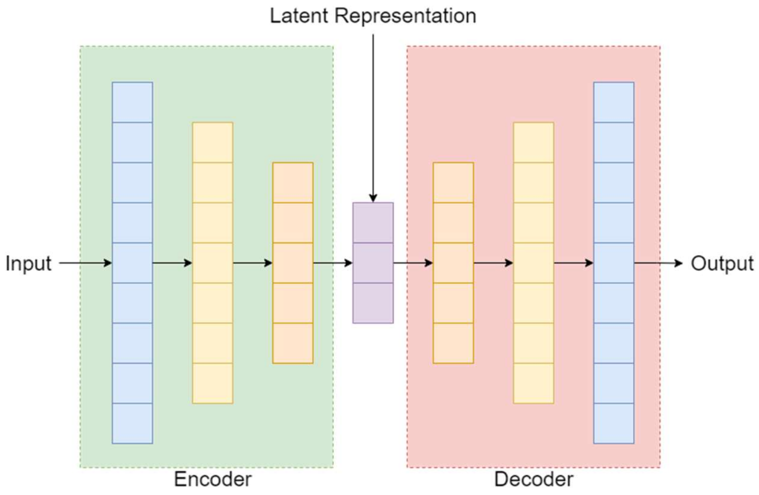

Figure 1: Traditional autoencoder model architecture

Traditional AEs were not designed to be used in image processing related tasks; instead, for this purpose a variant on the conventional AE architecture called Convolutional Autoencoder (CAE) is used. This variation extends the functionality of its predecessor by replacing the fully connected layers with convolution plus pooling layers in the encoder and transposed convolution layers in the decoder (Zhang, 2018).

The general CAE workflow can be divided into an encoding and decoding segment. In the encoding or feature extraction portion of the workflow; an image is passed as input and is subject to any number of convolution plus pooling layer sequences. During this process, by means of convolution operations the most important features of the image are extracted in the form of filters. Subsequently, by means of pooling operations the image size is reduced to half, effectively increasing information density, and enabling the extraction of higher complexity patterns through ensuing convolution layers. As with traditional AEs the product of this portion of the network is the latent representation, which in the case has the form of a feature map. Finally, in the decoding segment of the architecture, transposed convolution operations are used to upsample the input and reduce the number of filters from feature maps, starting with the latent representation, until the output has the same dimensionality of the input image. The workflow depicted in Figure 2 corresponds to a real-life use case in which a three-channel image with a resolution of 256x256 pixels is used as the input to a CAE architecture composed of two convolutions plus pooling layer sequences in the encoder and two transposed convolutions in the decoder.

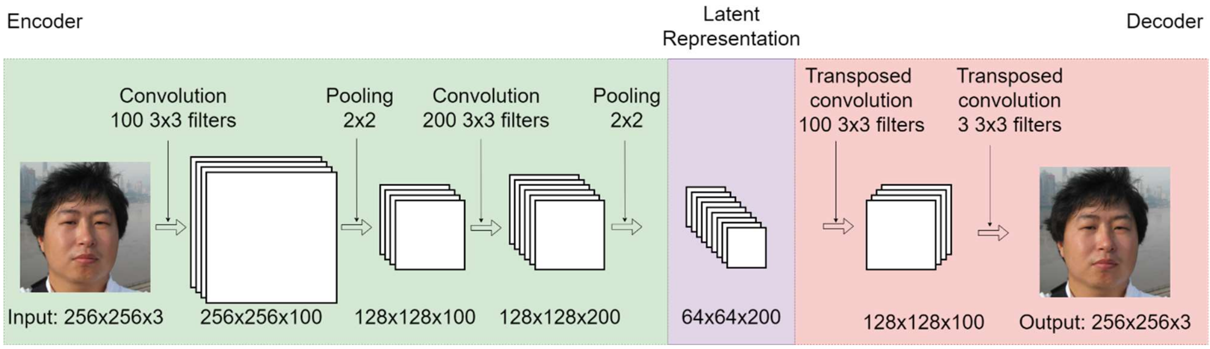

Figure 2: Workflow of a Convolutional Autoencoder

Very Deep Convolutional Networks (VGG)

Very Deep Convolutional Neural Networks are CNN models developed by Simonyan and Zisserman (2014) for ILSVRC (Berg, et al., 2015). Despite their similarity to other CNNs used for classification purposes, the VGG architecture stands out from its counterparts since it replaces singular large kernel-sized filters with multiple consecutive 3x3 kernel-sized filters. These changes make the networks decision function more discriminative and reduce the number of parameters (Simonyan & Zisserman, 2014). The VGG configuration used for this work employs five convolutional blocks and three fully connected layers for a total of sixteen weighted layers, hence the name VGG16 (see Figure 3).

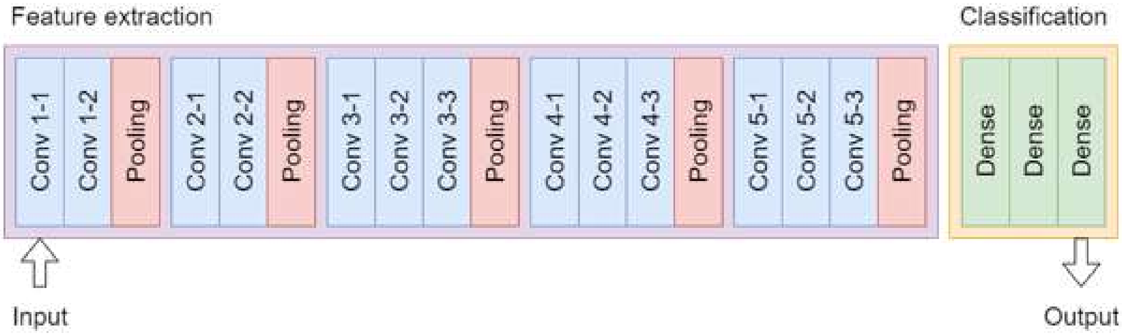

Figure 3: Original VGG16 architecture

The behavior of the VGG16 (and all other VGG models) can be considered standard for CNN classification networks. This implies that the workflow of the network can be divided into two segments: feature extraction and classification. First, in the feature extraction step, the process is identical to the CAEs encoding process (described in the Autoencoders section), in which an image acts as the input to the network and a feature map is created with the most important features of the input. Secondly, the classification step is done by the fully connected layers, which output a vector with the probabilities of the previously generated feature map pertaining to a certain class.

Proposed Model

The proposed model consists of a fully convolutional autoencoder architecture with a VGG16-based encoder and a custom decoder model. Broadly speaking, the objective of this model is to have the ability to receive a one-channel (LAB color space) grayscale image and output a two-channel image of the same size containing the color information. For this, it is necessary to train the model for finding correlations between patterns and colors in grayscale-color pairs of images. The general workflow for the colorization process of this network (see Figure 4), consists of three steps: (1) using the encoding portion of the network to obtain the latent representation of the input grayscale image, (2) employing this latent representation to create the two-channel output containing the color data and (3) stacking the input and output layers to obtain the colorization result. Both, the workflow of the real-life use case shown in Figure 4 and the architecture of the proposed model will be further explained in this section.

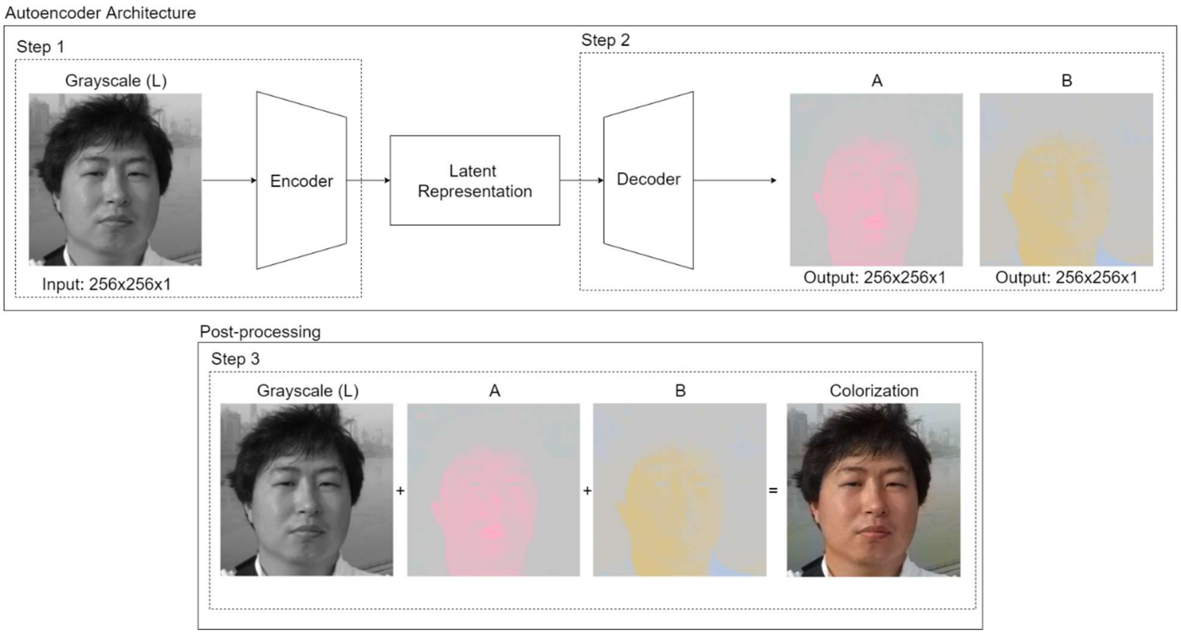

Figure 4: General colorization process for a real-life use case workflow

Encoder Architecture.

For the encoding portion of the network a modified VGG16 architecture was implemented. The modifications done to the original model are: (1) substituting the fixed size 224x224x3 input layer for a new input layer capable of processing one-channel images of any resolution; (2) removing the top fully connected layers; (3) replacing the max pooling layers, for a stride of two in some of the convolution layers; (4) changing the activation function of all convolution layers from ReLU to LeakyReLU; and (5) Adding batch normalization after each convolution layer. The resulting architecture is shown in Figure 5.

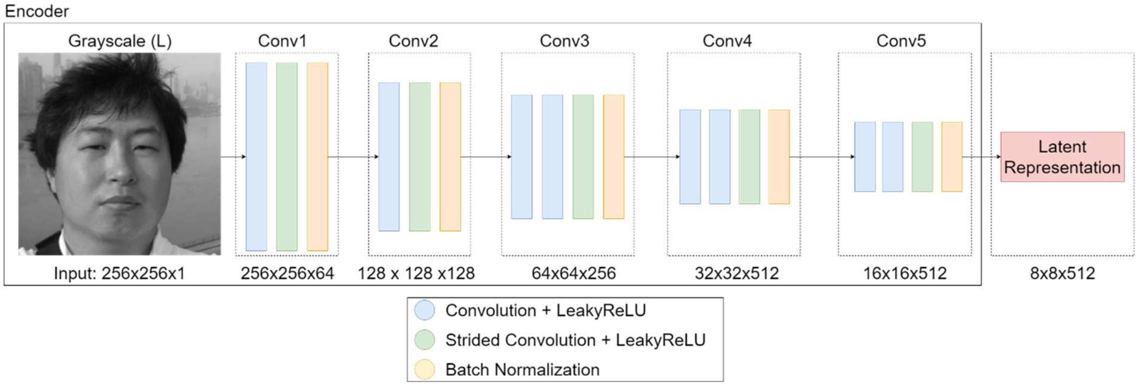

Figure 5: Detailed encoder model and workflow

The causes and effects of these modifications are as follows. First, substituting the fixed size input is critical for the model to be able to process one-channel grayscale images. Second, for adapting the VGG16 architecture to a fully convolutional network architecture, the top fully connected layers were removed; in their work, "Striving for simplicity: The all convolutional net", Brox et al. (2015) proved these CNNs to be more efficient during inference and learning. Third, removing the Pooling layers reduces image distortion. Fourth, changing the loss function to LeakyReLU, effectively avoids the dying ReLU problem (Lu, 2020). And last, the addition of Batch Normalization was proposed by Efros et al. (2017); as it accelerates and stabilizes the learning process of the neural network by normalizing the inputs of each convolutional layer.

The changes implemented to the architecture imply a different encoding/feature extraction workflow than that of the original VGG16 (see Figure 5). On the first two convolutional blocks of the architecture, features are extracted in the form of filters by the first convolution layer; subsequently the second convolution layer continues with feature extraction and reduces image size to half by performing the convolution operation using a stride of two. Lastly, the batch normalization layer normalizes the outputs of the convolutional block it pertains to. This process is then repeated by the remaining three blocks, with the difference that these have two convolution layers preceding the strided convolution operation. Identical to other CAEs, the product of the encoding process is the latent representation which in this case consists of 512 filters with a resolution of eight pixels by eight pixels.

Decoder Architecture.

For the decoder portion of the architecture a custom decoding model was created (see Figure 6). This model mirrors each convolutional block of the encoder with a deconvolutional decoder block. An important difference between this model and a traditional decoder model is that in our decoder architecture bilinear upsampling plus a convolution operation is used to replace the traditional transposed convolution layers. This alternate approach reduces checkerboard artifacts found by Dumoulin et al. (2016). Additionally, to avoid the dying ReLU problem, LeakyReLU activation was used for all convolution layers, apart from the last one, in which hyperbolic tangent (tanh) was implemented. Finally, a key characteristic of this decoder lies within the final convolution layer, which outputs a two-channel image of the same size of the one-channel input.

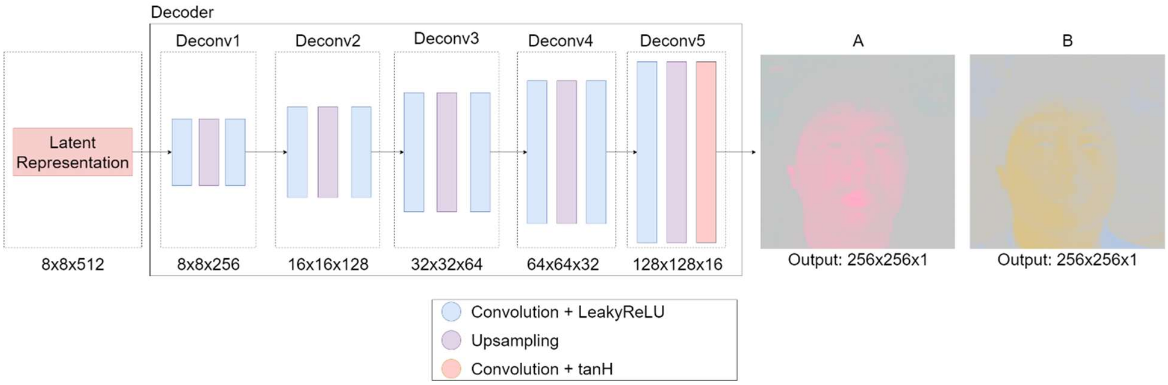

Figure 6: Detailed decoder model and workflow

The behavior of the decoding portion of the network (see Figure 6) is the same for the five deconvolutional blocks that comprise it. First, the initial convolution operation is used for reducing the number of filters to half. Second, bilinear upsampling is applied to increase image size by a factor of two. Third, another convolution operation is applied with the same number of filters as the preceding convolution layer. Since all five blocks follow the same structure, this behavior is maintained throughout the entirety of the network. In simple terms, starting with the latent representation, each block halves the number of filters and up samples image size by a factor of two until the output is a two-channel image with the same dimensions of the input image. Finally, it is important to mention that to obtain the colorized image, it is necessary to stack the input and output layers of the network.

Experimental setup

Data processing.

First, for making both training and processing the images faster, both datasets were resized to a resolution of 256x256 pixels. A lower resolution could make the process faster, but it would impact the expert criterion validation when assessing the quality of the colorized images. Second, the images were transformed from the RGB to the LAB color space. This color space can be understood as a three-axis system in which the L axis contains the lightness information (from black to white), the A axis contains the green-red information (goes from green to red), and the B axis contains the blue-yellow information (goes from blue to yellow).

Despite not being completely necessary, as there are other models which use different color spaces (Cheng et al., 2015), this transformation is very useful since by isolating the lightness/brightness of the image in a single channel (L) the color inference process gets simplified as the network will not have any impact on the lighting of the image and will only have to predict the A and B channels in which the color information resides.

Training and testing partitions.

For training and testing the proposed model, each dataset was empirically divided into two partitions formed by 5000 and 500 images to train and test, respectively.

Model configuration.

All the model parameters were determined empirically. Thus, we used Adam optimization with an initial learning rate of 0.0002 and β_1 coefficient set to 0.5 (as suggested by Efros et al. (2017)). Additionally, the number of iterations (epochs) was tuned to 100 units.

Assessment metrics.

We computed the mean absolute error (MAE/L1), mean squared error (MSE/L2) and peak signal-to-noise ratio (PSNR) metrics to validate the feasibility and robustness of the proposed method in terms of quantitative results. The first and second metrics take the mean of the absolute error and the mean of the squared error, of both the generated and original images to measure how different the images are color-wise. Meanwhile, the third metric measures the quality of the generated images. Since measuring the image colorization performance is a subjective task, we included the expert criterion validation as the qualitative validation to corroborate the methods performance on each dataset.

RESULTS AND DISCUSSION

Quantitative Results

According to the experimental design section, we validated the proposed method on two datasets of random images and the obtained results highlighted the quality of the method, as can be read in Table 1.

Table 1: Average Metrics for the Selected Models

Dataset	MAE (L1)	MSE (L2)	PSNR (dB)
Fruits-360	0.077	0.0152	27.721
FFHQ	0.067	0.0104	26.858

Considering the results shown in Table 1, it is possible to notice the successful performance of the proposed method. The obtained small MAE and MSE scores denote that the method generated samples (images) close to the ground truth. Meanwhile, a relatively high PSNR value indicates that a little error was introduced to the resulting images during the colorization task and thus, decreasing the image colorization quality.

Important clarifications regarding the good numeric performance are: (1) since quadratic error is considered in the loss function of the model, good results were expected and (2) despite these results being good, quantitative metrics are generally not good at predicting human visual response to image quality, therefore metrics are rarely used in image colorization research.

Qualitative Results

For expert criterion validation using the Fruits-360 dataset, it should be noted in Figure 7 some samples of the performance of the model were handpicked. Despite these results not being completely accurate when comparing the color predictions with the ground truth images, they show a subjectively good colorizing performance, leaving no gray borders or uncolored fragments of the images. Additionally, these images are useful for showing that the autoencoder has acquired enough knowledge to distinguish between different fruits and color them accordingly to what it learned. A clear example of this is found in images 2 and 5, which are predicted to have a very similar color palette.

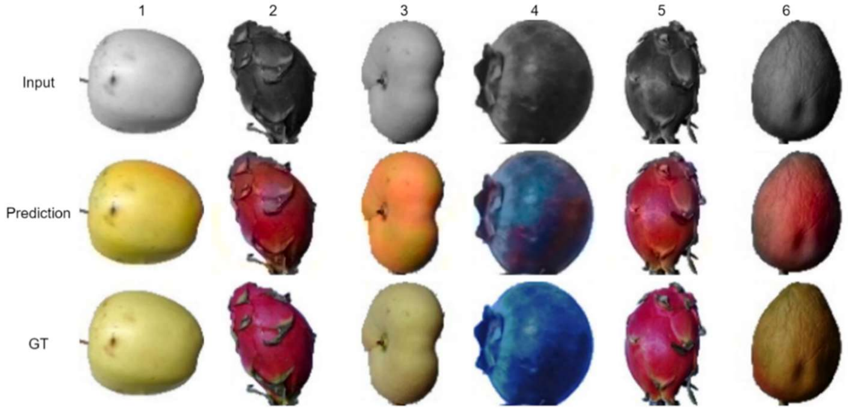

Figure 7: Successful performance examples of colorization for the Fruits-360 dataset

On the other hand, from Figure 8, it is possible to observe better visual colorization results on the FFHQ dataset. Generally, contrasting with the results obtained on the predictions for Fruits-360, a vast improvement can be seen in terms of the predictions not only being colored but also being colored with very similar shades as the ones appearing in their ground truth counterparts; images 1, 4 and 6 are prime examples of this. Another notable characteristic is that, in this set, the model does very well in terms of border detection, keeping the colors within the boundaries of their corresponding objects.

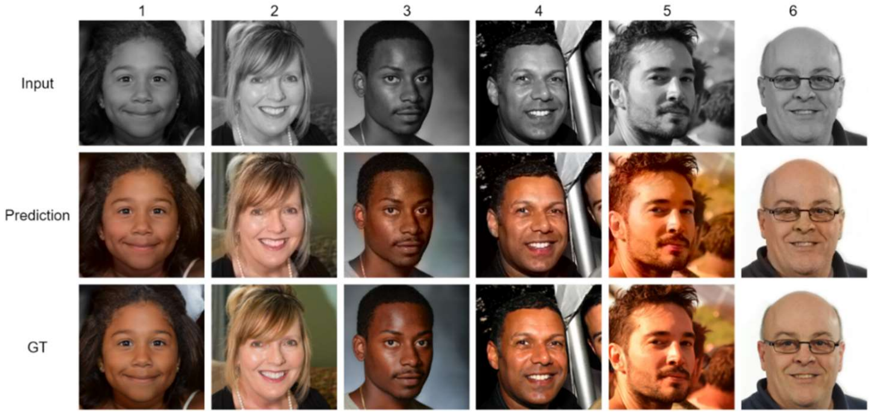

Figure 8: Successful performance examples of colorization for the FFHQ dataset

Additional information about the network's performance can be gathered by examining examples of green colorization. The colorization quality drops found in Figure 9 denote that, although the model was able to learn a general representation of green hues, this representation is not accurate enough to produce convincing colorization results. A possible cause for this behavior could be a shortage of green tonalities within the training set images. This implies that the model has high learning capabilities and that training with more examples of green would suffice to improve the green colorization quality.

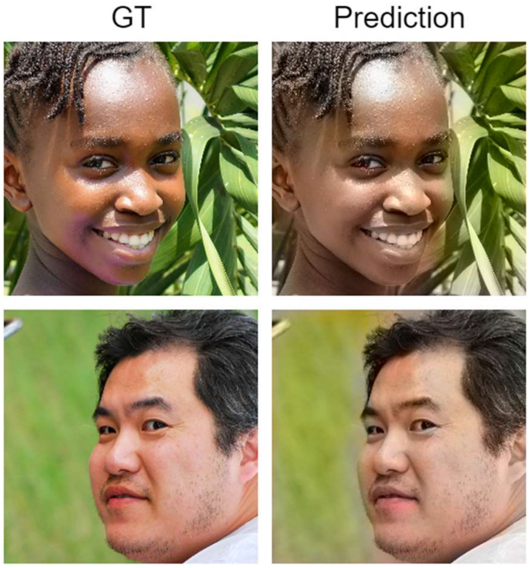

Figure 9: Examples of unexpected green image colorization

Limitations

Instances of vastly limited colorization performance are shown in Figure 10. These occur when the model infers the colors of images depicting uncommon lightning conditions. The prediction on the first row of the figure shows almost no color, which means that since the overexposed image has various tonalities of near-to-white colors, the best strategy to reduce L1 loss is to leave most of the picture uncolored and only colorizing the portions of the image that depicts darker tones. Meanwhile, for the example in the second row, a clear halo of color can be seen where the red light was located. This occurs since the red light turns the contours of the face less well-defined, therefore reducing the model's ability to keep colorization confined within them.

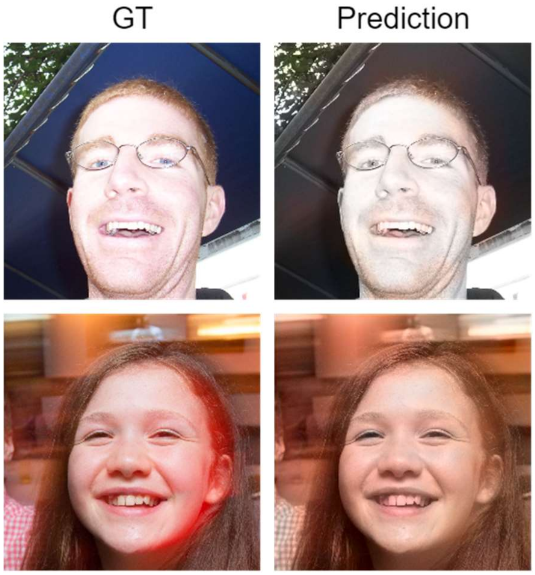

Figure 10: Example of colorization under different lightning conditions

Ultimately, between the performance in FFHQ and Fruits-360, some distinctions can be made. In terms of how close to ground truth the colorizations are, the model trained on FFHQ depicts closer colorization results than its Fruits-360 counterpart, as noted in the higher color saturation and detail in the predicted images seen in Figure 8. This is also supported by the lower L1 and L2 results. Conversely, in terms of border detection, the models performance is better in the Fruits-360 dataset where none of the predictions have color outside of the objects boundaries whereas the images in Figure 9 and Figure 10 do depict this behavior for FFHQ. Finally, in terms of image quality, a slightly higher degradation can be seen in most of the images pertaining to the FFHQ dataset. Although not very notable to expert criterion validation, this difference in quality can also be perceived in the marginally lower results for the PSNR of FFHQ with respect to the Fruits-360 dataset.

CONCLUSIONS AND FUTURE WORK

In this work an autoencoder architecture with a VGG16-based encoder and Deep CNN decoder was implemented to successfully perform image-to-image translation in the form of colorization. The quantitative evaluation shows that the proposed model produces colorized images that are very similar to ground truth in an L1 and L2 sense; furthermore, the PSNR results show that the model can colorize images with a relatively small loss of image quality. Most importantly, the qualitative results show that the network performs the colorization task with mostly good results. Additionally, unexpected green tonalities colorization highlighted the learning capabilities of the proposed model. Nevertheless, some limitations were found when attempting colorization of images depicting uncommon lightning conditions. These leads us to conclude that there are many improvements to be made in the field of image colorization to get better performance as well as being able to generalize the colorizing procedure.

As future work, we plan to include some improvements related to (1) conducting a study on how to overcome the lightning-related limitations found in this work, (2) training the model with more complex datasets to increase its generalization capabilities, (3) integrating the proposed method in a bigger composite model, i.e., GANs architecture to establish a benchmarking analysis, and (4) exploring other quantitative validation metrics to avoid using expert criterion.

REFERENCES

Aila, T., Karras, T., & Laine, S. (2018). A Style-Based Generator Architecture for Generative Adversarial Networks. *CoRR, abs/1812.04948*. Retrieved from http://arxiv.org/abs/1812.04948

Ashikhmin, M., Mueller, K., & Welsh, T. (2002, 7). Transferring Color to Greyscale Images. *ACM Trans. Graph., 21*, 277–280. doi:10.1145/566654.566576

Ballester, C., Raad, L., & Vitoria, P. (2020, 1). ChromaGAN: Adversarial Picture Colorization with Semantic Class Distribution. *arXiv:1907.09837 [cs]*. Retrieved from http://arxiv.org/abs/1907.09837

Berg, A. C., Bernstein, M., Deng, J., Fei-Fei, L., Huang, Z., Karpathy, A., . . . Su, H. (2015, 12). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision, 115*, 211–252. doi:10.1007/s11263-015-0816-y

Brox, T., Dosovitskiy, A., Riedmiller, M., & Springenberg, J. T. (2015). Striving for Simplicity: The All Convolutional Net. *Striving for Simplicity: The All Convolutional Net*.

Chang, H., DiVerdi, S., Finkelstein, A., Fried, O., & Liu, Y. (2015, 7). Palette-based photo recoloring. *ACM Transactions on Graphics, 34*, 1–11. doi:10.1145/2766978

Chang, S., Hasegawa-Johnson, M., Qian, K., Yang, X., & Zhang, Y. (2019). AutoVC: Zero-Shot Voice Style Transfer with Only Autoencoder Loss. In K. Chaudhuri, & R. Salakhutdinov (Ed.), *Proceedings of the 36th International Conference on Machine Learning. 97*, pp. 5210–5219. PMLR. Retrieved from http://proceedings.mlr.press/v97/qian19c.html

Chen, D., Hua, G., Liao, J., Yu, N., & Yuan, L. (2017, 7). StyleBank: An Explicit Representation for Neural Image Style Transfer. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Chen, J.-C., Huang, Y.-C., Tung, Y.-S., Wang, S.-W., & Wu, J.-L. (2005). An adaptive edge detection based colorization algorithm and its applications. *Proceedings of the 13th annual ACM international conference on Multimedia - MULTIMEDIA '05* (p. 351). Hilton: ACM Press. doi:10.1145/1101149.1101223

Cheng, Z., Sheng, B., & Yang, Q. (2015, 12). Deep Colorization. *2015 IEEE International Conference on Computer Vision (ICCV)* (pp. 415–423). Santiago: IEEE. doi:10.1109/ICCV.2015.55

Cohen-Or, D., Liang, L., Luan, Q., Shum, H.-Y., Wen, F., & Xu, Y.-Q. (2007). Natural Image Colorization. In J. Kautz, & S. Pattanaik (Ed.), *Rendering Techniques.* The Eurographics Association. doi:10.2312/EGWR/EGSR07/309-320

Dumoulin, V., Odena, A., & Olah, C. (2016). Deconvolution and Checkerboard Artifacts. *Distill*. doi:10.23915/distill.00003

Ebrahimi, M., Nazeri, K., & Ng, E. (2018). Image Colorization with Generative Adversarial Networks. *arXiv:1803.05400 [cs], 10945*, 85–94. doi:10.1007/978-3-319-94544-6_9

Efros, A. A., Geng, X., Isola, P., Lin, A. S., Yu, T., Zhang, R., & Zhu, J.-Y. (2017, 7). Real-time user-guided image colorization with learned deep priors. *ACM Transactions on Graphics, 36*, 1–11. doi:10.1145/3072959.3073703

Efros, A. A., Isola, P., & Zhang, R. (2016, 10). Colorful Image Colorization. *arXiv:1603.08511 [cs]*. Retrieved from http://arxiv.org/abs/1603.08511

Efros, A. A., Isola, P., Zhou, T., & Zhu, J.-Y. (2017, 7). Image-to-Image Translation with Conditional Adversarial Networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 5967–5976). Honolulu: IEEE. doi:10.1109/CVPR.2017.632

Ehinger, K. A., Hays, J., Oliva, A., Torralba, A., & Xiao, J. (2010). SUN database: Large-scale scene recognition from abbey to zoo. *2010 IEEE Computer Society Conference* *on Computer Vision and Pattern Recognition*, (pp. 3485-3492). doi:10.1109/CVPR.2010.5539970

Hinton, G. E., Rumelhart, D. E., & Williams, R. J. (1985). *Learning internal representations by error propagation.* Tech. rep., California Univ San Diego La Jolla Inst for Cognitive Science.

Larsson, G., Maire, M., & Shakhnarovich, G. (2017, 8). Learning Representations for Automatic Colorization. *arXiv:1603.06668 [cs]*. Retrieved from http://arxiv.org/abs/1603.06668

Levin, A., Lischinski, D., & Weiss, Y. (2004, 6). Colorization using Optimization. *ACM Transactions on Graphics, 23*. doi:10.1145/1015706.1015780

Lu, L. (2020, 6). Dying ReLU and Initialization: Theory and Numerical Examples. *Communications in Computational Physics, 28*, 1671–1706. doi:10.4208/cicp.oa-2020-0165

Muresan, H., & Oltean, M. (2017). Fruit recognition from images using deep learning. *CoRR, abs/1712.00580*. Retrieved from http://arxiv.org/abs/1712.00580

Nash, R., & O'Shea, K. (2015). An Introduction to Convolutional Neural Networks. *CoRR, abs/1511.08458*. Retrieved from http://arxiv.org/abs/1511.08458

Simonyan, K., & Zisserman, A. (2014, 9). Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv 1409.1556.*

Zhang, Y. (2018). A better autoencoder for image: Convolutional autoencoder. *ICONIP17-DCEC. Available online: http://users.cecs.anu.edu.au/~Tom.Gedeon/conf/ABCs2018/paper/ABCs2018_paper_58.pdf (accessed on 23 March 2017).*