Improving the Generalization of Colorized Image Detection with Enhanced Training of CNN

Weize Quan[1,2], Kai Wang[2], Dong-Ming Yan[1*], Denis Pellerin[2], and Xiaopeng Zhang[1]
[1]*NLPR, Institute of Automation, Chinese Academy of Sciences / University of Chinese Academy of Sciences, Beijing, China*
[2]*University Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, Grenoble, France*
qweizework@gmail.com, kai.wang@gipsa-lab.grenoble-inp.fr, yandongming@gmail.com

***Abstract*—Image colorization achieves more and more realistic results with the increasing power of recent deep learning techniques. It becomes more difficult to identify the synthetic colorized images by human eyes. In the literature, handcrafted-feature-based and convolutional neural network (CNN)-based forensic methods are proposed to distinguish between natural images (NIs) and colorized images (CIs). Although a recent CNN-based method achieves very good detection performance, an important issue (*i.e.*, the blind detection problem) still remains and is not thoroughly studied. In this work, we focus on this challenging scenario of blind detection, *i.e.*, no training sample is available from "unknown" colorization algorithm that we may encounter during the testing phase. This blind detection performance can be regarded as the generalization capability of a forensic detector. In this paper, we propose to first automatically construct negative samples through linear interpolation of paired natural and colorized images. Then, we progressively insert these negative samples into the original training dataset and continue to train the network. Experimental results demonstrate that our enhanced training can significantly improve the generalization performance of different CNN models.**

***Index Terms*—Image forensics, natural image, colorized image, convolutional neural network, generalization, negative samples**

I. Introduction

With the increasing popularity and sophistication of image editing technologies, it is now relatively easy to create edited images that are visually very plausible. For example, current advanced colorization algorithms, more or less leveraging the powerful capacity of deep neural networks, can automatically colorize a grayscale image to obtain a high-quality color image. Fig. 1 shows a group of images, the left-most one is the original color image, and the remaining three are colorized images produced by three state-of-the-art colorization algorithms (respectively with the name Ma [1], Mb [2] and Mc [3] from left to right), which take the grayscale version of the left-most image as input. It is indeed difficult to distinguish which images are colorized by naked human eyes. Although this technique brings convenience to people's live in fields like digital entertainment, it may also be maliciously used and potentially lead to security issues, such as confounding object recognition or scene understanding [4]. Therefore, distinguishing between natural images (NIs) and colorized images (CIs) has become an important research problem in image forensics.

Recently, Guo *et al.* [4] first considered and studied this new forensic problem. On the basis of the statistical difference between NIs and CIs in the hue, saturation, dark, and bright channels, two methods, namely, histogram-based and Fisher-encoding-based, were designed to catch the forensic difference between NIs and CIs. After having obtained the feature vectors, they trained the support vector machine (SVM) classifiers to identify fake colorized images. Zhuo *et al.* [6] greatly improved the detection performance using a CNN-based color image steganalyzer WISERNet (WIder SEparate-then-Reunion Network) [7]. However, in the challenging scenario of *blind detection*, *i.e.*, no training sample is available from "unknown" colorization methods that we may encounter during the testing phase of forensic detectors, the performance of both CNN-based [6] and handcrafted-feature-based [4] methods in general decreases. Hereafter, we call this blind detection performance as *generalization* performance. Take Fig. 1 as an example, the second and fourth images (produced by Ma and Mc) are misclassified as NI by a CNN model trained on NIs and CIs generated by Mb. In the meanwhile, although not being very rigorous, we choose to use the term "classification accuracy/performance" to indicate the detection performance on testing data in which CIs are generated by a same colorization method known by the training procedure.

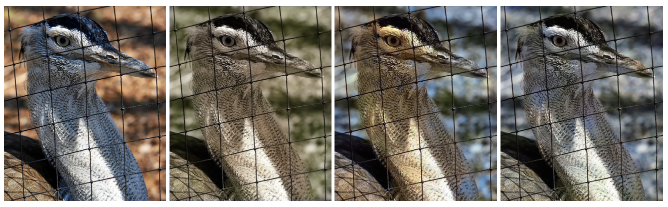


Fig. 1. From left to right: a natural image taken from ImageNet [5]; three colorized images generated by the colorization method proposed in [1], [2], and [3], respectively.

In order to cope with the challenging scenario of blind detection, in this paper we introduce a simple yet effective method. We construct negative samples via linear interpolation of paired natural and colorized images available in the training dataset, and iteratively add them into the original training dataset for additional and *enhanced CNN training*. This procedure is fully automatic, and can allow us to obtain stable and high generalization performance of the CNN.

The rest of this paper is organized as follows. Section II presents the technical details of the proposed method. Sec-

*Corresponding author.

tion III reports the performance evaluations for our method. Section IV draws the conclusions and proposes some future working directions.

II. Proposed Framework

A. Motivation

To the best of our knowledge, there is no existing work that considers the generalization capability yet for CNN-based image forensics. In fact, this is a highly challenging scenario because no training samples of the "unknown" colorization algorithms are available. In other words, we want the trained network to be able to successfully detect colorized images generated by new colorization methods that remain unknown during the training of CNN. This is a very realistic situation which can be commonly encountered after deploying a forensic detector in practical applications. We solve this challenging generalization problem through a simple yet effective approach, *i.e.*, inserting additional negative samples that are automatically constructed from available training samples, in order to carry out an enhanced training of CNN and thus to obtain an appropriate decision boundary for this classification problem. Besides considering the CNN model proposed in the very recent work of [6], in this paper, we also construct a different CNN model so as to validate and show that our enhanced training can work well on different networks.

B. Architecture of Networks

In this subsection, we describe the architecture of considered networks. The first layer (with so-called SRM, Spatial Rich Model, kernels [8]) of WISERNet is untrainable, while all weights of our designed network are trainable and thus we call it as AutoNet (Automatic Network). Let Ck(M or A) denote a Convolution-BatchNorm-ReLU(-MaxPool or -AveragePool) layer with k filters. Fk(R) denotes a fully-connected layer with k neurons (and with ReLU). The architecture of AutoNet is C32-C64M-C128M-C256M-C256M-C512M-C512M-C512-F2. All convolutional kernel sizes in AutoNet are 3×3. For layers 1-7, each convolutional layer (conv) is with the zero-padding of 1, and all max-pooling layers in AutoNet have the same kernel size of 3×3 and a stride of 2. For conv1, we use TanH as activation. Here we also give the architecture of WISERNet and more details can be found in [6]. The architecture of WISERNet is SRM-C72A-C288A-C1152A-F800R-F400R-F200R-F2, where SRM refers to channel-wise convolution where the convolutional kernels are fixed as the thirty 5×5 SRM filters borrowed from [8].

C. Negative Sample Insertion

According to our observation, there is a certain degree of performance decrease in the challenging blind detection scenario, not only for traditional handcrafted-feature-based methods [4], but also for CNN-based approaches (AutoNet and WISERNet [6]), although the latter has better performance. In details, for a traditional or CNN-based model trained on dataset constructed by one specific colorization algorithm, the test performance on datasets constructed by other colorization algorithms is sometimes limited for colorized images. The possible reason of this performance drop is that colorized images produced by a specific colorization algorithm tend to be equipped with a particular internal property, but CIs of different colorization algorithms are very likely to have different properties.

To clearly illustrate the encountered problem with an example, we train the AutoNet on the dataset constructed by colorization method Mb [2], and test on the datasets constructed by Ma [1] and Mc [3], respectively. It should be noted that Ma and Mc are the "unknown" colorization algorithms, and thus the corresponding samples of Ma and Mc are not used in the training process. We use t-distributed stochastic neighbor embedding (t-SNE) [9] to project the high-dimensional deep features (the output of conv8 of AutoNet, and its dimension is 512) of testing data constructed by above three colorization methods onto the two-dimensional map, and detailed visualization results are shown in Fig. 2. Comparing Fig. 2(a), (b) and (c), we find that the distributions of NIs (red squares) are relatively stable with a rather high intra-class variation, which is somehow expected; in the meanwhile, CIs (blue symbols) are more tightly clustered for each colorization algorithm but their locations change a lot for different methods [please compare the CIs in (a), (b) and (c), which correspond to Mb, Ma and Mc, respectively]. This is reasonable because the different colorization methods tend to have not exactly the same internal characteristics and hence the corresponding CIs have different locations in the feature space. When the features of CIs produced by "unknown" colorization algorithms (here Ma and Mc whose samples are not used for training) are near the decision boundary of the CNN (which is trained by using NIs and CIs produced by a "known" colorization algorithm, here Mb), and at the same time the decision boundary is relatively close to colorized images, there are high probabilities to misclassify the "unknown" CIs. For instance, many CIs in Fig. 2(b) (blue circles with red + in the figure) are wrongly predicted as NIs.

We would like to find a simple yet effective method to solve the encountered problem. The idea is that we make use of the available training samples (and only these samples) to construct an appropriate decision boundary which can lead to better generalization performance. A feasible and intuitive solution is to add negative samples (with same labels as CIs) near the initial decision boundary of the CNN, so as to make the CNN be more "strict" about the predictions of CIs and somehow push the classification boundary towards NIs. As such, it is expected that the "unknown" CIs located close to the initial decision boundary [*e.g.*, those shown in Fig. 2(b)] have more chance to be correctly classified with the new classification boundary which would be closer to NIs. More precisely, we construct negative sample through linear interpolation between *paired* NI and CI which share the same grayscale version and only differ in chrominance components. The corresponding formulation is shown below:

$$I_{NS} = \alpha \cdot I_N + (1 - \alpha) \cdot I_C, \tag{1}$$

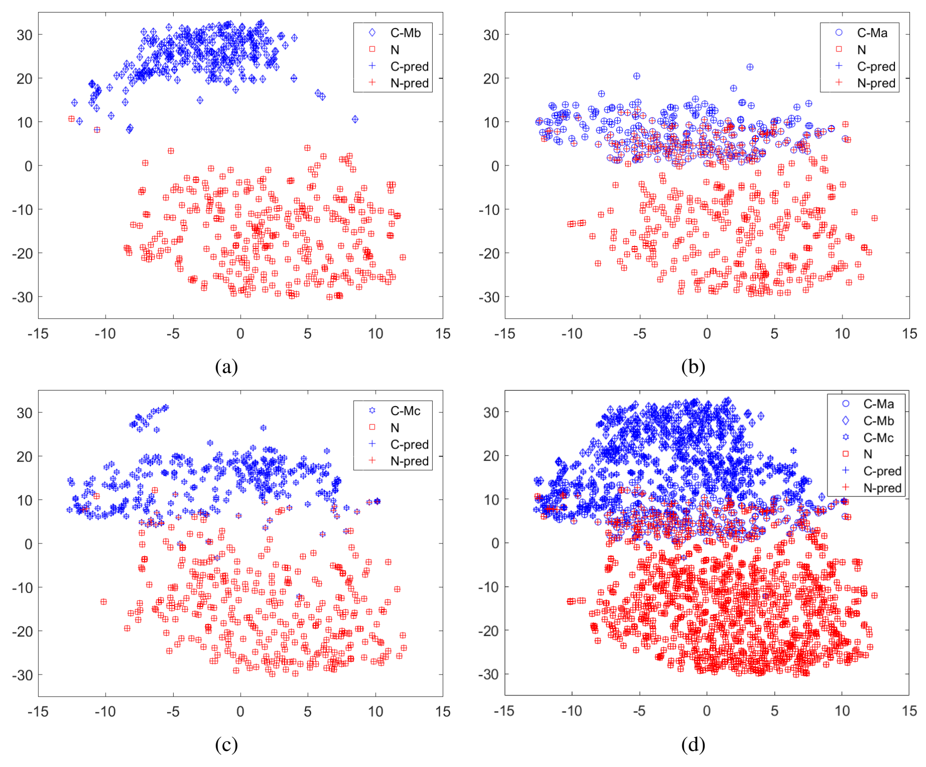


Fig. 2. The deep feature visualization with t-SNE [9]. The model is trained on the original dataset where CIs are generated by Mb. "C" means colorized images and "N" means natural images. "C-X" means the colorized images produced by X colorization method, for example, "C-Ma" corresponds to CIs generated by Ma colorization algorithm. "Y-pred" means that the predicted label of CNN is Y. We randomly select 900 natural images from validation dataset splitting them into three equal subsets of 300 images, and then we construct corresponding colorized images using Mb, Ma, and Mc for every 300 images. The deep feature is the output of conv8 of AutoNet, and the dimension is 512. (d) is the combination of (a) [Mb], (b) [Ma], and (c) [Mc].

where I_{NS} is the negative sample, I_N is the natural image, I_C is the corresponding colorized image, and $\alpha \in \{0.1, 0.2, 0.3, 0.4\}$ is the interpolation factor. This actually makes sense, as negative samples are in fact forensically negative (*i.e.*, considered as CIs), especially for our chosen weight values among $\{0.1, 0.2, 0.3, 0.4\}$ (*i.e.*, negative samples are closer to CIs than NIs). When α increases, the negative samples are progressively getting closer to the natural images and it is expected that the decision boundary is further moving towards NIs after enhanced training.

As analyzed above, adding negative samples and conducting additional training will push the classification boundary towards NIs. Thus, the classification accuracy on the NIs will gradually decrease as more and more negative samples are inserted. The classification accuracy of network on validation dataset also slightly decreases because the "known" CIs are almost all correctly classified and this accuracy mainly depends on the classification accuracy on the NIs. However, in the meanwhile the CIs constructed by "unknown" colorization algorithms are expected to be classified more correctly, implying a better generalization capability. Obviously, there is a trade-off between the classification accuracy (on data similar to the training samples) and generalization performance (mainly on "unknown" CIs) for the network. Therefore, without being able to directly measure the generalization during training of network, we consider the classification accuracy on NIs (on the so-called *natural validation dataset* $\mathcal{V}$) as a measure to select the final model in the process of additional training with negative sample insertion. In our work, we design a threshold-based model selection criterion. This threshold (θ) essentially determines the degree of final classification accuracy that can be accepted by user or current task. Generally speaking, larger θ means that the selected model has less high classification accuracy, but better generalization performance. Basically, we set $\theta = \beta \cdot error\_rate$, where β is a user defined parameter and $error\_rate$ is the classification error rate (in $\%$, measured on natural validation dataset $\mathcal{V}$) of the CNN model trained with the original training dataset $\mathcal{D}$ before negative sample insertion. This criterion simply defines the maximum tolerable value of the relative increase of error rate on $\mathcal{V}$ induced by enhanced training. In our experiments, we set $\beta = 2$. One exception is that when $error\_rate$ is very small (less than 1%), we set $\theta = 2\%$, meaning that we can slightly relax the constraint on classification error rate to obtain relatively large improvement of generalization performance.

Algorithm 1 illustrates the training process with negative sample insertion. It is worth noting that we only use CIs of a "known" colorization method but in a better way to construct a more appropriate decision boundary. In our experiments, this insertion is an iterative process with four iterations, *i.e.*, the α is increased from 0.1 to 0.4 with step of 0.1. Given a CNN model $\mathcal{M}$ trained by using original dataset $\mathcal{D}$, and some basic

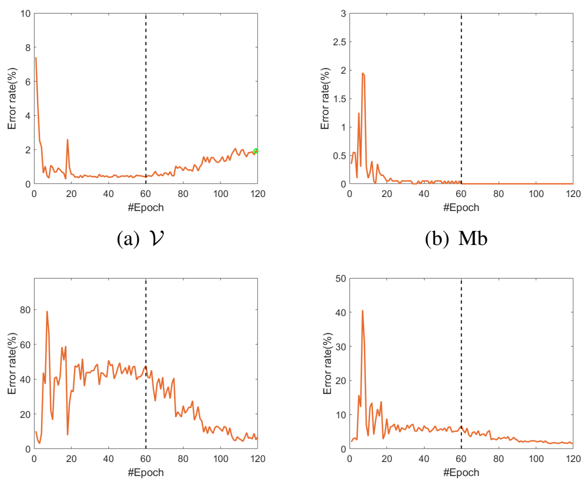
Algorithm 1 Enhanced training of CNN model with negative sample insertion

Input: $\mathcal{M}$, lr^0, S, $\mathcal{V}$, $\mathcal{D}$ and the set of corresponding natural and colorized image pairs $\mathcal{P}$ constructed from $\mathcal{D}$.
Output: final model after enhanced training.
Initialization: current learning rate $lr = lr^0$, set of negative samples $\mathcal{N} = \emptyset$, set of error rates on $\mathcal{V}$ of candidate CNN models $\mathcal{R} = \emptyset$.

```
 1: compute $error\_rate$ of $\mathcal{M}$.
 2: compute $\theta$.
 3: for all $\alpha \in \{0.1, 0.2, 0.3, 0.4\}$ do
 4:     construct negative samples from $\mathcal{P}$ using Eq. (1) and insert
        them into $\mathcal{N}$.
 5:     update training dataset: $\mathcal{D} = \mathcal{D} \cup \mathcal{N}$.
 6:     update the parameters of $\mathcal{M}$ for $S$ epochs. In the second half
        of training process, compute error rate on $\mathcal{V}$ for each model,
        and insert this value at the end of $\mathcal{R}$.
 7:     for all $I_{NS} \in \mathcal{N}$ do
 8:         if $I_{NS}$ is misclassified then
 9:             remove corresponding pair from $\mathcal{P}$.
10:         end if
11:     end for
12:     set $\mathcal{N} = \emptyset$.
13:     update current learning rate: $lr = lr \cdot 0.1$.
14: end for
15: select $i$-th model which satisfies $\max_i\{r_i | r_i \in \mathcal{R}, r_i < \theta\}$.
```

settings for CNN training, such as initial learning rate lr^0 and S epochs for each insertion, we first compute $error\_rate$ on $\mathcal{V}$ and then the threshold θ, which is used for final model selection. For each round of negative sample insertion, we construct negative samples and insert them into the dataset $\mathcal{D}$. Then, we update the parameters of model $\mathcal{M}$ using new training dataset, and compute the error rate on $\mathcal{V}$ starting from the second half of training process (*i.e.*, from $\lceil \frac{S}{2} \rceil$-th epoch for each insertion, where $\lceil . \rceil$ is the integer ceiling operator), because from that time the model becomes relatively stable. After each insertion, we test the negative samples produced by previous iteration. If a negative sample is misclassified, *i.e.*, the predicted label is NI and not consistent with its ground-truth label, then we stop using the corresponding pair to construct negative sample (*i.e.*, we remove corresponding pair from $\mathcal{P}$ as described in line 9 of Algorithm 1). In fact, this operation can slightly reduce the amount of negative samples, and does not weaken the performance of the network. After four iterations of insertion, we select the final CNN model. It is worth mentioning that when $\alpha \geqslant 0.5$, the negative samples will be close to NIs, and this is likely to have more impact on the classification of NIs. Here we take a conservative and experimentally effective approach, *i.e.*, stopping the negative sample insertion process after four iterations.

The complete training process of CNN model includes two stages: (1) using the original training dataset to train the deep model from scratch until convergence (normal training); (2) iteratively adding new negative samples into the original training dataset and continuing to train the model as summarized in Algorithm 1 (enhanced training). Fig. 3 shows the error rate curves of a complete training process of AutoNet. In the first



(a) $\mathcal{V}$ (b) Mb (c) Ma (d) Mc

Fig. 3. Error rate curves of a complete training of AutoNet. The network is trained on Mb [2], and tested on Ma [1] and Mc [3]. The error rates (in %) on CIs produced by these three methods are shown in (b), (c), and (d), respectively. The error rate on $\mathcal{V}$ is shown in (a). Black dotted line separates the two stages of normal training (60 epochs) and enhanced training (4×15=60 epochs). The green circle in (a) stands for the final selected model.

stage, the error rates on $\mathcal{V}$ and CIs produced by Mb obviously decline in the first 20 epochs and the network reaches the stability after about 50 epochs, as shown in Fig. 3(a) and (b). With the negative sample insertion, the error rate on $\mathcal{V}$ slightly increases, which can be found from the second part of Fig. 3(a). However, the generalization performance of network has a significant improvement on CIs produced by Ma [Fig. 3(c)] and a small improvement on Mc [Fig. 3(d)]. More numerical and visual results (including t-SNE visualization after enhanced training) are given in Section III.

III. Experimental Results

A. Implementation Details

All the experiments are implemented with PyTorch 0.3.1 [10]. The GPU version is GeForce® GTX 1080Ti of NVIDIA® corporation. All images in our experiments are resized to 256×256 using bicubic interpolation, and for each image, we convert its pixel values to $[-1, 1]$ (we first rescale the pixel values from the range $[0, 255]$ to the range $[0.0, 1.0]$, and then subtract these values by 0.5 and divide by 0.5). Stochastic gradient descent (SGD) with a minibatch of 20 is used to train AutoNet. Each minibatch contains 10 natural images and 10 colorized images. We randomly shuffle the order of training dataset after each epoch. For SGD optimizer, the momentum is 0.9 and the weight decay is 1e-4. The base learning rate is initialized to 1e-4. For the normal training (only using original training dataset) of AutoNet, we divide the learning rate by 10 every 20 epochs, and the training procedure stops after 60 epochs. For the normal training of WISERNet, we follow the setting described in [6]. As shown in line 13 of Algorithm 1, for the enhanced training of AutoNet and WISERNet, we adopt the same strategy about learning rate: the learning rate is divided by 10 every 15 epochs (it is enough to guarantee the convergence after new negative sample insertion), and the training procedure stops after 60 epochs, *i.e.*, 4 iterations of negative sample insertion.

Following [4] and [6], we also employ the *half total error rate* (HTER) to evaluate the performance of the proposed method. The HTER is defined as the average of misclassification rates (in %) of NIs and CIs. In this work, all reported results are the average of 7 runs.

B. *Effect of Negative Sample Insertion*

Before evaluating the proposed method, we provide the details of datasets used in our experiments. Following [4] and [6], three state-of-the-art colorization algorithms, Ma [1], Mb [2], and Mc [3] are adopted for producing CIs. NIs come from ImageNet dataset [5]. We use 10,000 natural images from ImageNet validation dataset to construct training dataset and validation dataset, and the ratio is 4:1. The exact indexes of these images are shared by the authors of [1]. Then, we remove the 899 grayscale images and 1 CMYK (cyan, magenta, yellow, and black) image from the remaining 40,000 images of ImageNet validation dataset (the total number of images in this dataset is 50,000), and obtain 39,100 natural images to construct testing dataset. Note that, the magnitude of testing dataset is far larger than the settings reported in [4] and [6]. We employ the three colorization methods mentioned above to produce the corresponding colorized images.

In this paper, we propose negative sample insertion to improve the generalization performance of CNN-based detectors. As described in Section II-C, this enhanced training uses natural validation dataset $\mathcal{V}$ to select the final model, and we randomly select 20,000 NIs from ImageNet test dataset [5] to construct $\mathcal{V}$. Table I reports the performance of AutoNet and WISERNet before (*i.e.*, the rows of "AutoNet" and "WISERNet") and after (*i.e.*, the rows of "AutoNet-i" and "WISERNet-i") negative sample insertion. We do not present the results of handcrafted-feature-based methods proposed in [4] because as shown in [6] and also verified by our experiments, CNN-based method has significantly better performance in terms of both accuracy and generalization. The difference between the results of the row of "WISERNet" and those reported in [6] is probably due to the differences in the generation of experimental data and the number of testing images (we use much more testing data). It is worth mentioning that here we focus on the generalization improvement after applying our proposed enhanced training for the two networks (*i.e.*, AutoNet and WISERNet), rather than the performance difference between them. We leave the architecture comparison and the design of CNN of better generalization as a future work. From Table I, we can see that the effect of negative sample insertion, *i.e.*, improving the generalization of network, is consistently stable for these two networks (except for one case, trained on Mc and tested on Mb for WISERNet, but with a very low final error rate of 1.08%). The negative sample insertion leads to slight decrease of the classification accuracy, however, the generalization performance of network usually has apparent improvement. For example, the initial generalization error of WISERNet trained on Mb and tested on Ma is 33.30%, and then reduces to 4.74% after enhanced training using negative samples, with a slight increase of classification error from 0.16% to 0.94%. This is also consistent with previous analysis (Section II-C) that there is a compromise between the accuracy and the generalization performance, and our negative sample insertion method can achieve a satisfying trade-off.

TABLE I
THE PERFORMANCE (HTER, IN %, LOWER IS BETTER) OF THE TWO CNN-BASED METHODS (AUTONET AND WISERNET [6]) ON IMAGENET VALIDATION DATASET [5]. FOR THE SAKE OF CLARITY, THE GENERALIZATION PERFORMANCE RESULTS ARE PRESENTED IN ITALICS.

Method	Ma			Mb			Mc		
	Ma	*Mb*	*Mc*	*Ma*	Mb	*Mc*	*Ma*	*Mb*	Mc
AutoNet	0.56	*10.57*	*10.62*	*31.65*	0.19	*6.16*	*13.93*	*1.91*	0.72
AutoNet-i	1.02	*6.94*	*5.12*	*5.13*	0.94	*1.92*	*3.33*	*1.75*	1.14
AutoNet-mixup	0.89	*12.45*	*15.35*	*20.68*	0.34	*10.04*	*8.42*	*2.25*	0.76
WISERNet	0.29	*2.21*	*10.74*	*33.30*	0.16	*7.88*	*5.80*	*0.59*	0.36
WISERNet-i	0.98	*1.22*	*2.29*	*4.74*	0.94	*2.04*	*2.46*	*1.08*	0.98

Very lately, we became aware of a recently proposed "mixup" learning principle [11] which regularizes the neural network and encourages the trained model to behave linearly in-between training examples. Although the linear interpolation is also used, there is an essential difference: "mixup" results in the linearly-transitioned decision boundary, while our method pushes the decision boundary towards NI. Based on the respective standing point, for the linear interpolation itself, [11] uses the interpolation factor in the range of $[0, 1]$ to combine pair of raw inputs and their labels, whereas our method uses that of $\{0.1, 0.2, 0.3, 0.4\}$ (forensically negative) and sets the label of new generated image as CI (the so-called negative sample). In addition, "mixup" is a form of data augmentation that implicitly affects the generalization of network, whereas our enhanced training explicitly controls the decision boundary and then improves the generalization of CNN-based detectors. In order to compare the "mixup" and our method, we train the model with "mixup" where the learning rate schedule is exactly the same as the normal training of AutoNet and the results are shown in Table I (the row of "AutoNet-mixup"). We set the "mixup" hyper-parameter $\alpha = 0.4$ as recommended in [11]. Obviously, the generalization of our enhanced training based on negative sample insertion is significantly better than that of "mixup", only with a slight decrease of the classification performance (please compare the rows of "AutoNet-i" and "AutoNet-mixup").

At last, we visualize deep features of AutoNet-i using t-SNE [9], and the results are shown in Fig. 4. Here, deep features are the output of conv8 of AutoNet-i, and its dimension is 512. The corresponding visualizations of the model before negative sample insertion are shown in Fig. 2. The testing data is also the same in Fig. 4 and Fig. 2. By comparing the border of correctly classified CIs, *i.e.*, blue symbols with a blue + inside, in Fig. 2(d) and Fig. 4(d), we can find that the latter has fewer misclassified CIs, and

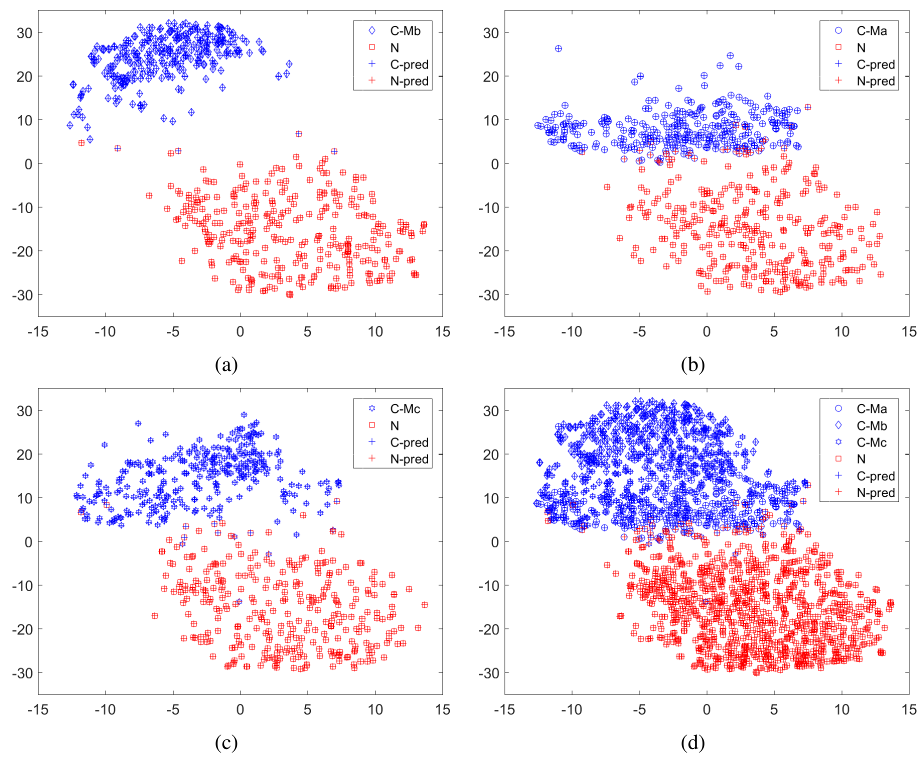


Fig. 4. The deep feature visualization of AutoNet-i with t-SNE [9]. The model is obtained through enhanced training of the previously trained model (used in Fig. 2). The meaning of symbols is same as that of Fig. 2. It is worth noting that in t-SNE the transformation used for dimension reduction and the obtained visualization depend on the input data. Therefore, transformation and visualization in this figure are different from those of Fig. 2.

the classification boundary is pushed towards NIs. The CIs generated by "unknown" colorization algorithms, especially Ma [1], are in consequence less misclassified, and this can be clearly observed by comparing Fig. 2(b) with Fig. 4(b). This confirms that our negative sample insertion scheme can push the decision boundary towards NIs to some extent and accordingly improve the generalization performance.

IV. Concluding Remarks

In this paper, we considered the challenging blind detection scenario and proposed an effective method based on negative sample insertion to improve the generalization capability of CNN-based models. The generalization performance is noticeably and consistently improved, with a very slight decrease of the classification accuracy. Our source code is available at https://github.com/weizequan/NIvsCI.

In the future, we are interested in employing the proposed enhanced training to improve the generalization performance of other kinds of forensic methods whenever applicable. We would like to explore other approaches to understanding and enhancing the generalization capability of neural networks.

Acknowledgment

This work is funded in part by Beijing Natural Science Foundation (L182059), National Natural Science Foundation of China (61620106003 and 61772523), and the French National Research Agency (DEFALS ANR-16-DEFA-0003, ANR-15-IDEX-02). W. Quan acknowledges the support from the UCAS Joint PhD Training Program. We would like to thank Dr. G. Larsson, Dr. R. Zhang and Dr. S. Iizuka for kindly sharing the source code of their colorization algorithms described respectively in [1], [2] and [3], Dr. L. van der Maaten for making the t-SNE tool [9] publicly available, L. Zhuo for the training details of their network [6] and Dr. Y. Guo for detailed and helpful discussions about their work [4].

References

[1] G. Larsson, M. Maire, and G. Shakhnarovich, "Learning representations for automatic colorization," in *Proceedings of the European Conference on Computer Vision*, 2016, pp. 577–593, https://doi.org/10.1007/978-3-319-46493-0_35.

[2] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Proceedings of the European Conference on Computer Vision*, 2016, pp. 649–666, https://doi.org/10.1007/978-3-319-46487-9_40.

[3] 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 Transactions on Graphics*, vol. 35, no. 4, pp. 1–11, 2016, https://doi.org/10.1145/2897824.2925974.

[4] Y. Guo, X. Cao, W. Zhang, and R. Wang, "Fake colorized image detection," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 8, pp. 1932–1944, 2018, https://doi.org/10.1109/TIFS.2018.2806926.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and F.-F. Li, "ImageNet: A large-scale hierarchical image database." in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255, https://doi.org/10.1109/CVPR.2009.5206848.

[6] L. Zhuo, S. Tan, J. Zeng, and B. Li, "Fake colorized image detection with channel-wise convolution based deep-learning framework," in *Proceedings of the Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, 2018, pp. 733–736, https://doi.org/10.23919/APSIPA.2018.8659761.

[7] J. Zeng, S. Tan, G. Liu, B. Li, and J. Huang, "WISERNet: Wider separate-then-reunion network for steganalysis of color images," *IEEE Transactions on Information Forensics and Security*, 2019, in press, https://doi.org/10.1109/TIFS.2019.2904413.

[8] J. Fridrich and J. Kodovsky, "Rich models for steganalysis of digital images," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 3, pp. 868–882, 2012, https://doi.org/10.1109/TIFS.2012.2190402.

[9] L. van der Maaten and G. E. Hinton, "Visualizing high-dimensional data using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[10] (visited on 2019-07-28). [Online]. Available: https://pytorch.org/

[11] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *Proceedings of the International Conference on Learning Representations*, 2018, pp. 1–13.