IJICIS, Vol.21, No.3, 107-118
DOI: 10.21608/ijicis.2021.92207.1118

International Journal of Intelligent Computing and Information Sciences

https://ijicis.journals.ekb.eg/

Image colorization using Scaled-YOLOv4 detector

M. H. Noaman*
Department of Computer Systems , Faculty of Computer and Information Sciences, Ain Shams University
Cairo, Egypt
Mennatullah.hesham@cis.asu.edu.eg

H. Khaled
Department of Computer Systems , Faculty of Computer and Information Sciences, Ain Shams University
Cairo, Egypt
Heba.khaled@cis.asu.edu.eg

H. M. Faheem
Department of Computer Systems , Faculty of Computer and Information Sciences, Ain Shams University
Cairo, Egypt
Hmfaheem@cis.asu.edu.eg

Received 2021- 8-23; Revised 2021-9-26; Accepted 2021-10-7

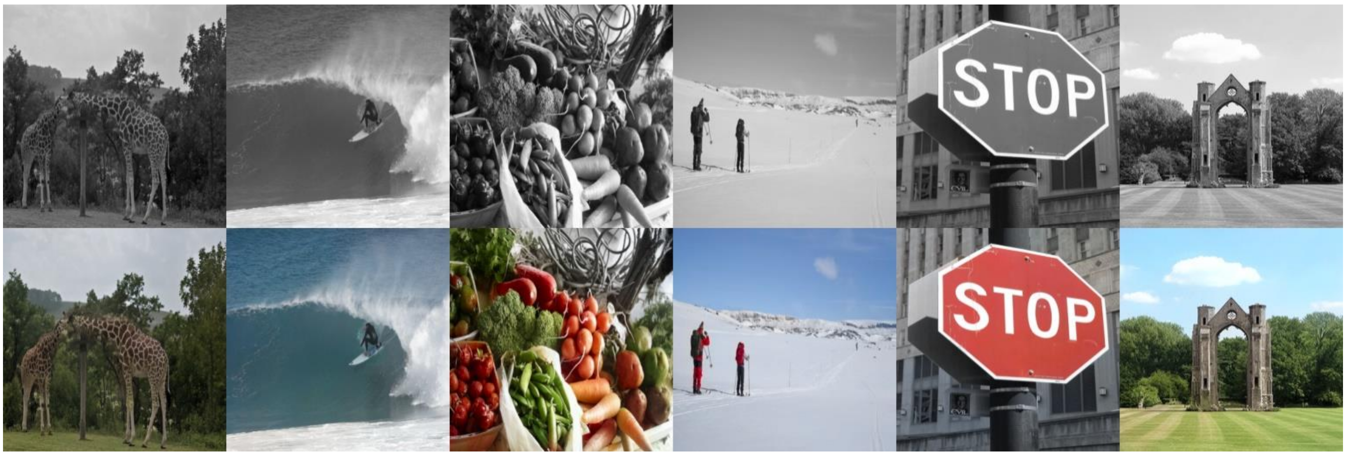


Figure 1 Samples of inputs and colorized images using Scaled-YOLOv4 detector

Abstract: *Image Colorization is the problem of defining colors for grayscale images. Recently many research works have been conducted to propose fully-automatic colorization methods. However, many of these papers failed in colorizing images with multiple objects accurately. This might be because of dealing with the whole multi-object image as a single input. Following the efforts made in the last few years, this paper aims at studying the effect of preceding the image colorization with an object detection phase, such that the colorization will be made for each object individually as well as the full image. After the colorization of each object and the full image, they are fused together to reach a more*

* Corresponding author: M. H. Noaman
Department of Computer Systems , Faculty of Computer and Information Sciences, Ain Shams University, Cairo, Egypt
E-mail address: Mennatullah.hesham@cis.asu.edu.eg

accurate colorized image. In our work, we used a more accurate detector (Scaled-YOLOv4) than that used by the state of the art to increase the quality of the colorization results. Comparing our results to literature, it is found that using Scaled-YOLOv4 increases the Peak signal-to-noise ratio (PSNR) by 2.6%. Results of colorized images with different extensions are compared, and png extension got 5.8% better value of Learned Perceptual Image Patch Similarity (LPIPS) metric than JPEG.

***Keywords**: Image colorization, Computer vision, CNN, Scaled-YOLOv4, Deep learning.*

1. Introduction

One of the most common computer vision problems is image colorization. Image colorization can be defined as the process of converting a grayscale image into a colored image by assigning a color value to each pixel. Many applications can use colorization to perform or enhance their performance, such as image compression, image/video transfer and object detection (as a preprocessing stage).

While it seems to be easy for a human to select potential colors for a grayscale image, the colorization problem witnesses higher difficulty. One reason is the difficulty of imagining several colors for different objects in the same image. Secondly, although some objects may be restricted to extinct colors in reality (grapes may be red, green or yellow, and the sky is blow), some other objects may have large numbers of potential colors, for instance a pen might have many colors in reality. In addition, the surrounding light conditions can influence the color of an object (the sky may have different colors during day and night times). The abovementioned factors increase the difficulty of the problem.

In literature, colorization problem is classified according to different classifications based on the input image type, the number of output colorized images, the methodology used for coloring and the techniques to solve the problem. The most common categorization dependents upon the colorization methods, It is categorized as following: color scribbles, example transfers and automatic direct predictions.

Scribble-based method involve human interference to assign colors to specific areas. The distribution of these colors depends on the assumption that adjacent pixels having the same luminance possess the same colors. Further modifications can be made for the resulting colorized images using other scribbles. One limitation for this method is that it requires relatively longer time from users to colorize an image.

In example-based method, a reference image is used for colorizing the required grayscale image through color information transfer. According to the source of the image, reference images are divided into two categories: user-supplied and web-supplied examples. The results of this type are highly dependent upon the used “reference images”.

Finally, automatic colorization method was designed to avoid the disadvantages of the previous two methods. Since a single reference image cannot contain all possible color combinations for a gray scale image, automatic methods use a massive number of reference images. Progress made in the field of deep learning has significant impact on the quality of the results of these methods.

In this paper, we will introduce an automatic colorization method that uses Scaled-YOLOv4 object detection. This approach is predicted to enhance the colorization of multi-object images. Scaled-YOLOv4 detector is a real-time one-stage detector that achieved the benchmark on COCO-Stuff dataset. This paper is considered the first research work that uses Scaled-YOLOv4 detector in colorization problem. Some samples are shown in Figure 1.

The rest of this paper is organized as follows: section 2 discusses the literature works related to automatic image colorization using natural images. Section 3 shows the experimentation approach while in section 4 results are discussed with datasets and measures are displayed. Finally, section 5 contains the conclusions and future work.

2. Related work

Since this paper addresses the automatic image colorization problem for natural images, this review demonstrates only related research to the automatic colorization. In the recent five years, deep neural networks proved a great success in different fields. Therefore, it is used to solve the image colorization problem; moreover, it proved to be a very good choice. The papers to be discussed are mainly divided into three categories according to the network architecture; Convolutional Neural Networks (CNN), Generative Adversarial Networks (GAN) and variational Autoencoders (VAE).

2.1. Convolutional Neural Networks (CNN)

Cheng et al, [1]was the first to use deep learning in image colorization and proved its effectiveness. Their method carefully analyzed descriptors of informative image features (low-level patch feature, med-level DAISY feature and high-level semantic feature). To improve the colorization quality, they used bilateral filtering. In 2016, Cheng et al.[2] presented a new version of the work, where they adapted the clustering of training images according to the global information features. The new framework outperforms the old one in both colorization quality (lower visible artifacts) and accuracy. The proposed model was shown to be flexible to learn different styles of colorization. In 2017, Cheng et al. [3] added multiple neural networks to their abovementioned work, where the colorization strategy turned to be a two-stage strategy. The first stage is an adaptive color style clustering, whilst the second stage comprises a neural network ensemble.

Deshpande et al. [4] described an image colorization automated method using a LEARCH framework. The model trained a quadratic objective function. They showed how to use a target histogram to get global constraints to improve the results. Larsson et al. [5] used a deep neural network and a color histogram prediction framework utilizing both semantic and low-level representation. They showed the superiority of their results over those of [4]. They also introduced a new benchmark for image colorization using the following metrics (PMSE: Root Mean Square Error & PSNR: Peak Signal-to-Noise Ratio). Lizuka et al. [6] presented the colorization of grayscale image by combining both global and local information. Their proposed model can perform the colorization of an image by transferring the image style. Although this model can optimize an image of any resolution, it has more efficient results with input images of 224x224 pixels. For more optimization, they used both Batch Normalization and Stochastic Gradient Descent (SGD).. They showed that Lab, RBG & YUV color spaces generally got almost similar results, but Lab got better results in some cases.

Zhang et al. [7] proposed the colorization problem as a classification task, based on the VGG classification network, and used class rebalancing at running time to enhance color contrast in the resulting images. To emphasize the influence various loss function, they proposed four different loss functions to train their CNN. They compared the four versions of their model with [8] and [5] to determine the most efficient method. Since the comparison using RMSR failed in capturing visual realism, they took another three different measurements to compare with. These measures were: (Perceptual Realism (AMT): Amazon Mechanical Turk-Semantic Interpretability (VGG Classification)-Raw Accuracy (AuC))

Varga et al. [9] took multiple discriminative and semantic information from VGG-16 and used it to train a two-step CNN architecture. The training occurs without pooling layers to predict the U and V color channels for the adopted YUV color space. Liang et al. [10] used CNN with two loss metric functions to quell the training errors. Stimulated by guided filtering, they presented a refinement scheme to tackle the artifacts that might appear on boundaries. Su et al. [11] suggested a deep colorization model with edge-refined colorization. To detect the value of U and V channels, they respectively trained two neural networks with two loss functions to enhance the precision of the results compared to those of Euclidean distance loss function.

The following research works are considered as attempts to enhance the results of the CNN. Zhang et al. [12] tried to avoid the colorization problems of boundary haziness and color inconsistency in homogeneous regions by adding a post-processing stage to fuse the results of a CNN model and a boundary guided CRF. Baldassarre et al. [13] designed a model that merges high-level features taken from the InceptionResNet-v2 pre-trained model with a deep CNN trained from scratch. The authors claim that their proposed model proved its ability to color high-level images but failed to color small details. Gokhan et al. [14] designed the (ColorCapsNet) model by making use of the segmentation and generative merits of the (CapsNet) to solve the colorization problem. Three modifications are made to the network in order to adapt the problem. First, the CapsNet model is modified to be able to map the grayscale input to the CIE Lab color space output. Second, the model's feature detector part is updated with VGG-19 pretrained model. Finally, Mean Squared Error (MSE) is used as the loss function. Mouzon et al. [15] modeled joint CNN with Variational model to attain the prediction power of the former and the accuracy of the latter. The resulted model can choose a color candidate from a group of colors while regularizing the result.

Inspired by the PixelCNN approach introduced by [16], several researchers tackled the colorization problem using this probabilistic method. Royer et al. [17], designed a colorization model that could attain acceptable stochastic sampling scheme that is able to produce multiple reasonable colorizations for a grayscale image. Guadarrama et al. [18], designed a method to generate multiple versions of colorized images for a single grayscale image. First, a conditional PixelCNN is trained to generate a low-resolution color for the image, then, the generated image and the original grayscale image are considered as inputs into a second CNN. Training the second CNN results in a high-resolution colorful image. In addition to the PixeCNN, Zhao et al. [19], added the object semantic data to enhance colorization. The neural network in their model is designed hierarchically with two branches. The first branch is made for object's type using semantic segmentation loss, while the second one learns object's colors using colorization loss. In addition, they tackled the problem of bleeding of color edges by adding, at inference, a joint bilateral upsampling layer. Zhao et al. [20] again tackled the Pixelated approach and semantic data to the problem; yet they managed to build a pixelated semantic embedding of colors, as well as a pixelated semantic generator. The network is trained to simultaneously optimize semantic segmentation with colorization.

In an attempt to address the problem from a different perspective, Jheng-Wei Su [21] solved the problem by detecting objects firstly, then used two similar colorization networks to extract objects' features and full-image features. After this, the results of the two networks are fused together through a proposed fusion module and a better feature map can be obtained.

2.2 Variational Autoencoders (VAE)

Another attempt to solve the problem that has not witnessed intensive attention in the available literature, is solving the problem using variational autoencoders (VAE). Deshpande et al. [22], used low dimensional embedding for the color fields, designed loss terms for the decoder of the VAE to prevent blurry outputs. Then, they built a conditional model for the multi-modal distribution linking grey image and the embeddings of the color field. The results proved the superiority of the proposed method to the results of the conditional generative adversarial network (cGAN) and those of the conditional variational autoencoder (CVAE) available then.

2.3 Generative Adversarial Networks (GAN)

Regarding the GAN architecture, Yun Cao [23], proposed a conditional GAN to model the distribution of natural colors of an item. The authors enriched their model by multi-layer condition to provide reality and multi-layer noise to boost diversity. Nazeri et al. [24] generalized the colorization method using conditional Deep Convolutional Generative Adversarial Network (DCGAN). They compared results from U-net and GAN and found that the images generated using U-net suffered from blurring effect due to the L2 loss function as in standard CNN. Another result is that mis-colorizing was a repeated problem in images containing high texture details. Sukanta et al. [25] added to the standard conditional GAN an encoder-decoder generator network that uses a classification cross entropy and perceptual losses added to the cGAN original objective function. Vitoria et al. [26] claimed that they enhanced the colorization results by coupling the GAN architecture with semantic information. Their model learns colors by combining the understanding of color and class distribution from semantic and perceptual information.
From the abovementioned review, it is obvious that the field of image colorization is a fertile field of research that still requires efforts in the near future. The potential improvements in the colorization problem are widely spread. One of the interesting points of research is investigating the effect of using object detection in the colorization process. Although the review shows an attempt made by Su et al. 2020, the problem still needs more research efforts for improvement. It is proposed that using more advanced detectors might enhance the colorization results, which is the main contribution of this paper.

3. Approach

3.1. Object Detection

Preceding an image colorization process with object detection might have positive impacts on the colorization results as discussed in Su et al. [21]. For this reason, it might be beneficial to exert some research efforts to investigate the effect of using object detection before image colorization. In their research, Su et al. [21] used Mask R-CNN (**R**egion Based **C**onvolutional **N**eural **N**etworks) as their object detector. However, object detection methods include other methods that outperform Mask R-CNN in measures. One of these outperforming object detection methods is Scaled-YOLOv4 whose comparison with Mask R-CNN according to several measures is shown in table 1. Table 1 shows the comparison based on COCO's standard detection evaluation; Average Precision (AP) at different Intersection over Union (IoU) thresholds. AP^{test} is the mean average for 10 (IoU) thresholds =.50:.05:.95 (primary challenge metric), $\boldsymbol{AP_{50}^{test}}$ is at IoU=.50, $\boldsymbol{AP_{75}^{test}}$ at IoU=.75, AP_S for small objects whose bounding box area is less than 32^2, AP_M for medium objects whose area is between 32^2 and 96^2, and AP_L for large objects with area greater than 96^2.

Table 1 Comparison between Mask R-CNN *[27]* and Scaled-YOLOv4-p7 *[28]* on COCO test-dev.

Object detection	backbone	AP^{test}	$\boldsymbol{AP_{50}^{test}}$	$\boldsymbol{AP_{75}^{test}}$	$\boldsymbol{AP_{S}^{test}}$	$\boldsymbol{AP_{M}^{test}}$	$\boldsymbol{AP_{L}^{test}}$
Mask R-CNN	ResNeXt-101-FPN	39.8	62.3	43.4	22.1	43.2	51.2
Scaled-YOLOv4-P7	CSP-P7	55.4	73.3	60.7	38.1	59.5	67.4

We also detected bounding box B_i using Scaled-YOLOv4-P7 for each object then cropped the corresponding object in the grayscale image X_i and ground truth colored image Y_i^{GT} from X and Y^{GT}

respectively and resized to resolution of 256x256 the cropped images due to the image colorization network input size.

3.2. Image colorization and Fusion module

In this work, we adopted the same image colorization and fusion models but changed the object detection method as the aim of this paper is to investigate the effect of the object detection method itself on the colorization output. The colorization network is used twice; firstly, it is used to colorize instance images from Scaled-YOLOv4-P7 detector while it is secondly used to colorize the full image. Then, we resized the instance features to match the size of the full image by padding with zero.

4. Experiment

This Section illustrates the experimental-related points including the used datasets and qualitative and quantitative measurements. The environment specifications are summarized. Finally, a discussion is made to illustrate the results.

4.1. Datasets

This paper uses three datasets to compare the effect of using Scaled-YOLOv4-P7 on the results of the colorization model. The first dataset is Place205 [29] which is composed of 20,500 testing images with resolution 256*256 from 205 place categories. This dataset was not used in any training for each of the compared methods. The second dataset is ImageNet ctest 10k list provided by [5]. The last dataset is COCO-Stuff [30] validation set which contains 5000 natural images with various objects defined.

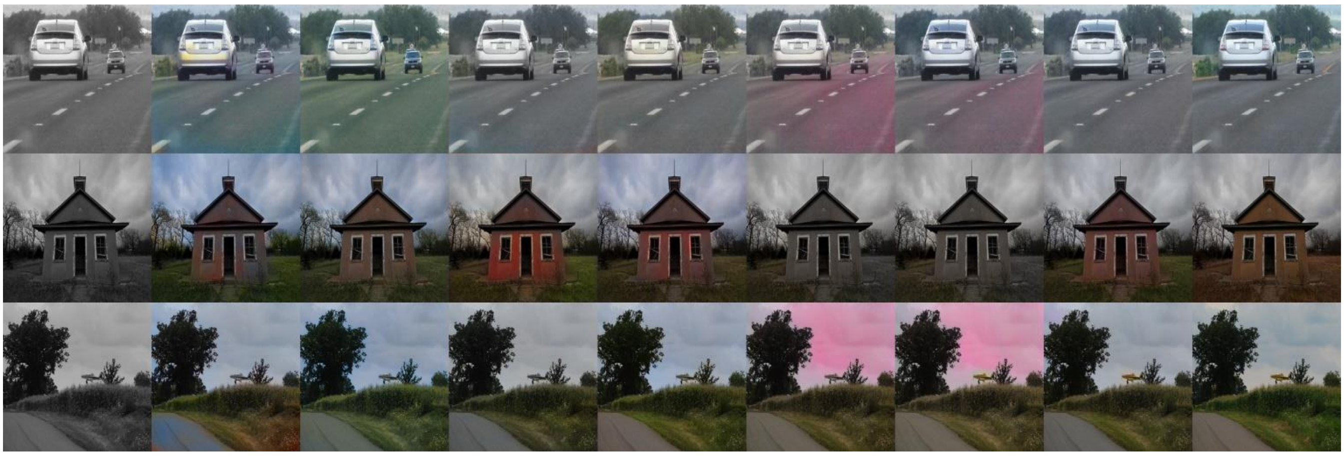
4.2. Measurements

4.2.1. Quantitative comparisons

This paper uses three evaluation metrics to compare the methods. They are: Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Metric (SSIM) and Learned Perceptual Image Patch Similarity (LPIPS) metric [117]. As for SSIM, the values computed from individual channels are averaged with window size 7 as a default value. For LPIPS, results are computed using version 0.1 with VGG backbone and staff false. Finally, For ImageNet ctest the environment could not afford calculating the LPIPS metric for the largest 55 images, so they were not included in the average results.

4.2.2. Qualitative comparisons

The results are compared to the state-of-the-art automatic colorization papers and the popular colorization online project Deoldify [31] as shown in Figure. 2, Figure.3 and Figure. 4 The selected images for comparisons are chosen from the three used datasets. Figure. 5 shows some failed colorizing images for both Scaled-YOLOv4 and Mask R-CNN.

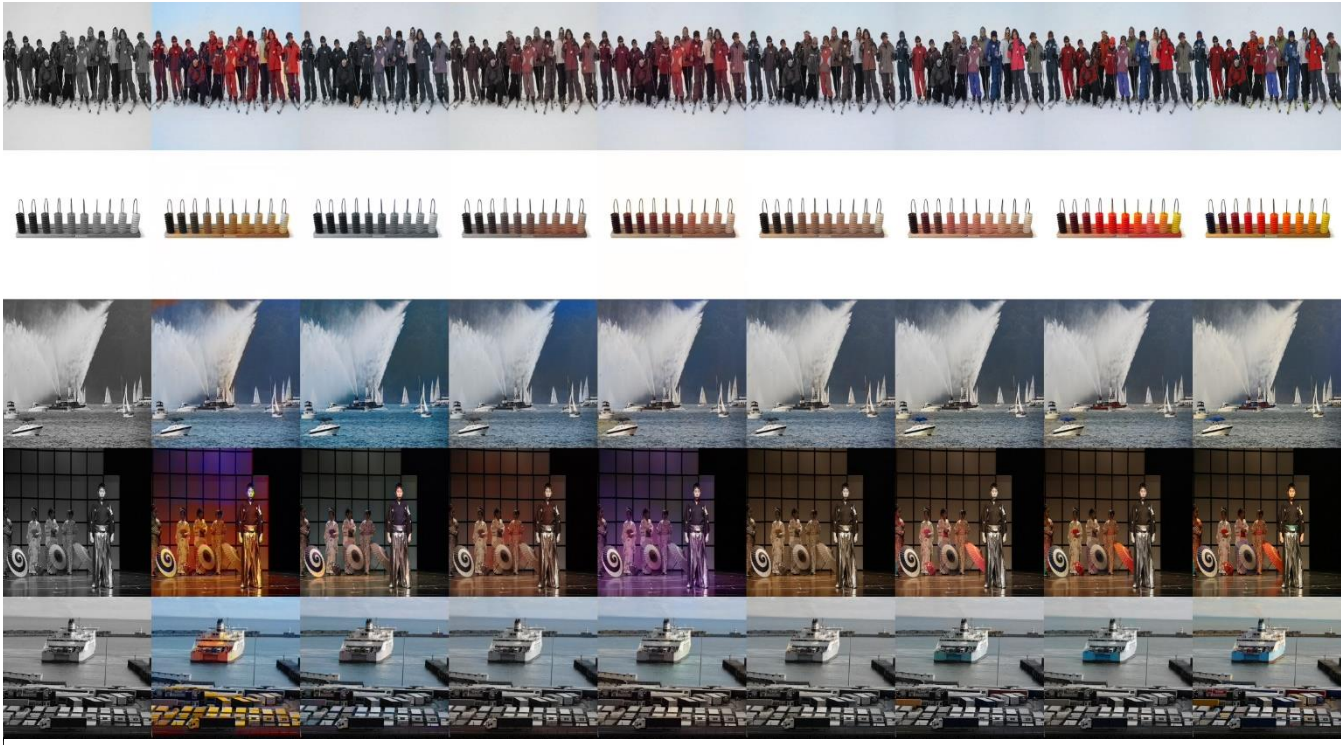


(a) Input (b) Zhang et al. [7] (c) Deoldify [31] (d) Larsson et al. [5] (e) Lizuka et al.[6] (f) Zhang et al. [32] (g) Jheng et al.[21] (h) Scaled-YOLOv4-p7 (our) (i) GT

Figure 2 Some Samples form Places205 dataset for coloized images

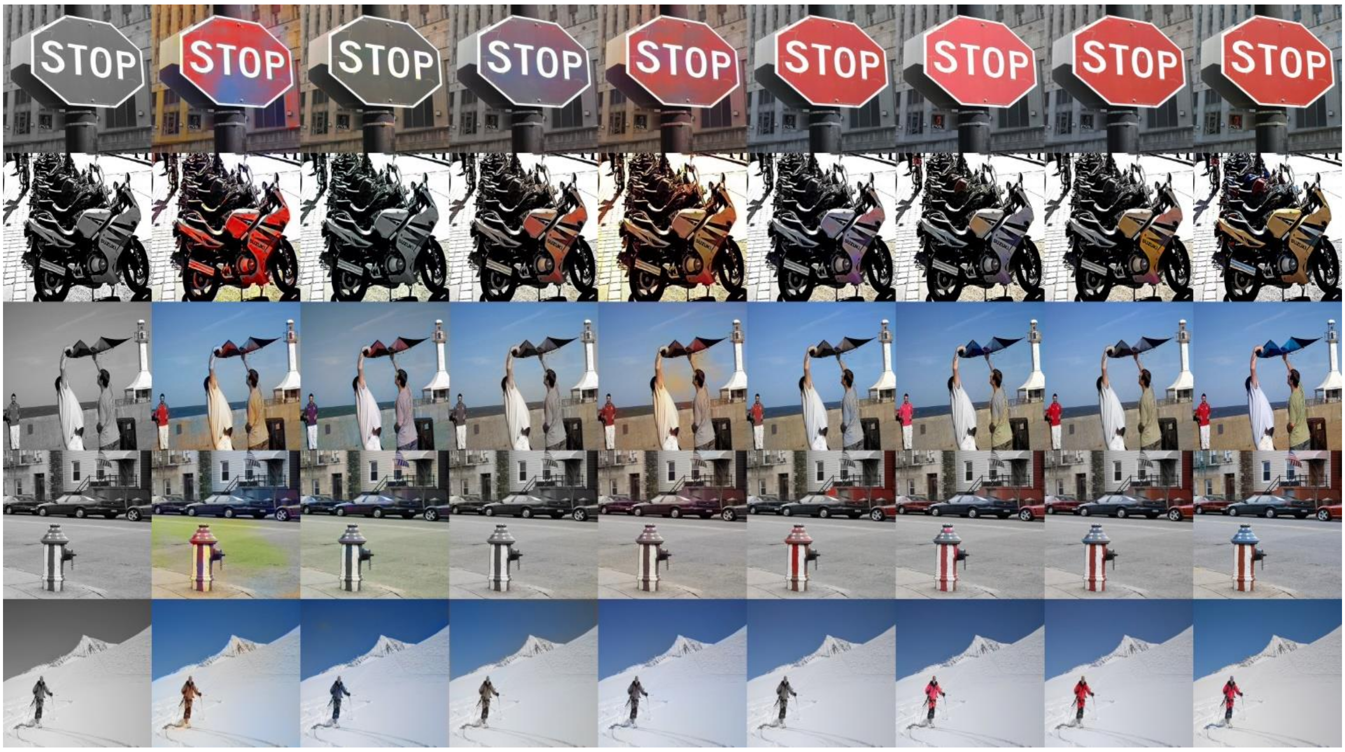
4.3. Environment specifications

All the testing experiments were done on Ubuntu 16.04 LTS 64-bit, memory 15.5 GiB, Processor Intel® Core™ i7-8750H CPU @ 2.20GHz × 12 and single GeForce GTX 1060/PCIe/SSE2 6G GPU.



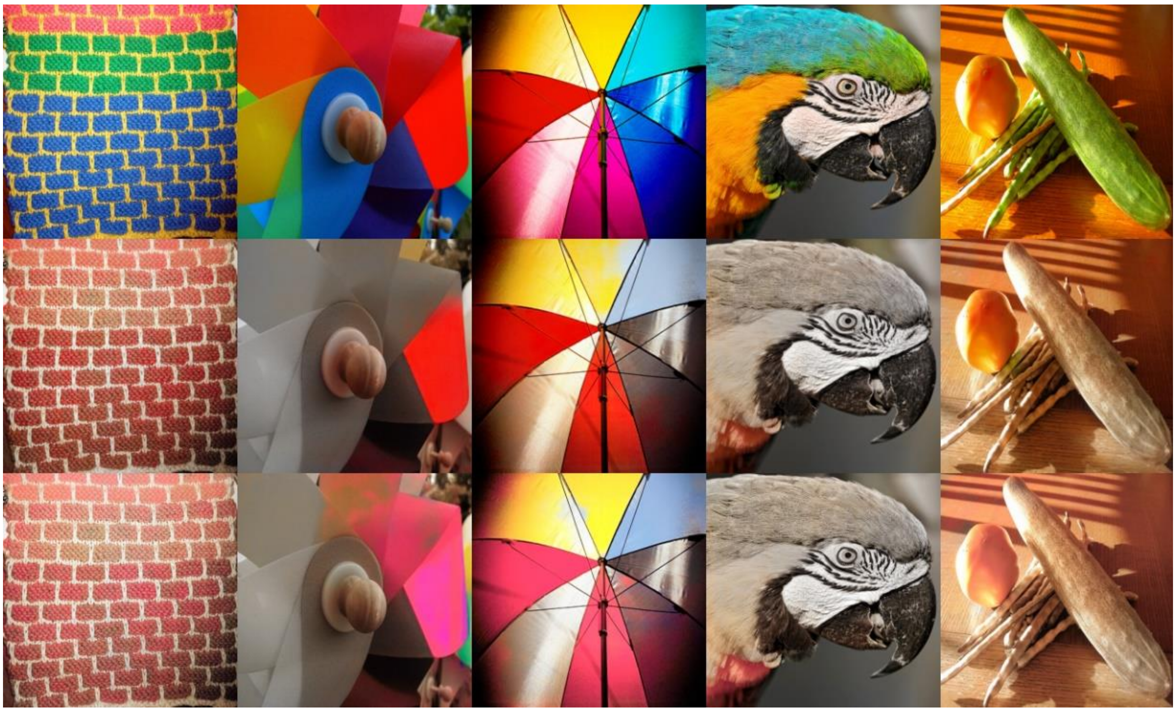
(a) Input | (b) Zhang et al. [7] | (c) Deoldify [31] | (d) Larsson et al. [5] | (e) Lizuka et al.[6] | (f) Zhang et al. [32] | (g) Jheng et al.[21] | (h) Scaled-YOLOv4-p7 (our) | (i) GT

Figure 4 Some Samples form COCO dataset for coloized images



(a) Input | (b) Zhang et al. [7] | (c) Deoldify [31] | (d) Larsson et al. [5] | (e) Lizuka et al.[6] | (f) Zhang et al. [32] | (g) Jheng et al.[21] | (h) Scaled-YOLOv4-p7 (our) | (i) GT

Figure 3 Some sampels form Imagenet dataset ctest list



(a) Grund-truth

(b) Scaled-YOLOv4 (ours)

(c) Jheng et al.[21]

Figure 5 Some sample of failer case.

4.4. Results and discussion

This section discusses the results of the experiments made to investigate the effect of using Scaled-YOLOv4-P7 object detector on the solution of image colorization process. The images of all references were all downloaded from [21] project's website, and all the results shown in comparisons were computed using our environment. Although the extensions of ImageNet images are all JPEG with different sizes, it is found that the output images of references [5]–[7], [31] have png extensions and fixed size of 256*256, while the output images of references [21], [32] have the same extensions (JPEG) and sizes of the ground truth images. The same results are found with COCO-Stuff dataset but with ground truth extensions of jpg. In order to be able to compare the output images with the ground truth images, they must have the same size. To guarantee that they have the same size, we resized the output images of all compared methods with the same approach. As for Place205, both ground truth and output images have the same size and extension, which made their comparison more accurate.

As show in Table 2, Using Scaled-YOLOv4-P7 detector in colorization produces better results than Mask R-CNN detector for metric PSNR and SSIM on Places205 dataset. For PSNR metric, our results outperform the results of all the papers in the comparison. Regarding the SSIM metric, we achieved better results of 0.951 over Jheng et al. [21] and equal results to Larsson et al. [5]. Finally, for LPIPS metric, our results are better than all references except for Jheng et al. [21].

Table 2 Quantitative comparison for Places205 validation split. First block models trained on Imagenet dataset. Second block models finetuned on the COCO-Stuff training set

Method	LPIPS↓	PSNR↑	SSIM↑
Lizuka et al. [6]	0.146	25.581	0.95
Larsson et al. [5]	0.161	25.722	**0.951**
Zhang et al. [7]	0.205	22.581	0.921
Deoldify [31]	0.161	23.983	0.939
Zhang et al. [32]	0.153	25.72	0.947
Jheng et al. [21]	**0.125**	26.725	0.95
Scaled-YOLOv4-p7 (our)	0.133	**26.903**	**0.951**

Comparing the previously discussed methods using both ImageNet and COCO-Stuff datasets, it was found that some papers produced the colorized images with png extensions, while others produced their images with groundtruth extensions. For this reason, we managed to produce our colorized image with both extensions to investigate if the extension has an impact on the results.

Table 3 shows the results of the methods comparisons using ImageNet and COCO-Stuff. it is obvious that our results with png is the best for both datasets in all metrics. On the other hand, when using the groundtruth extension, we achieved better results in all metrics; LPIPS, PSNR and SSIM.

Table 3 Quantitative comparison for Imagenet and COCO-Stuff splits. First block models trained on Imagenet dataset. Second block models finetuned on the COCO-stuff training set

Method	ImageNet ctest 10k				COCO-Stuff validation split			
	Image extinction	LPIPS↓	PSNR↑	SSIM↑	Image extinction	LPIPS↓	PSNR↑	SSIM↑
Ground truth	JPEG				jpg			
Lizuka et al. [6]	png	0.207	23.122	0.908	png	0.194	23.347	0.915
Larsson et al. [5]	png	0.193	24.109	0.913	png	0.189	24.108	0.920
Zhang et al. [7]	png	0.240	21.630	0.892	png	0.236	21.639	0.898
Deoldify [31]	png	0.193	23.586	0.908	png	0.186	23.707	0.916
Zhang et al. [32]	JPEG	0.161	25.932	0.921	jpg	0.152	26.368	0.928
Jheng et al. [21]	JPEG	0.149	25.757	0.921	jpg	0.137	26.372	0.929
Scaled-YOLOv4-p7 (our)	JPEG	0.155	26.345	0.923	jpg	0.138	27.248	0.933
Scaled-YOLOv4-p7 (our)	png	**0.146**	**26.433**	**0.926**	png	**0.130**	**27.439**	**0.935**

5. Conclusions

The aim of this paper is to study the effect of using object detection in image colorization. We used Scaled-YOLOv4-P7 instead of Mask R-CNN detectors before image colorization. Scaled-YOLOv4 was used to study the effect of using a real-time detector on the colorization results and due to its superiority in comparison with Mask R-CNN on COCO dataset. Our results proved that using object detection improves the quality of the colorization results. Furthermore, improving the object detection method or using a better object detection method also increases the quality of the colorization results. Finally, producing the colorized images in png extensions produces better colorization qualities. As for future work, it might be beneficial to use other object detectors with colorization, in addition, segmentation techniques might improve the results if used with object detection and colorization.

References

[1] Z. Cheng, Q. Yang, and B. Sheng, “Deep colorization,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, vol. 2015 Inter, pp. 415–423, doi: 10.1109/ICCV.2015.55.

[2] Z. Cheng, Q. Yang, and B. Sheng, “Deep colorization,” *arXiv e-prints*, p. arXiv-1605, 2016.

[3] Z. Cheng, Q. Yang, and B. Sheng, “Colorization Using Neural Network Ensemble,” *IEEE Trans. Image Process.*, vol. 26, no. 11, pp. 5491–5505, Nov. 2017, doi: 10.1109/TIP.2017.2740620.

[4] A. Deshpande, J. Rock, and D. Forsyth, “Learning large-scale automatic image colorization,” *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2015 Inter, pp. 567–575, 2015, doi: 10.1109/ICCV.2015.72.

[5] G. Larsson, M. Maire, and G. Shakhnarovich, “Learning representations for automatic colorization,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9908 LNCS, pp. 577–593, 2016, doi: 10.1007/978-3-319-46493-0_35.

[6] S. Iizuka, E. Simo-Serra, and H. Ishikawa, “Let there be color!: Joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification,” *ACM Trans. Graph.*, vol. 35, no. 4, pp. 1–11, 2016, doi: 10.1145/2897824.2925974.

[7] R. Zhang, P. Isola, and A. A. Efros, “Colorful image colorization,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9907 LNCS, pp. 649–666, 2016, doi: 10.1007/978-3-319-46487-9_40.

[8] R. Dahl, “Automatic Colorization,” *Http://Tinyclouds.Org/Colorize/*, no. January, pp. 1–13, 2016, [Online]. Available: http://tinyclouds.org/colorize/.

[9] D. Varga and T. Sziranyi, “Fully automatic image colorization based on Convolutional Neural Network,” in *Fully automatic image colorization based on Convolutional Neural Network*, Jan. 2016, vol. 0, pp. 3691–3696, doi: 10.1109/ICPR.2016.7900208.

[10] X. Liang, Z. Su, Y. Xiao, J. Guo, and X. Luo, “Deep patch-wise colorization model for grayscale images,” *SA 2016 - SIGGRAPH ASIA 2016 Tech. Briefs*, 2016, doi: 10.1145/3005358.3005375.

[11] Z. Su, X. Liang, J. Guo, C. Gao, and X. Luo, “An edge-refined vectorized deep colorization model for grayscale-to-color images,” *Neurocomputing*, vol. 311, pp. 305–315, Oct. 2018, doi: 10.1016/j.neucom.2018.05.082.

[12] W. Zhang, C. W. Fang, and G. Bin Li, “Automatic Colorization with Improved Spatial Coherence and Boundary Localization,” *J. Comput. Sci. Technol.*, vol. 32, no. 3, pp. 494–506, 2017, doi: 10.1007/s11390-017-1739-6.

[13] F. Baldassarre *et al.*, “Deep koalarization: Image colorization using CNNs and inception-resnet-v2,” *arXiv*, no. June 2017, pp. 1–12, 2017.

[14] G. Ozbulak, G. Gökhan¨ Gökhan¨ozbulak, and G. Ozbulak, “Image colorization by capsule networks,” *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work.*, vol. 2019-June, pp. 2150–2158, 2019, doi: 10.1109/CVPRW.2019.00268.

[15] T. Mouzon, F. Pierre, M.-O. O. J. O. Berger, M.-O. O. J. O. Berger, and V. Cnn, “Joint CNN and Variational Model for Fully-Automatic Image Colorization,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2019, vol. 11603 LNCS, pp. 535–546, doi: 10.1007/978-3-030-22368-7_42.

[16] A. Van Den *et al.*, “Conditional Image Generation with PixelCNN Decoders.”

[17] A. Royer, A. Kolesnikov, and C. H. Lampert, “Probabilistic image colorization,” *arXiv*, pp. 1–15, 2017.

[18] S. Guadarrama *et al.*, “Pixcolor: Pixel recursive colorization,” *arXiv*, pp. 1–17, 2017.

[19] J. Zhao, L. Liu, C. G. M. M. Snoek, J. Han, and L. Shao, "Pixel-level semantics guided image colorization," *arXiv*, pp. 1–12, 2018.
[20] J. Zhao *et al.*, "Pixelated Semantic Colorization," *Int. J. Comput. Vis.*, vol. 128, no. 4, pp. 818–834, 2019, doi: 10.1007/s11263-019-01271-4.
[21] J.-W. Su, H.-K. Chu, and J.-B. Huang, "Instance-aware Image Colorization." Accessed: Apr. 30, 2021. [Online]. Available: https://ericsujw.github.io/InstColorization.
[22] A. Deshpande, J. Lu, M.-C. C. Yeh, M. J. Chong, and D. Forsyth, "Learning diverse image colorization," *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, no. Section 4, pp. 2877–2885, 2017, doi: 10.1109/CVPR.2017.307.
[23] Y. Cao, Z. Zhou, W. Zhang, and Y. Yu, "Unsupervised Diverse Colorization via Generative Adversarial Networks," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10534 LNAI, pp. 151–166, 2017, doi: 10.1007/978-3-319-71249-9_10.
[24] K. Nazeri, E. Ng, and M. Ebrahimi, "Image colorization using generative adversarial networks," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018, vol. 10945 LNCS, pp. 85–94, doi: 10.1007/978-3-319-94544-6_9.
[25] S. S. Halder, K. De, and P. P. Roy, "Perceptual Conditional Generative Adversarial Networks for End-to-End Image Colourization," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11362 LNCS, pp. 269–283, Dec. 2018, doi: 10.1007/978-3-030-20890-5_18.
[26] P. Vitoria, L. Raad, and C. Ballester, "ChromaGAN: Adversarial picture colorization with semantic class distribution," *Proc. - 2020 IEEE Winter Conf. Appl. Comput. Vision, WACV 2020*, pp. 2434–2443, Jul. 2020, doi: 10.1109/WACV45572.2020.9093389.
[27] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN." Accessed: Jun. 08, 2021. [Online]. Available: https://github.com/.
[28] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Scaled-YOLOv4: Scaling Cross Stage Partial Network," Nov. 2020, Accessed: Aug. 07, 2021. [Online]. Available: https://arxiv.org/abs/2011.08036v2.
[29] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning Deep Features for Scene Recognition using Places Database," *Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, Accessed: Aug. 11, 2021. [Online]. Available: http://places.csail.mit.edu.
[30] H. Caesar, J. Uijlings, and V. Ferrari, "COCO-Stuff: Thing and Stuff Classes in Context," 2018. Accessed: May 31, 2021. [Online]. Available: http://calvin.inf.ed.ac.uk/datasets/coco-stuff.
[31] "jantic/DeOldify: A Deep Learning based project for colorizing and restoring old images (and video!)." https://github.com/jantic/DeOldify (accessed Aug. 07, 2021).
[32] R. Zhang *et al.*, "Real-time user-guided image colorization with learned deep priors," in *ACM Transactions on Graphics*, 2017, vol. 36, no. 4, doi: 10.1145/3072959.3073703.