Detecting Computer Generated Images with Deep Convolutional Neural Networks

Edmar R. S. de Rezende*, Guilherme C. S. Ruppert*, and Tiago Carvalho†
*CTI Renato Archer, Campinas-SP, Brazil 13069-901
Email: {edmar.rezende,guilherme.ruppert}@cti.gov.br
†Federal Institute of São Paulo (IFSP), Campinas-SP, Brazil 13069-901
Email: tiagojc@gmail.com

***Abstract*—Computer graphics techniques for image generation are living an era where, day after day, the quality of produced content is impressing even the more skeptical viewer. Although it is a great advance for industries like games and movies, it can become a real problem when the application of such techniques is applied for the production of fake images. In this paper we propose a new approach for computer generated images detection using a deep convolutional neural network model based on ResNet-50 and transfer learning concepts. Unlike the state-of-the-art approaches, the proposed method is able to classify images between computer generated or photo generated directly from the raw image data with no need for any pre-processing or hand-crafted feature extraction whatsoever. Experiments on a public dataset comprising 9700 images show an accuracy higher than 94%, which is comparable to the literature reported results, without the drawback of laborious and manual step of specialized features extraction and selection.**

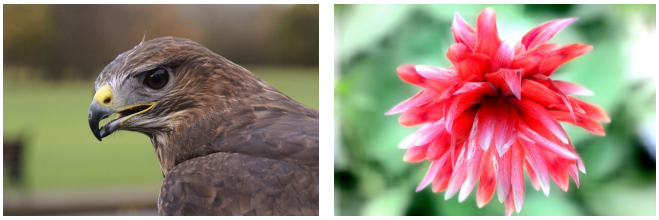
I. Introduction

Data from the *Global Games Market Report*[1] shows that, in 2015, digital games industry moved more than 91.8 billions of dollars. Such industry is always looking for innovation and improvements, in a way to respond to it's consumers wishes for realistic games, with almost perfect graphics.

In the same way, cinematographic industry also lives a "gold rush", using techniques able to produce movies so realistic that its impressive results could deceive an inattentive person, even when using just Computer Graphics (CG) characters.

This crusade for perfect visual quality, result in more and more robust and precise CG methods for people, objects, and digital scenarios generation. Associated with high processing power in current computers, such methods are able to generate results so impressive as the one depicted in the last Star Wars movie [2] where the actress Carrie Fisher has been digitally reproduced with the same appearance of the beginning of her carrier, in the 70's.

However, as pointed by *Holmes et.al.* [1], once the perfect CG image generation goal is achieved, it brings with itself challenges for other science areas as, for example, the challenge of discerning between a photo generated (PG) — the one generated by a digital camera — and an image generated by CG methods. Figure 1, depicts an example of how difficult is to discern between PG and CG images. Imagine, for example, the retaliation that a CG image reporting a terroristic act, as the execution of a missing reporter, spreading out on the Internet could cause. In this context, the accurate detection of CG images has become more important in the last years. Such differentiation has even more legal implications when attached to child pornography detection. In Brazil, for example, according to Law 11.829, of 2008 November 25th, any person who produces, reproduces, direct, take pictures or record, in any way, scenes involving explicit sexual or pornographic act involving children or teenagers can be sentenced from 4 to 8 years in jail. But what happen if the scene has been produced by CG methods? The legal consequences are the same?



(a) PG (b) CG

Fig. 1. Example of how challenging is recognize a PG and a CG image by simple visual analysis.

Different methods on Digital Forensics have been proposed to identify the difference between CG and real images and videos [2]–[5] but, despite to present improvements, such methods are far from the complete problem resolution. Usually, such methods works with approaches focused in discovery inconsistency in very specific details, as the work proposed by *Conotter et.al.* [6], which uses information associated with blood flow captured from videos involving people constructed using CG. Another kind of approach involves machine learning application for CG images identification [2].

In the last few years, Deep Neural Networks (DNN) have become the standard approach for image classification tasks. Deep learning algorithms [7]–[9] are learning methods with multiple levels, each transforming the representation at one level (starting with the raw input) into a representation at a higher, slightly-more-abstract level. With the composition of enough such transformations, very complex functions can be learned. Its key aspect is that these feature layers are not designed by human engineers, rather they are learned from data using a general purpose learning procedure.

This transition from traditional approach based on hand-crafted feature extractors combined with shallow classifiers to DNNs is due to the overwhelming performance of deep Convolutional Neural Networks (CNNs) on classification challenges such as the ILSVRC (ImageNet Large Scale Visual Recognition Challenge) [10].

Over the years, there has been a trend where the deeper the model is, the better performance the model can get on the ImageNet challenge. In 2012, the AlexNet architecture with 8 layers resulted in a top-5 classification error of 16.4% on the ImageNet challenge [11]. In 2014, the VGG16 model with 16 layers and VGG19 model with 19 layers resulted in a top-5 classification error of 7.3% [12], and the GoogleNet model with 22 layers resulted in a top-5 classification error of 6.7% [13]. And finally in 2015, the Residual Network (ResNet) model with 152 layers resulted in a top-5 classification error of 3.57% [14].

[1]https://goo.gl/xkWPon
[2]http://www.imdb.com/title/tt3748528/

In this paper we present a new method for CG image detection using a deep CNN based on the Residual Network model with 50 layers (ResNet-50) [14]. Using a transfer learning approach [15], we transfered the weights of ResNet-50 layers pre-trained on ImageNet dataset to our model, replacing the last layer by a trained classifier, being able to classify PG and CG images with 94% accuracy.

Among the main contributions of this paper we can highlight: (1) proposition of a new approach for CG images detection based on a deep CNN model combined with a transfer learning approach; (2) an accuracy around 94%, comparable with state-of-the-art methods; (3) decrease of complexity in features engineering process when compared with state-of-the-art methods.

The rest of the paper is organized as follows: Section II describes some of the main works related with CG images detection in Digital Forensics literature. Section III describes in details the proposed methodology. Section IV presents the main experiments performed for methodology validation, exposing the obtained results and comparing them with literature methods. Finally, Section V presents the main conclusions and future research directions.

II. Related Work

When talking about the topic of discern between CG and real images, many literature works have been developed. Some of these works investigate how is the behavior of people when exposed to this kind of image and its impact in law. Is the case of *Holmes et.al.* [1] work, where the authors discuss the legal problems caused by highly realistic images generated by CG methods, specially for child pornography. To show how easy, or not, is to deceive users using this kind of images, the authors propose two experiments: the first one involves a training stage for users, and the second one, where the users did not receive this training. This experiment consists in expose each user to 60 images (15 real images containing one man each, 15 CG images containing one man each, 15 real images containing one woman each, and 15 CG images containing one woman each). The user were asked to answer the sex (man or woman) and if the image has been generated by CG or not. In the first round of experiments, the users did not receive the training and users accuracy stayed around 50% in CG images detection. In the second experiment, the users received a simple training, which shows that they improved their accuracy at the task. Based on these experiments, and also using images information, the authors conclude that when CG image quality is improved, it becomes even harder for people to detect differences based on a simple visual analysis.

Farid [16] emphasizes that in the United States, when analyzing child pornography images, if the child in the image has been generated by CG, not being a real human being, the content does not represent a crime. The work also presents approaches to detect some kinds of deformation and retouches in images using color filters.

Looking for detecting computer generated person in videos, but also using perceptual details, *Conotter et.al.* [6] proposed to use information associated with blood flow. Capturing tiny movements on cheeks and forehead, the authors produce a characteristics signal for real and CG images. In CG images, this signal is characterized by the presence of many peaks, while in real images the sinal is most of time flat.

In machine learning field, proposed methods usually extract different features for training a specialized classifier to identify patterns of CG and real images. *Tokuda et.al.* [2] propose to use a combination of a big number of feature extraction algorithms associated with different classifiers fusion techniques in a way to detect CG images. The authors report an 97% accuracy in a dataset of 9700 images.

Tan et.al. [17], based upon the statement that texture features has a strong ability to distinguish CG and real images, have used Local Ternary Patterns (LTP) for features extraction and posterior classification. Using a Support Vector Machine (SVM) [18] classifier, the authors achieve an accuracy of approximately 97% in a dataset of 2200 images collected from different sources, as for example, the Columbia University natural image library [19].

III. Proposed Method

The CG detection method proposed in this work relies upon a deep CNN architecture to classify each image from the dataset using their raw RGB values of the pixels as features, without necessity of manual feature extraction. Figure 2 depicts an overview of entire method's pipeline.

The dataset consists of variable-resolution images, while our model requires a constant input dimensionality. Therefore, we resize the images to a fixed resolution of 224×224. The only pre-processing we do is subtracting the mean RGB value computed on the ImageNet dataset from each pixel, as proposed by *Krizhevsky et.al.* [11].

Using a transfer learning approach, we transfer the weights of ResNet-50 layers pre-trained on ImageNet dataset to our deep CNN model, removing the last 1000 fully-connected (fc) softmax layer. Then, we pass the pre-processed training set images through the deep CNN to extract the bottleneck features[3]. In our model, the bottleneck features are the activation maps generated by the average pooling layer.

The bottleneck features are then used to train a new classifier for CG image detection. This approach is equivalent to replace the last 1000 fc softmax layer of ResNet-50 by the new classifier freezing the parameters of the convolutional layers during the training process, with the advantage of a much smaller training time.

Finished the training process, the new trained classifier is stacked on the top of the layers transfered from ResNet-50 to build our deep CNN model, which is used to classify the test images. In this work, we propose two distinct deep CNN models: the first one with a 2 fully-connected softmax layer at the top, and the second one with a SVM classifier at the top. Fig. 3 shows a comparison between the original ResNet-50 and the two proposed deep CNN architectures.

A. Transfer Learning Process

Transfer learning consists in transferring the parameters of a neural network trained with one dataset and task to another problem with a different dataset and task [15].

The usual transfer learning approach consists in training a base network and then copying its first n layers to the first n layers of a target network. The remaining layers of the target network are then trained toward the target task. One can choose to backpropagate the errors from the new task into the base (copied) features to fine-tune them to the new task, or the transferred feature layers can be left frozen, meaning that they do not change during training on the new task. The choice of whether or not to fine-tune the first n layers of the target network depends on the size of the target dataset and the number of parameters in the first n layers. If the target dataset is small and the number of parameters is large, fine-tuning may result in overfitting, so the features are often left frozen. On the other hand,

[3]Bottleneck term refers to a topology of a neural network where the hidden layer has significantly lower dimensionality than the input layer, assuming that such layer — referred to as the bottleneck — compresses the information needed for mapping the neural network input to the neural network output, increasing the system robustness to noise and overfitting. Conventionally, bottleneck features are the output generated by the bottleneck layer.

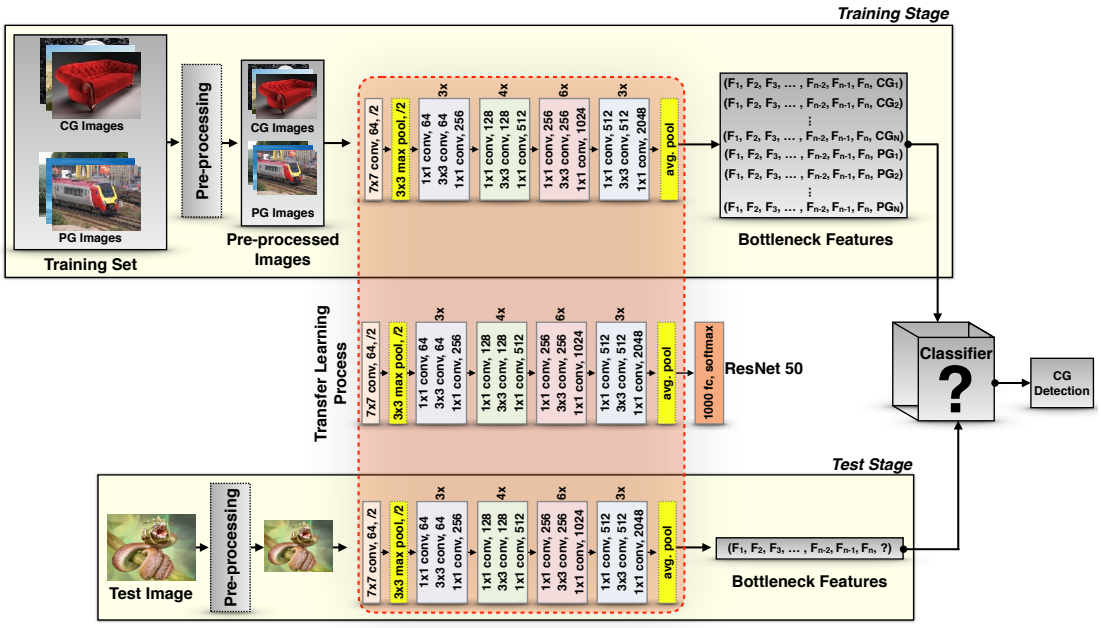


Fig. 2. Overview of proposed method. Transferring ResNet-50 parameters to our model to extract bottleneck features, which are used to train a classifier.

if the target dataset is large or the number of parameters is small, so that overfitting is not a problem, then the base features can be fine-tuned to the new task to improve performance.

At first glance, this process could sound meaningless because the traditional common sense in machine learning expects that the training should be performed specifically for the target dataset and task. However, many deep neural networks trained on natural images exhibit a curious phenomenon in common: on the first layers they learn features that appear not to be specific to a particular dataset or task, but general in that they are applicable to many datasets and tasks. Features must eventually transition from general to specific by the last layers of the network.

When the target dataset is significantly smaller than the base dataset, transfer learning can be a powerful tool to enable training a large target network without overfitting. Recent studies have taken advantage of this fact to obtain state-of-the-art results [20] [21] [22], collectively suggesting that these layers of neural networks do indeed compute features that are fairly general.

In our transfer learning approach, we use ResNet-50 as the base model. ResNet-50 was pre-trained for object detection task on the ImageNet 2012 dataset [10] containing 1.28 million images of 1000 classes. We copied the first 49 layers of ResNet-50, replacing its top layer by a 2 fully-connected softmax layer in the first proposed model, and by a SVM classifier in the second proposed model. In both models, the transferred feature layers were left frozen during training on the CG detection task.

B. ResNet-50 Architecture

Residual Networks (ResNets) [14] are deep convolutional networks where the basic idea is to skip blocks of convolutional layers by using shortcut connections to form shortcut blocks named residual blocks. The residual block can be expressed in a general form:

$$y_l = h(x_l) + F(x_l, W_l),$$

$$x_{l+1} = f(y_l)$$

where x_l and x_{l+1} are input and output of the l-th block, respectively. F is a residual mapping function, $h(x_l) = x_l$ is an identity mapping function, and f is a rectified linear unit (ReLU) function [23]. These stacked residual blocks greatly improve training efficiency and largely resolve the degradation problem present in deep networks.

In ResNet-50 architecture, the basic blocks are composed of a sequence of convolutional layers with 1×1, 3×3 and 1×1 filters respectively, that follow two simple design rules: (i) for the same output feature map size, the layers have the same number of filters; and (ii) if the feature map size is halved, the number of filters is doubled. The down-sampling is performed directly by convolutional layers that have a stride of 2 and batch normalization [24] is performed right after each convolution and before ReLU activation.

The identity shortcuts can be directly used when the input and output are of the same dimensions. When the dimensions increase, two options are considered: (i) The shortcut still performs identity mapping, with extra zero entries padded for increasing dimensions. This option introduces no extra parameter; (ii) The projection shortcut is used to match dimensions (done by 1×1 convolutions). For both options, when the shortcuts go across feature maps of two sizes, they are performed with a stride of 2.

The network ends with a global average pooling layer and a 1000-way fully-connected layer with softmax activation. The total number of weighted layers is 50.

C. Top Classifier

The original work that presented ResNet-50 [14] proposes an architecture where the last layer is a 1000 fully-connected softmax layer. In our method, we explore the same architecture adapting to a 2 fully-connected softmax since we only have two classes here. This last layer works as the top classifier, doing the classification task itself, while the previous layers can be seen as feature extraction layers.

Furthermore, we also extend the method by replacing this last layer for a SVM classifier, to evaluate the performance of a different

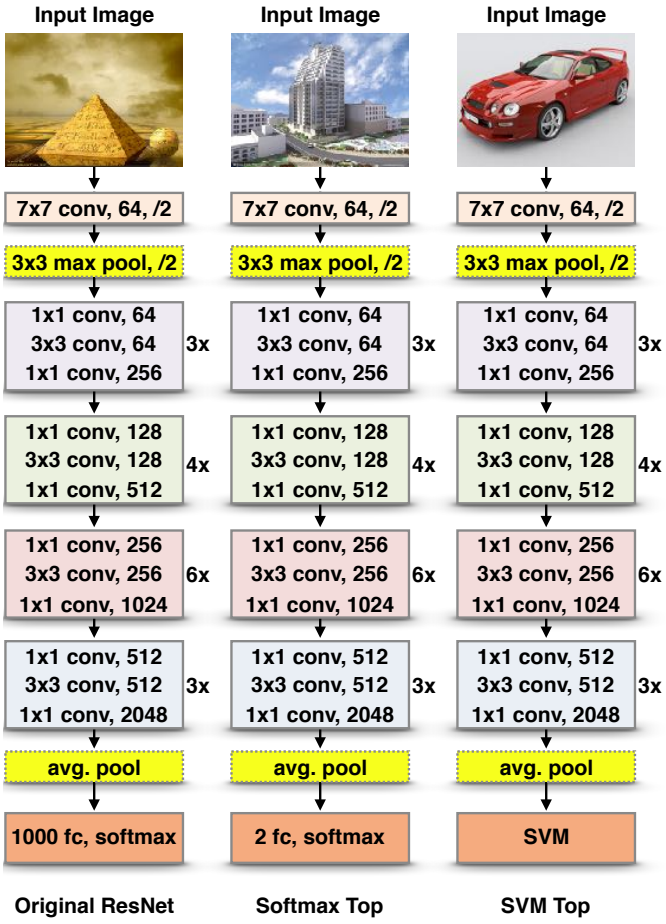


Fig. 3. Comparison between the original ResNet-50 and the two proposed deep CNN architectures, the first one replacing the 1000 fully-connected softmax by a 2 fully-connected softmax layer and the second one by a SVM classifier in the top layer.

classifier on top of the architecture. Figure 3 shows the comparison of the architectures.

IV. Experiments and Results

This section presents the main experiments performed to validate proposed approach.

A. Dataset

Proposed method has been tested over a public dataset proposed by *Tokuda et.al.* [2]. The dataset comprises 9700 equally divided between CG and PG images of different kinds of scenarios as outdoor, animals, objects, people, cars and others. As reported by the authors, all of the images have been collected from Internet and compressed in JPEG format, presenting physical sizes between 12 KB and 1.8 MB. The dataset contains images with different resolutions and, different from *Tokuda et.al.*, we worked over the entire image, without crop its central region.

B. Validation Protocol

In a way to guarantee a fair comparison with results reported by *Tokuda et.al.* [2], we applied the same five fold cross-validation protocol, reporting besides average accuracy, the accuracy and execution time of each test fold.

C. Implementation Details

We implemented proposed method using Python 3.5 with Keras 2.0.3[4] and ThensorFlow 1.0.1[5]. All performed tests have been executed in a machine with an Intel(R) Xeon(R) CPU E5-2620 2.00GHz with 96GB of RAM without GPU usage.

D. Visualization of Bottleneck Features

As described in Section III, our method takes advantage of transfer learning process to generate ResNet-50 bottleneck features, projecting the 150528 input features ($224 \times 224 \times 3$ RGB values of the pixels of each image) in a lower-dimensional space of 2048 features. This process intends to generate a set of features with a better degree of separability, which could allow the top classifier to achieve a higher classification accuracy.

To evaluate if the bottleneck features would in fact produce the desired boost in classification accuracy, we applied the t-Distributed Stochastic Neighbor Embedding (t-SNE) [25] dimensionality reduction technique to visualize our high-dimensional features. We projected the 150528 input features and the 2048 bottleneck features in 2D, and plot them as points colored according to their class, as depicted in Figure 4. Red squares represent PG samples while blue circles represent CG samples. It is possible to observe that the operations performed by ResNet-50 layers projected the raw pixels into a better separable feature space.

E. Classification By Softmax Algorithm

At this round of experiments, we classify our samples using a deep CNN architecture similar to the original ResNet-50. The only difference is that we use a 2 fully-connected softmax top layer instead of using the original 1000 fully-connected softmax layer. Our top layer has been trained with categorical cross-entropy cost function and Adam optimizer for a limit of 2000 epochs, using early stopping with patience of 20 epochs to prevent unnecessary training and overfitting. The weights have been initialized using glorot uniform approach [26] and the bias terms were initialized to zero.

Figures 5 and 6 presents, respectively, the loss and accuracy of our model for each fold. Solid lines represent the performance in training set while dashed lines represent performance in test set.

Analyzing those figures it is possible to observe that after 200 epochs the loss starts to stabilize while the accuracy presents very small improvements for both, training and test sets. It is also possible to observe that the early stopping approach interrupted the training preventing a possible overfitting.

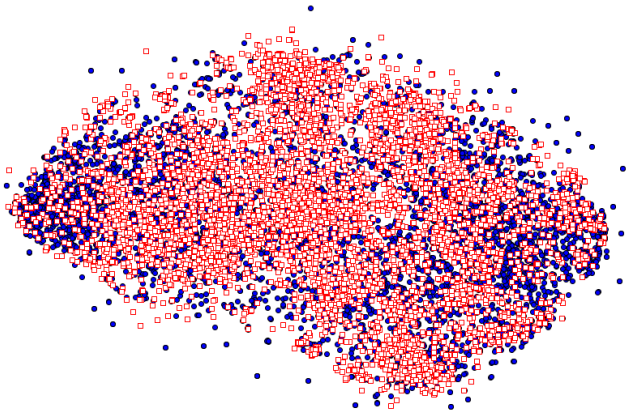
For a better understanding, the results for each fold is also presented in Table I. This table shows that 92.28% average accuracy was achieved with a training time around 184 seconds, and an average of 417 training epochs.

TABLE I
Softmax Accuracy by Folds.

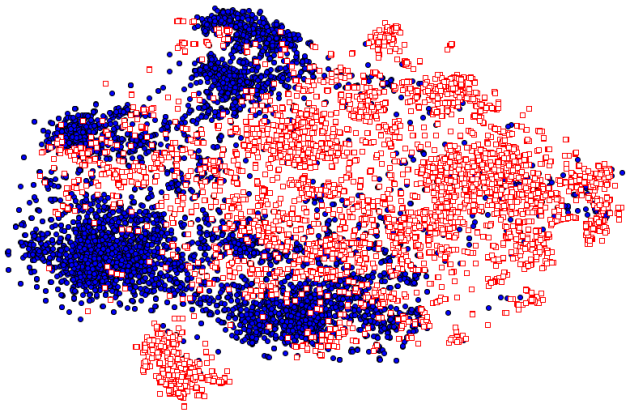
Fold	Accuracy	Epochs	Time (s)
0	0.9284	258	129.31
1	0.9180	360	173.00
2	0.9330	527	221.50
3	0.9144	413	180.52
4	0.9201	525	217.62
Average	0.9228	417	184.39

Additionally, in Table II we present confusion matrix, with and without normalization, showing results for each class.

[4]https://keras.io
[5]https://www.tensorflow.org



(a) Raw image pixels



(b) Bottleneck features

Fig. 4. t-SNE visualization of (a) the raw image pixels and (b) ResNet-50 bottleneck features. Red squares represent PG samples while blue circles represent CG samples.

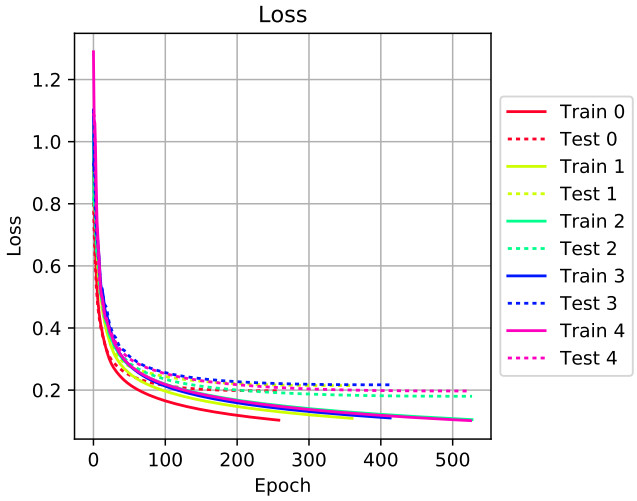


Fig. 5. Softmax 5-fold train/test loss.

F. Classification by SVM Algorithm

Given that the core of proposed method is the transfer learning process, in theory, we can replace the top layer of ResNet-50 by any machine learning classifier. In Section IV-E, for example, we replaced original 1000 fully-connected softmax with a simple 2 fully-connected softmax since we are dealing with a two classes problem. At this section, we evaluate the impact of replacing the top layer by a Support Vector Machine (SVM) classifier [18].

We use an SVM with RBF kernel where the parameters C and gamma have been obtained through a gridsearch process with $C \in \left[10^{-2}, 10^{-1}, ..., 10^{10}\right]$ and $\gamma \in \left[10^{-9}, 10^{-8}, ..., 10^{3}\right]$. The best C obtained was 10.0 and the best γ was 0.001.

Results of each fold are presented in Table III. This tables shows an average accuracy of 94.05% with a training time around 166 seconds.

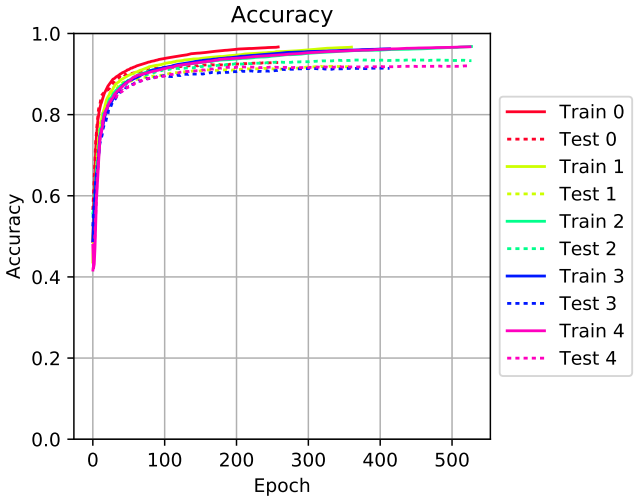


Fig. 6. Softmax 5-fold train/test accuracy.

TABLE II
SOFTMAX CONFUSION MATRIX (A) WITHOUT AND (B) WITH NORMALIZATION

	CG	PG
CG	4472	378
PG	371	4479

(a) Without Normalization

	CG	PG
CG	0.9221	0.0779
PG	0.0765	0.9235

(b) Normalized

TABLE III
SVM ACCURACY

Fold	Accuracy	Time (s)
0	0.9402	167.76
1	0.9345	163.68
2	0.9490	166.15
3	0.9340	167.50
4	0.9448	168.43
Average	0.9405	166.71

Table IV present confusion matrix, with and without normalization, when using SVM as top layer classifier.

TABLE IV
SVM CONFUSION MATRIX (A) WITHOUT AND (B) WITH NORMALIZATION

	CG	PG
CG	4518	332
PG	245	4605

(a) Without Normalization

	CG	PG
CG	0.9315	0.0685
PG	0.0505	0.9495

(b) Normalized

G. Comparative Analysis

In previously sections, we reported results obtained with our deep CNN model using a softmax and an SVM top layer. Our better result of 94.04% is achieved when using an SVM as top layer. The ROC curve and the precision-recall are depicted in Figure 7 and Figure 8, respectively. Also, at this same scenario, we obtained a both with an area under the curve (AUC) of 0.99.

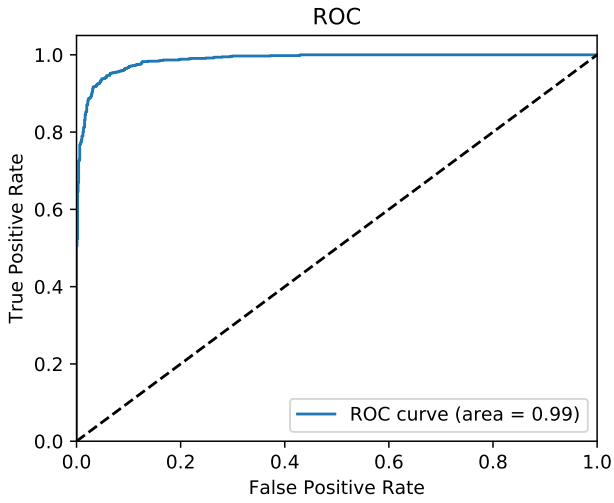


Fig. 7. ROC curve of the SVM classifier

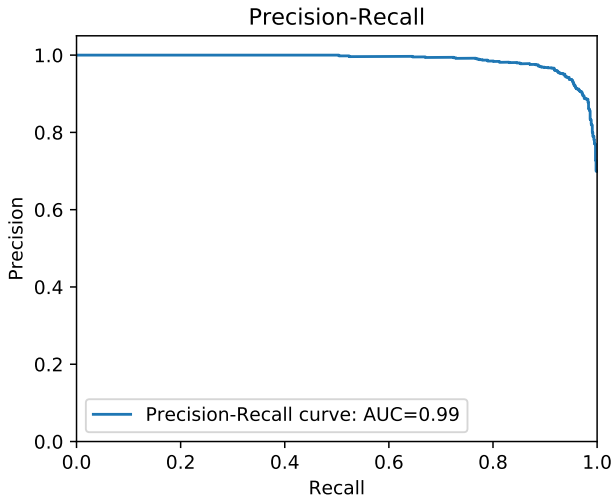


Fig. 8. Precision-recall curve of the SVM classifier

It is important to highlight that, as presented in Figure 9, analyzing the learning curve from the SVM classifier, it is possible observe that the cross-validation score is not still around the maximum. This fact make us believe that adding more training samples could still improve best result.

In *Tokuda et.al.* [2] work, the authors present an extensive comparison between many literature approaches dedicated to solve the problem of detect CG and PG images. The main characteristic of each method investigated by the authors is reported in Table V. Additionally, we included the characteristics of our proposed method

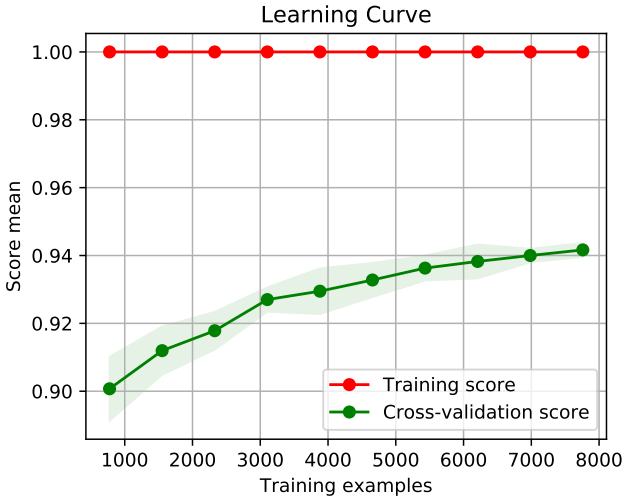


Fig. 9. SVM Learning Curve

using: a deep CNN model stacked with a 2 fully-connected softmax layer at the top (DNN1) and a deep CNN model stacked with an SVM classifier at the top (DNN2).

TABLE V
CONCEPTS USED IN THE RELATED WORKS EVALUATED BY *Tokuda et.al.* [2] AND IN THE METHODS PROPOSED HERE. FOR EACH OF THE FIFTEEN METHODS, IT IS SHOWN THE INDEXES, THE IDENTIFIER USED, THE MAIN CONCEPT USED BY THE METHOD AND THE RELATED FEATURES.

Method	Basis	Feature
Li [27]	Second order differences	Edges/Texture
LSB [28]	Camera noise	Acquisition
LYU [29]	Wavelet transform	Edges/Texture
POP [30]	Interpolator predictor	Acquisition
BOX [31]	Boxes counting	Auto-similarity
CON [32]	Contourlet transform	Edges/Texture
CUR [33]	Curvelet transform [33]	Edges/Texture
GLC [34]	Cooccurrence matrix	Texture
HOG [35]	Histogram of oriented grads	Shape
HSC [36]	Histogram of shearlet coeff	Curves
LBP [37]	Local binary patterns	Edges/Texture
SHE [38]	Shearlet transform	Edges/Texture
SOB [39]	Sobel operator	Edges
DNN1	Deep CNN transfer + Softmax	Raw image pixels
DNN2	Deep CNN transfer + SVM	Raw image pixels

Since our experimental protocol is exactly the same adopted in *Tokuda et.al.* [2], using the same five fold cross-validation protocol over the same dataset, we use results reported by the authors to compare our method with many literature methods. Table VI presents these results. From the table, we see that the accuracies of literature methods have a large range of values going from 0.930 (highest) to 0.552 (lowest). Proposed methods DNN2 overcome all literature methods based on single features and proposed method DNN1 present an average accuracy just lower than Li method [27]. This fact shows the expression power of transfer learning approach in features extraction process.

It is important to highlight that in *Tokuda et.al.* [2], the authors report results for features fusion with average accuracy rates from 0.928 to 0.973. However, even using a single kind of feature, our method performs better than lowest fusion approach. Also, it is

TABLE VI
COMPARISON AMONG APPROACHES FOR DISTINGUISHING CGS AND PGS. TABLE IS SORTED FROM HIGHEST TO LOWEST AVERAGE ACCURACY. FOR EACH OF THE FIFTEEN METHODS, IT IS SHOWN THE NUMBER OF DIMENSIONS OF THE FEATURE SPACE (M), THE ACCURACIES FOR EACH CLASS, AND ITS AVERAGE ACCURACY.

Method	m	CG	PG	Average accuracy
DNN2	**150528**	**0.932**	**0.950**	**0.941**
Li	144	0.948	0.911	0.930
DNN1	**150528**	**0.922**	**0.924**	**0.923**
LYU	216	0.942	0.899	0.920
CON	696	0.918	0.887	0.902
LBP	78	0.904	0.838	0.871
CUR	2328	0.806	0.805	0.805
HSC	96	0.818	0.787	0.802
HOG	256	0.754	0.720	0.740
SHE	60	0.748	0.677	0.713
LSB	12	0.672	0.651	0.662
GLC	12	0.640	0.630	0.635
POP	12	0.570	0.575	0.573
BOX	3	0.541	0.568	0.554
SOB	150	0.554	0.552	0.553

important to highlight that this technique still has a lot of potential to be explored as, for example, fusion of features extracted from different deep CNNs.

V. CONCLUSIONS AND RESEARCH DIRECTIONS

Along this work we presented a new method for CG images detection using a deep convolutional neural network model based on ResNet-50 and transfer learning concepts. After a simple pre-processing, each image in our dataset is feed into our deep CNN model and, as result, we obtain a 2048 dimension feature vector, here called bottleneck features. These feature vectors are used to train machine learning classifiers to detect if an image is, or not, produced by computer graphics methods.

Applying t-SNE dimensionality reduction technique to visualize our high-dimensional features, it is possible to observe that bottleneck features generated by ResNet-50 transfered layers present a higher degree of separability than the raw image input features, which makes the classification task easier.

Also, after different rounds of experiments, is not difficult to conclude that, when compared with methods that take advantage of single features (without fusion or combination) to perform CG images detection, the proposed method performs better than literature methods, showing that extracted bottleneck features present a higher expression power than other hand-crafted feature extraction methods.

Despite does not overcoming all fusion techniques, proposed method present competitive results, overcoming at least one fusion approach.

Analyzing the learning curve from the SVM classifier, it is possible observe that the training score is not still around the maximum. One approach which we intend to investigate in future works is the addition of more samples in training set. With a larger dataset would be possible to fine-tune the deep CNN transferred feature layers, improving the performance of our models.

Finally, as future research directions, we intend to explore the fusion of bottleneck features extracted from different deep CNNs models.

ACKNOWLEDGMENTS

The authors would like to thank the financial support of IFSP-Campinas, FAPESP (grant 2016/21145-5) and CNPq (grants 302923/2014-4, 313152/2015-2 and 423797/2016-6). We also would like to thank the authors *Tokuda et.al.* [2] who helped us with dataset acquirement.

REFERENCES

[1] O. Holmes, M. S. Banks, and H. Farid, "Assessing and improving the identification of computer-generated portraits," *ACM TAP*, vol. 13, no. 2, p. 12, 2016.

[2] E. Tokuda, H. Pedrini, and A. Rocha, "Computer generated images vs. digital photographs: A synergetic feature and classifier combination approach," *Elsevier JVCI*, vol. 24, no. 8, pp. 1276 – 1292, 2013.

[3] D. Dang-Nguyen, G. Boato, and F. G. B. D. Natale, "Discrimination between computer generated and natural human faces based on asymmetry information," in *IEEE EUSIPCO*, 2012, pp. 1234–1238.

[4] D. Dang-Nguyen, G. Boato, and F. G. B. D. Natale, "Identify computer generated characters by analysing facial expressions variation," in *IEEE WIFS*, 2012, pp. 252–257.

[5] H. Farid and M. J. Bravo, "Perceptual discrimination of computer generated and photographic faces," *Digital Investigation*, vol. 8, pp. 226–235, 2012.

[6] V. Conotter, E. Bodnari, G. Boato, and H. Farid, "Physiologically-based detection of computer generated faces in video," in *IEEE ICIP*, 2014, pp. 248–252.

[7] Y. Bengio *et al.*, "Learning deep architectures for ai," *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[8] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[10] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *Springer IJCV*, vol. 115, no. 3, pp. 211–252, 2015.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Adv Neural Inf Process Syst*, 2012, pp. 1097–1105.

[12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE CVPR*, 2015, pp. 1–9.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE CVPR*, 2016, pp. 770–778.

[15] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Adv Neural Inf Process Syst*, 2014, pp. 3320–3328.

[16] H. Farid, "Creating and detecting doctored and virtual images: Implications to the child pornography prevention act," Tech. Rep. TR2004-518, 2004.

[17] D. Q. Tan, X. J. Shen, and H. P. Qin, J.and Chen, "Detecting computer generated images based on local ternary count," *Springer PRIA*, vol. 26, no. 4, pp. 720–725, 2016.

[18] C. M. Bishop, *Pattern Recognition and Machine Learning*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[19] C. V. L. C. University, http://www.cs.columbia.edu/CAVE/, accessed on May 19th, 2017.

[20] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition." in *"ICML"*, vol. 32, 2014, pp. 647–655.

[21] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *"ECCV"*. Springer, 2014, pp. 818–833.

[22] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *arXiv preprint arXiv:1312.6229*, 2013.

[23] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[24] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[25] L. v. d. Maaten and G. Hinton, “”visualizing data using t-sne”,” *JMLR*, vol. 9, no. Nov, pp. 2579–2605, 2008.
[26] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks.” in *Aistats*, vol. 9, 2010, pp. 249–256.
[27] W. Li, T. Zhang, E. Zheng, and X. Ping, “Identifying photorealistic computer graphics using second-order difference statistics,” in *IEEE FSKD*, vol. 5, 2010, pp. 2316–2319.
[28] T.-T. Ng and S.-F. Chang, “Identifying and prefiltering images distinguishing between natural photography and photorealistic computer graphics,” *IEEE SPM*, vol. 26, no. 2, pp. 49–58, 2009.
[29] S. Lyu and H. Farid, “How realistic is photorealistic?” *IEEE TSP*, vol. 53, no. 2, pp. 845–850, 2005.
[30] H. F. A.C. Popescu, “Exposing digital forgeries in color filter array interpolated images?” *IEEE TSP*, vol. 53, no. 10, pp. 3948–3959, 2005.
[31] L. Liebovitch and T. Toth, “A fast algorithm to determine fractal dimensions by box counting,” *Physics Letters A*, vol. 141, pp. 386–390, 1989.
[32] M. Do and M. Vetterli, “Contourlets: a directional multiresolution image representation,” in *IEEE ICIP*, 2002, pp. 357–360.
[33] E. Candes and D. Donoho, *Curvelets A Surprisingly Effective Nonadaptive Representation for Objects with Edges*. Vanderbilt University Press, 2000.
[34] R. Haralick, K. Shanmugam, and I. Dinstein, “Textural features for image classification,” *IEEE SMC*, vol. 3, no. 6, pp. 610–621, 1973.
[35] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *IEEE CVPR*, 2005, pp. 886–893.
[36] W. Schwartz, R. da Silva, L. Davis, and H. Pedrini, “A novel feature descriptor based on the shearlet transform,” in *IEEE ICIP*, 2011, pp. 1053–1056.
[37] T. Ojala, M. Pietikainen, and T. Maenpaa, “A generalized local binary pattern operator for multiresolution gray scale and rotation invariant texture classification,” in *IEEE ICAPR*, 2001, pp. 399–408.
[38] G. Kutyniok and W.-Q. Lim, “Compactly supported shearlets are optimally sparse,” *Elsevier JAT*, pp. 1564–1589, 2011.
[39] R. Gonzalez and R. Woods, *Digital Image Processing*. Prentice-Hall, 2007.