Color quantization in transfer learning and noisy scenarios: an empirical analysis using convolutional networks

Tiago S. Nazare*‡, Gabriel B. Paranhos da Costa*, Rodrigo F. de Mello*, Moacir A. Ponti*
*ICMC, University of São Paulo, São Carlos, Brazil.
‡Data Science Team, Itaú-Unibanco, São Paulo, Brazil.
Email: tiagosn@usp.br, gbpcosta@usp.br, mello@icmc.usp.br, moacir@icmc.usp.br

***Abstract*—Transfer learning is seen as one of the most promising areas of machine learning. Lately, features from pre-trained models have been used to achieve state-of-the-art results in several machine vision problems. Those models are usually employed when the problem of interest does not have enough supervised examples to support the network training from scratch. Most applications use networks pre-trained on noise-free RGB image datasets, what is observed even when the target domain counts on grayscale images or when data is degraded by noise. In this paper, we evaluate the use of Convolutional Neural Networks (CNNs) on such transfer learning scenarios and the impact of using RGB trained networks on grayscale image tasks. Our results confirm that the use of networks trained using colored images on grayscale tasks hinders the overall performance when compared to a similar network trained on a quantized version of the original dataset. Results also show that higher quantization levels (resulting in less colors) increase the robustness of CNN features in the presence of noise.**

I. Introduction

Deep Learning has become very popular along the last years, especially in the context of computer vision applications [1]. Convolutional Neural Networks (CNNs) have achieved great performance in several complex machine vision applications (e.g. image classification and object recognition) [2]. Nevertheless, the training of such models requires large quantities of labeled data, which might not be available in some scenarios [3]. In that context, transfer learning has been used to address such issue by taking advantage of using models trained on other domains, which are either useful as starting point for training or to support feature extraction [4].

Lately, pre-trained networks were even employed in domains with low-quality images, such as surveillance videos [5] and heavily compressed images [6], performing significantly well. However, those studies do not investigate the effects of using CNNs trained on RGB domains to extract features from grayscale/noisy images. Recent studies pointed out that changes in image quality can hamper the performance of pre-trained models [7], [8], but no mention to the use of RGB-based networks on grayscale domains is given. Motivated by this gap, we designed an experimental setup to test the following hypothesis:

Any opinions, findings, and conclusions expressed in this manuscript are those of the authors and do not necessarily reflect the views, official policy or position of the Itaú-Unibanco, FAPESP and CNPq.

1) Instead of applying transfer learning using some RGB model on a grayscale dataset – *given data is not enough to proceed with the CNN trainining from scratch* – it is better to train a new model using a grayscale version of the original set;
2) CNN features learned from grayscale images with a reduced number of possible colors are more robust to noise.

Our experimental analysis shows that changing the quantization level of images decreases the performance of a model even in the same dataset that it was trained beforehand. Moreover, models trained in grayscale versions of datasets seem to be better suited to serve as feature extractors/transfer learning for other grayscale datasets. Finally, the results also show that models trained in quantized datasets are more robust to noise.

II. Related work

Due to the performance achieved in transfer learning scenarios [4], pre-trained CNNs became a very appealing resource when dealing with applications where labeled data is not widely available for training. Such CNNs are usually trained on noise-free RGB-image datasets (e.g. ImageNet [9] and CIFAR-10 [10]). In spite of being trained using good-quality images, those networks tend to perform well even when applied to grayscale [5] or noisy [2] domains. On the other hand, recent studies have shown that changes in image quality can affect image classification results [7], [8].

Dodge and Karam [7] showed that several state-of-the-art CNNs, such as VGG [11], lack in terms of resilience to certain types of changes, such as: blur, noise, contrast and image compression. This particular study considers the entire model (convolutional and dense layers) and it was conducted using models trained on the original dataset to classify distorted versions of the test set (same domain).

In [8], the robustness of several CNN architectures was assessed with regards to Gaussian and salt & pepper noise. This study pointed out that training networks with noisy images makes them more robust even to other types of noise. Nonetheless, all their experiments were conducted within the same dataset, that is, transfer learning scenarios were not considered.

This kind of resilience is also studied regarding hand-crafted features. For instance, [12] investigated the stability of some LBP [13] variants to noise, while [14] performed experiments to evaluate the robustness of LBP and HOG [15] descriptors to noise.

Differently from the aforementioned studies, we devoted our efforts to study the effects of using models trained in a certain dataset – composed only of RGB images – as classifiers or feature extractors for other datasets with quantized grayscale images or noisy images. Hence, we focus on the analysis of the transfer learning capabilities of the learned features, instead of studying them within the same dataset.

III. Experimental setup

In order to test our hypotheses, we designed a experimental setup composed of three scenarios. In the **first experiment**, we trained 21 different models using the CIFAR-10 dataset, as follows:

- We selected three architectures (Simple CNN, Base Model CNN and ResNet-20) and trained 7 different models for each architecture;
- Each of the seven models based on a certain architecture was trained using a different version of the CIFAR-10 dataset. The first model was trained using the original (RGB) training set, while the others used grayscale versions of the training set with 256, 128, 64, 32, 16 and 8 gray levels (quantization), respectively.

The trained models were then used to evaluate every version (RGB and quantized) of the CIFAR-10 test set. With this baseline experiment, we aim at establishing how dependent a model is to the color configuration of its training set.

In the **second experiment**, we analyzed the robustness of the CNNs features – learned by the 21 models trained during the first experiment – with regards to color quantization in transfer learning scenarios. In order to do so, we used four datasets: CIFAR-10, CIFAR-100 fine, CIFAR-100 coarse and Fashion-MNIST. In our setup, we took every pair *(dataset, CNN architecture)* and proceeded as follows:

- Similar to what was performed during the first experiment, we created seven versions of each dataset: one RGB and six grayscale versions with 256, 128, 64, 32, 16 and 8 colors;
- Each trained model was used to extract features for every version of every dataset, resulting in 49 different feature sets for each dataset (196 features sets in total);
- For each feature set, we trained a logistic regression classifier[1] on the training set and evaluated it on the test set.

Figure 1 illustrates the pipeline used in the second experiment, which allows us to have a better understanding of the robustness of features as the quantization changes, this when compared to the first experiment. This is due to the fact that – by using the original CNN as feature extractor (convolutional part) and replacing the classifier (dense part) – we are directly evaluating the generated feature space.

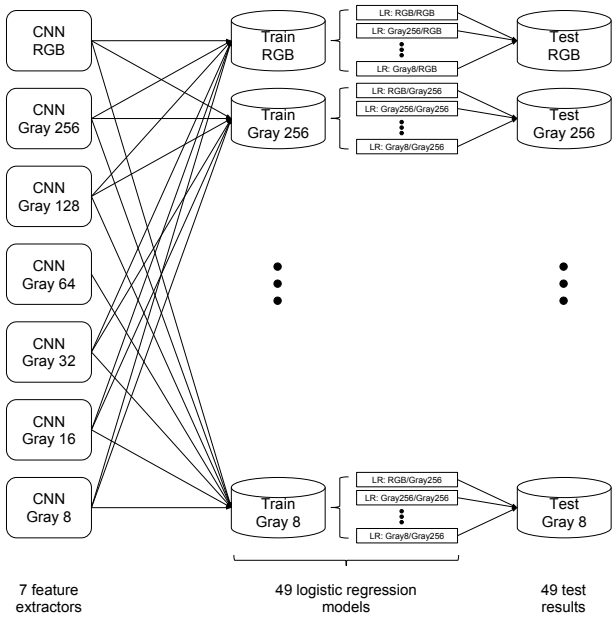

Fig. 1. Diagram illustrating the pipeline used in the second experiment. For each pair *(dataset version, feature extraction CNN)*, a different logistic regression model is learned from the training set. Then, this classification model is assessed using a test set with the same quantization employed in training.

Finally, the **third experiment** uses the same models trained during the first experiment, however to investigate the robustness the learned features have in noisy scenarios. To this end, we employed the 21 models obtained from experiment one to extract features from the original Fashion-MNIST training set (grayscale images with 256 possible colors). Next, like in experiment two, we trained one logistic regression for each feature set version. These logistic regression models were then evaluated on three noisy versions of the Fashion-MNIST test set, affected by Gaussian noise with standard deviations equal to 10, 20 and 30, respectively. Thus, differently from experiment two, in this experiment the original Fashion-MNIST test set is always used (grayscale images with 256 possible colors), only changing the noise level. Also, it is important to notice that the Gaussian noise is only applied to the test set and that the only difference between the learned classification models comes from the feature space learned by the CNNs due to the different color schemes on their training sets (CIFAR-10: RGB and quantizatized to 256, 128, 64, 32, 16 and 8 gray levels).

In a nutshell, experiments one and two focused on testing the first hypothesis, while experiment three aimed to test the second hypothesis. Consequently, having such results, we want to shed some light on whether training a new network, using some quantized version of the original dataset, is beneficial

[1]We employed the `SGDClassifier` with log loss from `scikit-learn`. This is equivalent to training a logistic regression using stochastic gradient descent (`http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html`).

when performing transfer learning from RGB to grayscale datasets (with and without the presence of noise).

A. Image quantization

Image quantization focuses on reducing the number of possible color values of an image. This process attempts to reduce the amount of memory required to represent an image, while maintaining a visual representation that is as close as possible to the original image. As pointed out in [16], such approach can greatly reduce the feature vector size (for hand-crafted features) and even improve classification accuracies.

There are several ways of quantizing images, as explained in [16]. In our experiment, we always quantized images by uniformly grouping colors. For example, if we wish to convert a grayscale color space with 256 possible colors to 64 colors, we divide the original space in bins of 4 colors and map those 4 input colors to a single output color. In our case, the value of the output color is the maximum among the possible color values within the bin range.

B. Network architectures

Our experiments are carried out using models based on the following CNN architectures:

Simple CNN[2]*:* this architecture is composed of four convolutional layers divided into two blocks and two dense layers. The first block contains two convolutional layers with 32 filters (3×3) each, followed by a max pooling layer and a dropout layer. The second block consists of the remaining two convolutional layers (64 filters, 3×3), also followed by a max pooling and a dropout layer. This is then connected to a dense layer with 512 processing units, a dropout layer and, finally, the softmax layer.

Base Model CNN [17]: is similar to the Simple CNN architecture having a greater number of trainable parameters on the convolutional section of the network, that is the part used for feature extraction. This architecture was employed in [17] where it is called Base Model C.

ResNet-20 [18]: consists of twenty convolutional layers organized into six residual units. Each one residual unit uses the structure shown in Figure 2, alternating between a direct skip connection and using a single convolutional layer as part of the skip connection. The output of the last residual unit and the last skip connection go through an average pooling layer, which is then passed to a dense (softmax) layer that outputs the classification probabilities.

All the architectures used are illustrated in Figure 3, where the leftmost diagram represents the Simple CNN, the middle one shows the Base Model CNN network architecture, and the rightmost diagram, the ResNet-20. The layers shown in colored blocks are the ones used on experiments where feature extraction is performed, the entire networks are used otherwise. We also present the number of trainable parameters of both the entire network and the feature extractor (convolutional part) in Table I.

[2]Network architecture used in the example code: `https://github.com/keras-team/keras/blob/master/examples/cifar10_cnn.py`

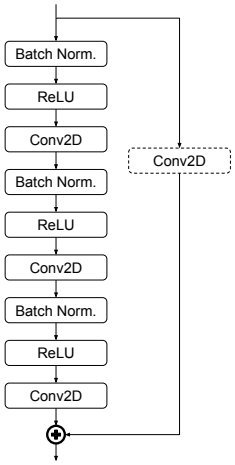

Fig. 2. Residual Unit.

The reason for choosing those particular networks are: the Simple CNN is a model that has a simpler feature extractor, given it has far less parameters on the convolutional part when compared to the other two models. The Base Model CNN – despite having more parameters in the convolutional part when compared to the Simple CNN – only uses simple CNN building blocks in a pretty straightforward manner. This network was also chosen as a comparison to the Simple CNN model, allowing to analyze whether an increase in the number of parameters improves the robustness of the obtained feature space. Lastly, the ResNet-20 network has a similar number of trainable parameters when compared to the Base Model CNN, however it takes advantage of a more complex structure to combine its layers. The ResNet-20 is also a deeper architecture.

We believe that the chosen architectures fairly covers the different, widely used, CNNs. Also, as our objective is to better understand the impacts that changing image quality, with regards to color quantization and noise, has on classification performance, we do not focus on obtaining state-of-the-art results. Instead, we analyze the changes in accuracy so we can have a good estimate of how CNNs behave in such scenarios.

TABLE I
NUMBER OF TRAINABLE PARAMETERS IN EACH ONE OF THE NETWORK ARCHITECTURES USED DURING THE EXPERIMENTS.

model	trainable parameters	
	full network	feature network
Simple CNN[2]	1,250,858	65,568
Base Model CNN [17]	457,284	456,874
ResNet-20 [18]	570,602	568,032

C. Datasets

The datasets used in our experimental setup are listed and explained next.

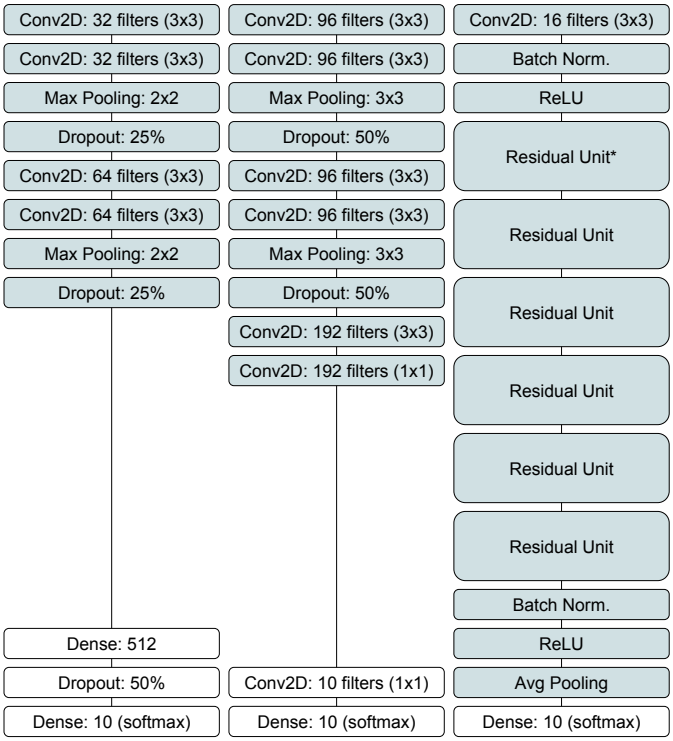

Fig. 3. Network architectures employed throughout our experiments. From left to right: Simple CNN, Base Model CNN and ResNet-20. Blocks in green represent convolutions used to extract features.

CIFAR-10 [10]: this dataset contains 60,000 32×32-RGB images equally split into 10 classes and it is divided into training and test sets with 50,000 and 10,000 images, respectively.

CIFAR-100 [10]: is a dataset with 100 classes, where each one has 600 32×32-RGB images. Training and test sets are provided and contain 500 and 100 images of each class, respectively. Two sets of classes are provided for this dataset. The first, and most used, contains 100 classes (referred to as *CIFAR-100 fine*). The second set groups the original 100 classes into 20 superclasses, being referred to as *CIFAR-100 coarse*.

Fashion-MNIST [19]: is a 10 class dataset composed of 28×28 grayscale images (256 colors). This dataset also has 50,000 images for training and 10,000 images for testing. To facilitate our transfer learning pipeline, we applied zero padding on those images to turn them into 32×32 images.

IV. Results and Discussion

As stated in Section III, our **first experiment** trained 21 distinct architectures (Simple CNN, Base Model CNN and ResNet-20) on seven versions of the CIFAR-10 dataset, where the first is the original (RGB) version and the others are grayscale ones quantized in 256, 128, 64, 32, 16 and 8 colors, respectively.

After training, we computed the accuracy of every model on all versions of the CIFAR-10 test set. The results are illustrated in Figure 4, from which we notice the models obtained from grayscale versions performed better then the original RGB when employed on grayscale versions of CIFAR-10. On the RGB test data they maintained a very similar performance. Moreover, models at greater quantization levels (8 and 16 colors) were specially good at maintaining a more stable accuracy across all test set versions.

Next, in our **second experiment**, we took the models trained on the first experiment as feature extractors for the CIFAR-10 and three other datasets (CIFAR-100 fine, CIFAR-100 coarse and Fashion-MNIST). To do so, we removed the dense part (classifier) from the network and use the activation maps for the last convolutional layer as a descriptor. This allows a for a direct evaluation of the robustness of the CNN features. As in experiment one, we created seven quantized versions of each dataset. Next, we took every possible combination of dataset and feature extractor to proceed as follows: 1) extract features from the training set and train a logistic regression; 2) extract features from the test set and evaluate this logistic regression model. Please see Section III for a more detailed explanation of this experiment.

The results from the second experiment are shown in Figure 5. By analyzing the plots, we draw some conclusions. First, when dealing with grayscale images, the feature extractor trained with RGB images is, in general, worse than most of the feature extractors trained on grayscale images. As an example – when using the ResNet-20 feature extractors to classify the 256-color grayscale version of *CIFAR-100 fine* – the RGB feature extractor achieved 41.13% of accuracy, while the grayscale with 64 colors obtained 44.89%. To show that this is not an isolated case we computed the Friedman statistical significance test to compare the models obtained using each CNN architecture with regards to the quantization level that they were trained with. For the Simple CNN architecture, the network trained on 64-color images was able to obtain the best average ranking, as shown in Table II. The Friedman test resulted in a p-value of $1\mathrm{e}{-7}$, therefore we can reject the null hypothesis (that there is no significant difference between the ranking of the classifiers) at a 5% level of significance.We also conducted this statistical significance test for the models based on both the Base Model CNN and ResNet-20 architectures, the rankings obtain by such tests are presented in Tables III and IV, respectively. For the test with the Base CNN models we obtained a p-value of $1\mathrm{e}{-8}$, while for the ResNet-20 models we obtained a p-value of $4\mathrm{e}{-8}$. Thus, we have enough evidence to reject both null hypothesis at a 5% level of significance.

The second conclusion is that, as in experiment one, the feature extractors obtained from images with less colors (mainly 8 and 16) are more consistent throughout the tests when it comes to accuracy. They also have comparable performance to the best model in most cases.

Lastly, in the **third experiment**, we compared the same feature extractors – trained on experiment one – with regards to resilience to noise. In order to do so, similarly to experiment two, we trained logistic regression models based on all feature extractor. However, we only trained them on the original

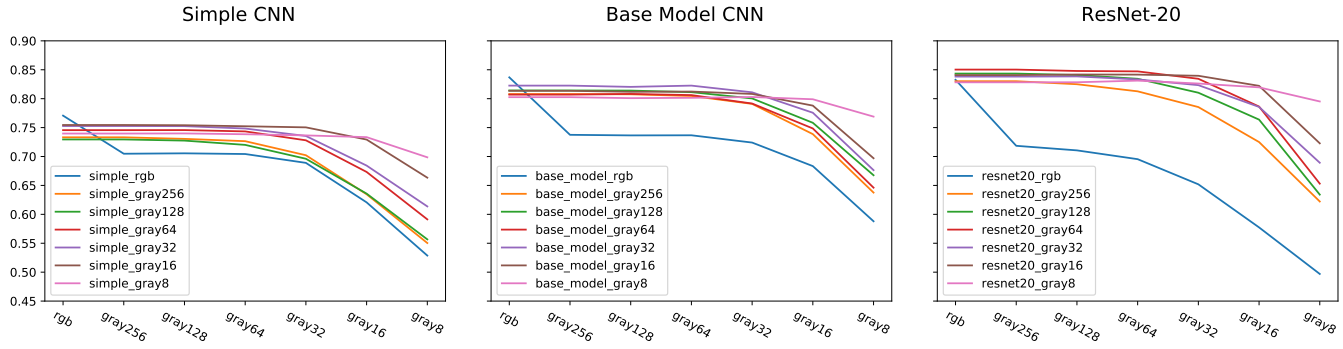

Fig. 4. Results of the first experiment. In all plots, the **x-axis** indicate the test set version, while the **y-axis** indicate the accuracy. Each line corresponds to the results obtained by a different model – that were trained in a particular version of the CIFAR-10 training set. Finally, each network architecture has its own separate plot.

TABLE II
AVERAGE RANKINGS FOR THE MODELS BASED ON THE SIMPLE CNN ARCHITECTURE.

model	average ranking
Simple CNN rgb	5.29
Simple CNN gray256	3.79
Simple CNN gray128	3.02
Simple CNN gray64	**2.62**
Simple CNN gray32	3.96
Simple CNN gray16	3.71
Simple CNN gray8	5.61

TABLE III
AVERAGE RANKINGS FOR THE MODELS BASED ON THE BASE MODEL CNN ARCHITECTURE.

model	average ranking
Base Model CNN rgb	5.43
Base Model CNN gray256	5.43
Base Model CNN gray128	4.52
Base Model CNN gray64	2.91
Base Model CNN gray32	**2.68**
Base Model CNN gray16	3.82
Base Model CNN gray8	3.21

TABLE IV
AVERAGE RANKINGS FOR THE MODELS BASED ON THE RESNET-20 ARCHITECTURE.

model	average ranking
ResNet-20 rgb	5.89
ResNet-20 gray256	4.70
ResNet-20 gray128	3.20
ResNet-20 gray64	2.96
ResNet-20 gray32	**2.75**
ResNet-20 gray16	4.18
ResNet-20 gray8	4.32

version of Fashion-MNIST (noise-free grayscale images with 256 colors). Then, we tested such classifiers on the original test set distorted by Gaussian noise with standard deviations equal to 10, 20 and 30, respectively. **Notice the Gaussian noise was applied only on the test set.** The results of such experiment are presented in Figure 6.

A closer look at those results shows that the models using features from heavily quantized images, especially the ones from 8 and 16 colors, generally performed significantly better than the other ones. One noticeable example occurred with the Simple CNN feature extractors when the images were affected by a Gaussian noise of standard deviation equals to 30. In this particular case, the feature extractor trained using images with 8 possible colors obtained an accuracy of 63.96%, while the one trained using images of 256 colors was capable of achieving only 43.41%.

V. REPRODUCIBILITY REMARKS

In order to facilitate the reproducibility of all experiments, all source codes are available at `https://github.com/tiagosn/cnn_rgb_grayscale` .

VI. CONCLUSIONS

We studied the impacts of using CNN architectures trained on different color scheme images as feature extractors for RGB and grayscale domains with different levels of quantization. After conducting our experiments, we concluded that, if computing power is available, it is better to train a new network on a grayscale/quantized version of the original dataset, since there is empirical evidence that it will achieve accuracy improvements when applied to a grayscale domain. When comparing the classification results of our features in different quantization levels we observed that, for two out of three architectures used in the experiments, the model trained with images quantized to 32 gray levels obtained the best results. Nevertheless, the results obtained by the features learned from images with 64 gray levels seem to more consistent. That is, the average ranking over all architectures achieved when using 64 gray levels was better than when using 32 gray levels.

Additionally, we evaluated the robustness of the learned features in noisy scenarios using different quantization levels. Our results confirm that models trained on datasets with less colors – especially the ones trained with 8 and 16 color levels – perform considerably better when classifying noisy images. Therefore, training a CNN on a quantized version of the original dataset can be advantageous when performing transfer learning to images that are expected do be affected by noise.

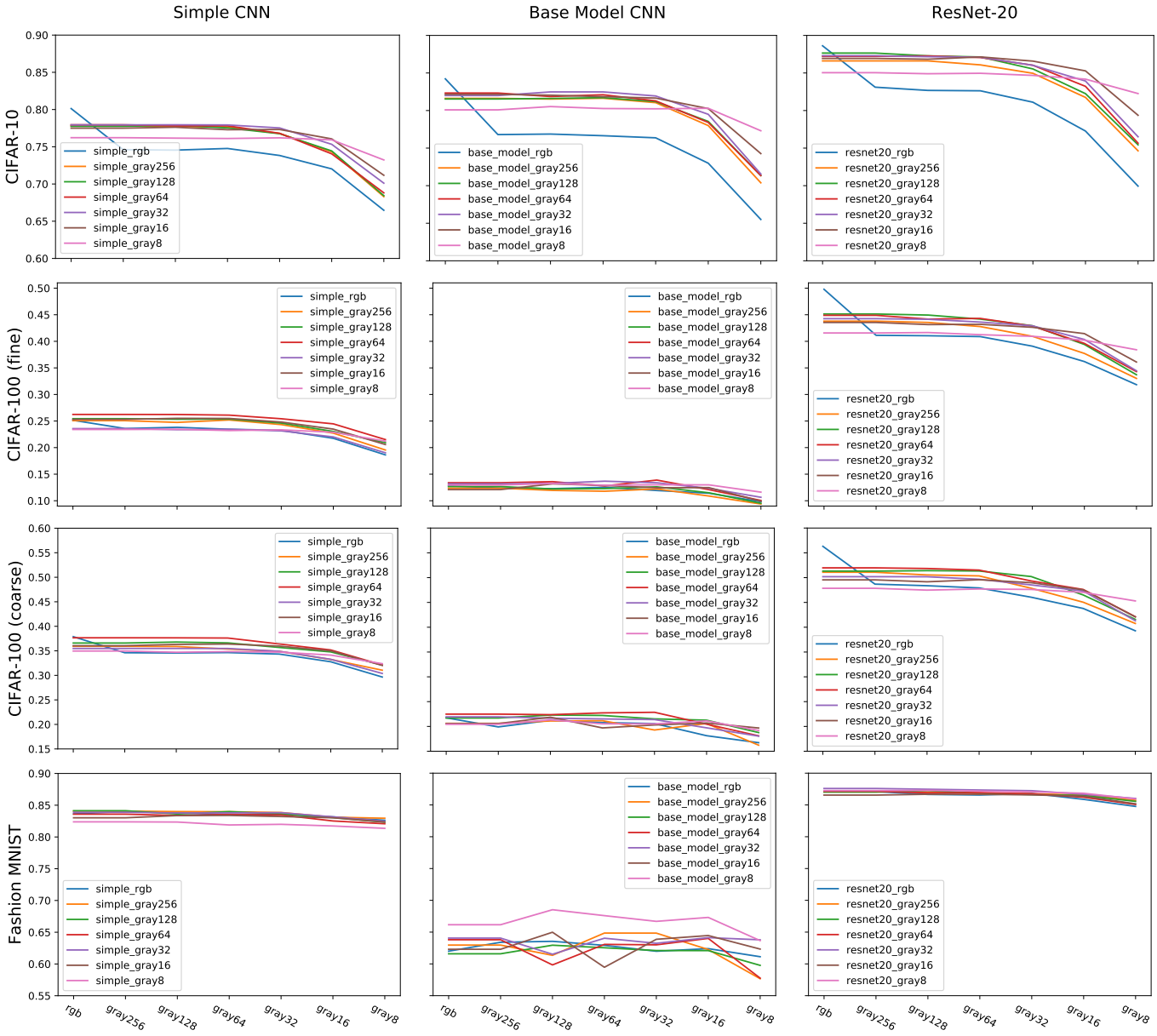

Fig. 5. Results of the second experiment. In all plots, the **x-axis** indicates the test set version, while the **y-axis** indicates the accuracy. The lines correspond to results obtained for different models – inferred using a particular version of the training sets. Rows contain the results for a particular dataset, while columns identify models based on architectures. **Notice that each row has a different range for the y-axis**.

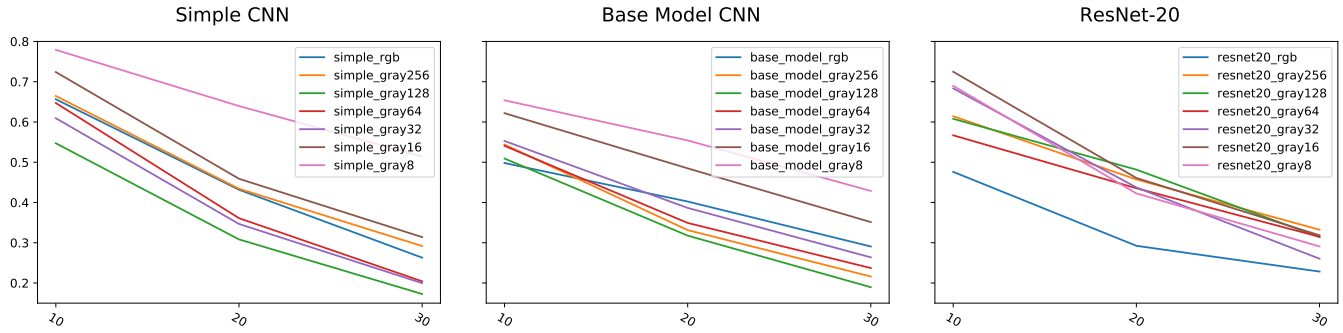

Fig. 6. Results of the third experiment. In all plots, the **x-axis** indicates the standard deviation of the Gaussian noise used to distort the test set, while the **y-axis** corresponds to the accuracy obtained by the logistic regression classifier.

VII. Future work

As future work, we intend to analyze the robustness of CNN features in terms of:

- Better understanding the resilience to noise and quantization of each feature map learned by each network layer. This can be done by conducting experiments similarly to as in [4];
- Exploring other types of noise, such as salt & pepper;
- In [20] and [16], authors showed that the color quantization technique can have a great impact on classification results, when using hand-crafted features. Thus, a more detailed study on the best quantization approach would be of great importance for CNN features;
- Performing experiments with other CNN architectures (e.g. VGG [11] and Inception [21]) and additional datasets, such as ImageNet.

Acknowledgment

This work was supported by FAPESP (grants #2015/04883-0, #2015/05310-3, #2016/16111-4, #2017/16548-6), CNPq (grants #307973/2017-4 and #302077/2017-0), Itaú-Unibanco, and partially supported by the CEPID-CeMEAI (FAPESP grant #2013/07375-0).

References

[1] M. A. Ponti, L. S. Ribeiro, T. S. Nazare, T. Bui, and J. Collomosse, "Everything you wanted to know about deep learning for computer vision but were afraid to ask," in *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T)*, vol. 00, Oct. 2018, pp. 17–41.

[2] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, 2018.

[3] I. B. Barbosa, M. Cristani, B. Caputo, A. Rognhaugen, and T. Theoharis, "Looking beyond appearances: Synthetic training data for deep cnns in re-identification," *Computer Vision and Image Understanding*, vol. 167, pp. 50–62, 2018.

[4] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, ser. CVPRW '14. Washington, DC, USA: IEEE Computer Society, 2014, pp. 512–519.

[5] M. Ravanbakhsh, M. Nabi, H. Mousavi, E. Sangineto, and N. Sebe, "Plug-and-play cnn for crowd motion analysis: An application in abnormal event detection," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, vol. 00, 2018, pp. 1689–1698.

[6] J. Ahn, J. Park, D. Park, J. Paek, and J. Ko, "Convolutional neural network-based classification system design with compressed wireless sensor network images," *PLOS ONE*, vol. 13, 05 2018.

[7] S. Dodge and L. Karam, "Understanding how image quality affects deep neural networks," in *2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX)*, Jun. 2016, pp. 1–6.

[8] T. S. Nazaré, G. B. P. da Costa, W. A. Contato, and M. Ponti, "Deep convolutional neural networks and noisy images," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, M. Mendoza and S. Velastín, Eds. Cham: Springer International Publishing, 2018, pp. 416–424.

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[10] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

[11] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[12] G. Kylberg and I.-M. Sintorn, "Evaluation of noise robustness for local binary pattern descriptors in texture classification." *EURASIP J. Image and Video Processing*, vol. 2013, p. 17, 2013.

[13] T. Ojala, M. Pietikainen, and D. Harwood, "Performance evaluation of texture measures with classification based on kullback discrimination of distributions," in *ICPR94*, 1994, pp. A:582–585.

[14] G. B. Paranhos da Costa, W. A. Contato, T. S. Nazare, J. E. S. Batista Neto, and M. Ponti, "An empirical study on the effects of different types of noise in image classification tasks," in *XII Workshop de Visão Computacional (WVC 2016)*, 2016.

[15] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, June 2005, pp. 886–893 vol. 1.

[16] M. Ponti, T. S. Nazaré, and G. S. Thumé, "Image quantization as a dimensionality reduction procedure in color and texture feature extraction," *Neurocomputing*, vol. 173, pp. 385–396, 2016.

[17] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," *arXiv preprint arXiv:1412.6806*, 2014.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *European Conference on Computer Vision*. Springer, 2016, pp. 630–645.

[19] H. Xiao, K. Rasul, and R. Vollgraf. (2017) Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms.

[20] C. Kanan and G. W. Cottrell, "Color-to-grayscale: does the method matter in image recognition?" *PloS one*, vol. 7, no. 1, p. e29740, Jan. 2012.

[21] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Computer Vision and Pattern Recognition (CVPR)*, 2015.