TRACKING MULTIPLE IMAGE SHARING ON SOCIAL NETWORKS

Quoc-Tin Phan [†], *Giulia Boato* [†]

[†] DISI, University of Trento, Trento, Italy

Roberto Caldelli [◇] [⋆], *Irene Amerini* [◇]

[◇] MICC, University of Florence, Florence, Italy
[⋆] CNIT, Parma, Italy

ABSTRACT

Social Networks (SN) and Instant Messaging Apps (IMA) are more and more engaging people in their personal relations taking possession of an important part of their daily life. Huge amounts of multimedia contents, mainly photos, are poured and successively shared on these networks so quickly that is not possible to follow their paths. This last issue surely grants anonymity and impunity thus it consequently makes easier to commit crimes such as reputation attack and cyberbullying. In fact, contents published within a restricted group of friends on an IMA can be rapidly delivered and viewed on a SN by acquaintances and then by strangers without any sort of tracking. In a forensic scenario (e.g., during an investigation), succeeding in understanding this flow could be strategic, thus allowing to reveal all the intermediate steps a certain content has followed. This work aims at tracking multiple sharing on social networks, by extracting specific traces left by each SN within the image file, due to the process each of them applies, to perform a multi-class classification. Innovative strategies, based on deep learning, are proposed and satisfactory results are achieved in recovering till triple up-downloads.

***Index Terms*—** Social networks, image forensics, deep learning, image sharing, multiple up/downloads.

1. INTRODUCTION

The explosion of mobile cameras nowadays triggered tremendous amount of multimedia data and social networks are the privileged channels for their uncontrolled delivery. Nevertheless, due to the increasing popularity of user-friendly editors, image manipulations are carried out easier than ever. Unfortunately, those concerns are not the only side of the story. Recent advances in AI-based technologies enable the generation of artificial visually plausible images that are indistinguishable from real ones. If those tools or technologies are abused, the negative impact of such malicious images could be hardly measurable. In the last decade pioneering methods have been invented to support the validation of images in various aspects for instance: exposing image forgeries [1], identifying the acquisition camera [2], linking images with respect to their acquisition cameras [3], establishing the relationships of near-duplicate images [4]. By the recent advance of deep learning and big amount of data available, several forensic problems can be solved at better performance [5, 6, 7, 8]. Despite the fact that multimedia forensics methods are moving forwards and exposing their potential on in-the-lab data, their robustness is questioned on images coming from SNs, since when uploaded and shared via SN platforms images undergo strong processing such as JPEG compression and resizing, which substantially remove forensic traces [9]. Indeed, to optimize transfer bandwidth as well as display quality, most SNs enforce their own compression/resizing policy, which is generally neither published nor fixed [10]. Those concerns raise the need to identify the SN platform or sharing apps in order to recover partial knowledge about the provenance of the image under investigation. The case of single sharing was first investigated in [11] and later extended in [12] by analyzing the histogram of DCT coefficients. Those works ignored information contained into metadata from JPEG headers, which were shown to be highly distinctive across different platforms [10, 13]. Nevertheless, such information only characterizes the last sharing platform. Traces of multiple JPEG compression instead reside in DCT coefficients [14, 15].

In this work, we tackle the problem of tracing back the history of images shared multiple times over SNs by exploiting traces left in DCT coefficient maps and information from JPEG headers. The final goal is to identify the platform or the chain of platforms that an image has been shared through. The main achievements of the paper are: i) the combined use of image content and metadata based features exploiting a deep learning framework; ii) the detection of the over-SNs path an image follows, dealing with single-double-triple sharing.

The authors gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

2. THE PROPOSED METHODOLOGY

This section introduces the proposed method by firstly describing the adopted features based on DCT coefficients (Sec. 2.1) and on image metadata (Sec. 2.2). Sec. 2.3 describes the way these features are combined within the proposed CNN.

Such CNN, whose architecture is sketched in Fig. 1, takes as input a DCT coefficient map of $B \times B$ (B is multiple of 8) and outputs probability scores over class labels. Two solutions are proposed and compared: one based only on DCT features, named *P-CNN* (Patch-based CNN), and another which takes into account the fusion of DCT and metadata features, named *P-CNN-FF* (Patch-based CNN and Feature Fusion).

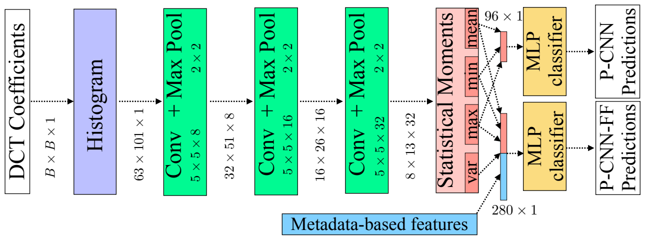


Fig. 1. Architecture of P-CNN and P-CNN-FF.

2.1. DCT coefficient-based features

A typical way to learn statistics of DCT coefficients on each spatial frequency is computing their histogram, which has been successfully incorporated into CNN architectures [16, 17]. Since the number of histogram bins in [16] is equal to the number of Gaussian distributions, computing such histogram requires evaluating multiple Gaussian PDFs of each input value. Due to its computational inefficiency, we adopt the simpler approach in [17], where histogram computation is done merely through convolution and sigmoid activation. Denote D_{c_1,c_2} the $\frac{B}{8} \times \frac{B}{8}$ matrix containing DCT coefficients associated to AC frequency (c_1, c_2) (DC frequency is ignored). The accumulative histogram A_{c_1,c_2} at bin b is computed as:

$$A_{c_1,c_2}(b) = \frac{64}{B^2} \sum_{i,j \in \left[1, \frac{B}{8}\right]} \sigma\left(\gamma(D_{c_1,c_2}(i,j) - b)\right), \quad (1)$$

where $\sigma(\cdot)$ is the sigmoid function. A_{c_1,c_2} is the number of coefficients larger than b. Indeed, if γ is set to a large value (10^6 in [17]), the sigmoid function will output 1 if $D_{c_1,c_2}(i,j)$ is essentially bigger than b and 0 otherwise. The final histogram is obtained by differentiating neighboring bins of $A_{c_1,c_2}(b)$ through 1D convolution with kernel $[1, -1]$. In [17], bin edges b can be tunable during training. In our CNN, however, learning b is infeasible due to gradient vanishing, as $\sigma(\cdot)$ ouput always reaches its extreme, i.e., 0 or 1. Thus, we fix b as increasing integer values from -50 to 50 and obtain an accumulate histogram of 101 bins [11].

Histograms of 63 AC DCT coefficients are then fed to the proposed P-CNN composed by 3 consecutive convolutional layers of receptive field 5×5 with kernels dimension $8, 16, 32$ respectively. Then activation is added after each convolution layer to obtain non linearity, followed by max pooling with stride $[2, 2]$ to reduce the size of feature maps. Afterwards, we incorporate the statistical moments layer described in [18] to extract 4 statistical values (max, min, mean and variance) of each of 32 feature maps. This layer significantly reduces dimensionality of feature maps, and stabilizes the training. Since training with both variance and mean computation might lead to instabilities, thus we then decided not to use variance. Subsequently, max, min and mean values of each of feature maps are concatenated in a vector of 96 features (3×32) and fed to Multilayer Perceptron (MLP) classifier (two hidden layers of $128, 64$ hidden units and one output layer of K output units), and probabilities over K classes are computed through K-way softmax (see Fig. 1). P-CNN is trained up to 100 epochs using Adam optimizer. The initial learning rate is set to 10^{-4} and exponentially decayed after 50 epochs to improve convergence.

2.2. Metadata-based features

The analysis in [13] has shown that information from JPEG header are strongly discriminant for identifying the last sharing platform. As SNs employ their own JPEG compression and resizing, quantization tables and image size can be considered as useful cues. Furthermore, specifications of encoding process also present peculiarities of SN platform. Following [13], we have extracted this set of 152 metadata-based features:

- *Quantization tables (128 features):* quantization tables of luminance and chrominance channels reflecting the quality factor of the (last) JPEG compression.
- *Huffman encoding tables (2 features)*: number of encoding tables used for AC and DC component.
- *Component information (18 features):* 6 features, each for Y-Cb-Cr, describe component id, horizontal/vertical sampling factors, quantization table index, AC/DC coding table indices.
- *Optimized coding and progressive mode (2 features):* binary values indicating the use of these two modes.
- *Image size (2 features):* min and max image dimension.

2.3. Feature fusion

To leverage the distinctive power of deep features of P-CNN and metadata-based features, we use the CNN to extract the 4 statistical values (max, min, mean, variance) of each of 32 feature maps so it yields $4 \times 32 = 128$ features available for each image patch, as shown in Fig. 1. Although variance is treated as constant during P-CNN training due to instability, once P-CNN is optimal, these features encode the variability of CNN feature maps. Thus, the P-CNN-FF variant, exploiting both types of features, use 128 deep features (4×32) concatenated with 152 metadata-based features, thus working on a 280-d feature array. The final classifier is simply a MLP classifier followed by K-way softmax. P-CNN-FF is trained on concatenated features for 100 epochs using Stochastic Gradient Descent with momentum 0.9, initial learning rate 10^{-2}.

The learning rate is divided by 5 if the training loss does not decrease after two consecutive epochs.

3. TEST SET-UP AND EXPERIMENTAL RESULTS

This section describes the experimental set-up in terms of carried out tests and adopted datasets (Sec. 3.1) and achieved results in different application scenarios (Sec. 3.2 and 3.3).

3.1. Datasets and test configurations

In order to evaluate the classification performances of the proposed techniques in tracking the different path followed by an image on various SNs, two different datasets have been built [1]. The first one, generated by the RAISE dataset [19], is named *R-SMUD* (RAISE Social Multiple Up-Download) and contains images shared over three SNs: *Facebook* (FB), *Flickr* (FL) and *Twitter* (TW). 50 images (raw format) have been selected from RAISE and cropped on top-left corner at sizes: 377×600, 1012×1800 and 1687×3000 with an aspect ratio of $9 : 16$. All cropped images are subsequently JPEG compressed (the independent JPEG group's software has been adopted) at quality factors $QF = 50, 60, 70, 80, 90, 100$. This yields to 900 images in total, shared at maximum 3 times through the three platforms. By considering all the possible combinations with repetitions, we get $Comb = \sum_{k=1}^{J} (SN)^k$, where SN represents the number of social networks ($SN = 3$) and J indicates the maximum number of sharing (e.g., for $J = 3$ it yields $Comb = 3^1 + 3^2 + 3^3 = 39$).

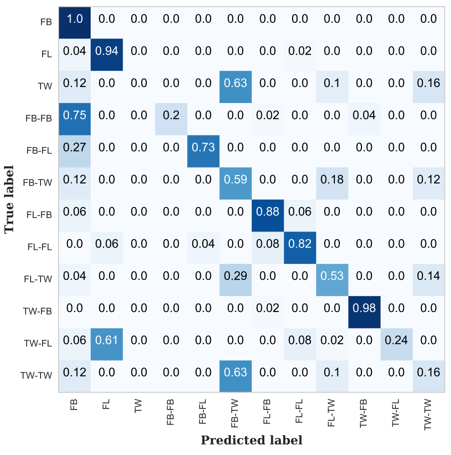
The second dataset, called *V-SMUD* (VISION Social Multiple Up-Download) consists of 510 JPEG images from VISION dataset [9] shared via FB, FL, TW, at maximum 3 times, following different testing configurations.

For each dataset, three testing configurations have been evaluated by varying the number of considered up-downloads and, consequently, of the number of classes to be recognized:

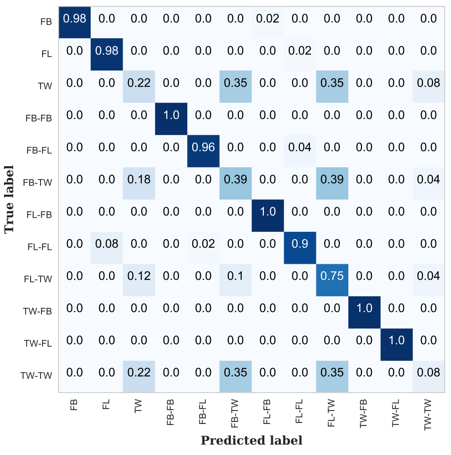
- C1: images shared once ($J = 1$) via FB, FL, TW resulting in a *3-class* classification problem.
- C2: images in C1 plus their double sharing instances ($J = 2$), resulting in a *12-class* classification problem.
- C3: images in C2 plus their triple sharing instances ($J = 3$), resulting in a *39-class* classification problem.

Each class will correspond to the exact path followed by an image from a single bounce to a double bounce (e.g., FB-FL) till a triple bounce (e.g., FB-TW-FB, where the image uploaded on *Facebook* is then shared on *Twitter* and after the download it is shared again on *Facebook*). The proportion among training, validation and test images is 80%, 10% and 10% respectively, while image patches of 64×64 pixels are used as both P-CNN and P-CNN-FF input. Scene-disjoint splitting strategy has been employed to avoid images of the same scene appearing simultaneously in the training, validation and test set.

[1]Datasets can be downloaded at http://loki.disi.unitn.it/R-V-SMUD/



(a) P-CNN



(b) P-CNN-FF

Fig. 2. Confusion matrices (image level) on V-SMUD dataset for configuration C2 for (a) P-CNN and (b) P-CNN-FF .

3.2. Experiments on single and double up/download

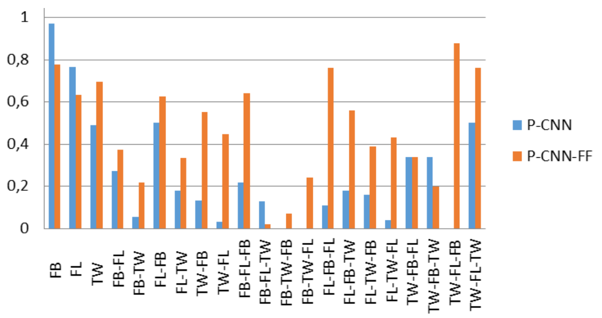
In this subsection experimental results for the cases of single and double up/download (C1 and C2 respectively) are presented in comparison with two state-of-the-art methods [11] and [12]. In Table 1, the different achieved performances are listed in terms of accuracy for 2 levels of analysis: *patch level* and *image level*. Image-level accuracy is obtained by majority voting from all image patches. It can be observed that the P-CNN approach, though providing satisfactory results, does not succeed in outperforming previous methods (except for the V-SMUD dataset at *image level*) while the P-CNN-FF one improves all other methods achieving very good results, providing a satisfactory accuracy also for C2. By looking at Fig. 2, the improvement achieved with P-CNN-FF with respect to

Table 1. Performance (Accuracy %) comparison on C1 and C2.

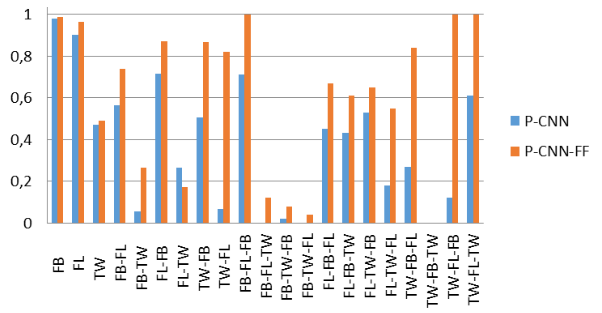
Method	*R-SMUD*				*V-SMUD*			
	Patch level		*Image level*		*Patch level*		*Image level*	
	C1	C2	C1	C2	C1	C2	C1	C2
[11]	-	-	93.70	39.91	-	-	90.20	46.73
[12]	93.25	51.38	94.81	45.18	92.56	60.22	98.69	54.90
P-CNN	85.63	45.35	89.63	43.24	85.84	53.79	100.00	58.82
P-CNN-FF	**99.87**	**73.19**	**99.87**	**65.91**	**100.00**	**81.97**	**100.00**	**77.12**

Table 2. Performance (Accuracy %) comparison on C3.

Method	*R-SMUD*		*V-SMUD*	
	Patch	*Image*	*Patch*	*Image*
[11]	-	17.29	-	23.68
[12]	17.23	16.95	17.95	16.41
P-CNN	20.91	19.32	26.96	27.10
P-CNN-FF	**43.41**	**36.18**	**52.68**	**49.72**



(a) R-SMUD



(b) V-SMUD

Fig. 3. Diagonal of the confusion matrix for C3 on (a) R-SMUD and (b) V-SMUD (same consecutive SNs aggregated).

P-CNN is evident and the classification is very accurate if we do not consider the classes when Twitter is the final social network (see Fig. 2(b)). Furthermore, another interesting result is obtained if we consider only the detection of the final social network, that is if we take as correct decision when images, independently from the followed path over the SNs, are rightly classified according to the last social network; looking at the confusion matrices in Fig. 2, it is possible to achieve an accuracy of 92% with P-CNN and 100% with P-CNN-FF (results on V-SMUD only are reported for sake of conciseness).

3.3. Experiments on triple up/download

In this subsection, results obtained in the more challenging case where a triple up/download can also happen (C3 with 39 classes) are presented. By looking at Table 2, it can easily be grasped that performances drastically decrease, though the proposed P-CNN-FF method definitely provides the highest accuracy. However, if we give a closer look within the confusion matrices, we can highlight some interesting achievements. In fact, if we aggregate some of the 39 classes following the criterion that consecutive up-downloads on the same SN do not affect the image (i.e., FB-FB-FL corresponds to class FB-FL, or FL-TW-TW corresponds to FL-TW), accuracy becomes again satisfactory, as depicted in Fig. 3. The values obtained on the main diagonal of the aggregated confusion matrices are represented for the P-CNN and P-CNN-FF methods with respect to the two datasets. On the basis of the adopted criterion, the aggregated classes are now 21 and an overall accuracy of 60.6% is achieved for P-CNN-FF with respect to the V-SMUD dataset with a gain of around 8% respect to the no-aggregated case. Moreover, if we analyse only the detection of the final SN as in Sec. 3.2, we can appreciate the same behaviour obtaining again around 100% accuracy using the P-CNN-FF method for both datasets. So the introduction of the proposed P-CNN-FF method leads to the possibility to evidence the paths followed by an image on SNs satisfactorily up to three shares respect to related works and it also permits to obtain an almost perfect classification related to the detection of the last SN of the chain.

4. CONCLUSIONS

In this paper two different CNN methods have been introduced to detect multiple up-download of an image over SNs. Up to triple sharing has been taken into account and two new datasets have been introduced. The obtained results demonstrated a good ability of the proposed CNN-based approach with features fusion to distinguish among different social platforms determining the exact path the image has followed. In particular, the combined use of image content and metadata based features demonstrate its superiority respect to state of the art techniques. Future works will be devoted to investigate the specific behavior of some SNs and to improve the accuracy for the triple shares configuration.

5. REFERENCES

[1] H. Farid, “A survey of image forgery detection,” *IEEE Signal Processing Magazine*, vol. 2, no. 26, pp. 16–25, 2009.

[2] J. Lukas, J. Fridrich, and M. Goljan, “Digital camera identification from sensor pattern noise,” *IEEE Trans. on Information Forensics and Security*, vol. 1, no. 2, pp. 205–214, 2006.

[3] M. Goljan, M. Chen, and J. Fridrich, “Identifying common source digital camera from image pairs,” in *IEEE Int. Conference on Image Processing*, 2007, vol. 6, pp. VI – 125–VI – 128.

[4] Z. Dias, A. Rocha, and S. Goldenstein, “Image phylogeny by minimal spanning trees,” *IEEE Trans. on Information Forensics and Security*, vol. 7, no. 2, pp. 774–788, 2012.

[5] B. Bayar and M. C. Stamm, “A deep learning approach to universal image manipulation detection using a new convolutional layer,” in *Proc. of the 4th ACM Workshop on Information Hiding and Multimedia Security*, 2016, pp. 5–10.

[6] L. Bondi, L. Baroffio, D. Gera, P. Bestagini, E. J. Delp, and S. Tubaro, “First steps toward camera model identification with convolutional neural networks,” *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 259–263, 2017.

[7] D. Cozzolino and L. Verdoliva, “Camera-based image forgery localization using convolutional neural networks,” in *Proc. of European Signal Processing Conference*, 2018.

[8] M. Huh, A. Liu, A. Owens, and A. A. Efros, “Fighting fake news: Image splice detection via learned self-consistency,” in *Proc. of The European Conference on Computer Vision*, 2018.

[9] D. Shullani, M. Fontani, M. Iuliani, O. A. Shaya, and A. Piva, “VISION: a video and image dataset for source identification,” *EURASIP Journal on Information Security*, vol. 2017, no. 1, pp. 15, 2017.

[10] Oliver Giudice, Antonino Paratore, Marco Moltisanti, and Sebastiano Battiato, “A classification engine for image ballistics of social data,” in *Proc. of Image Analysis and Processing*, 2017, pp. 625–636.

[11] R. Caldelli, R. Becarelli, and I. Amerini, “Image origin classification based on social network provenance,” *IEEE Trans. on Information Forensics and Security*, vol. 12, no. 6, pp. 1299–1308, 2017.

[12] I. Amerini, T. Uricchio, and R. Caldelli, “Tracing images back to their social network of origin: A CNN-based approach,” in *Proc. of IEEE Workshop on Information Forensics and Security*, 2017, pp. 1–6.

[13] Q.-T. Phan, C. Pasquini, G. Boato, F. G. B. De Natale, “Identifying image provenance: an analysis of mobile instant messaging apps,” in *Proc. of IEEE 20th Int. Workshop on Multimedia Signal Processing*, 2018.

[14] A. C. Popescu and H. Farid, “Statistical tools for digital forensics,” in *Information Hiding*, vol. 3200 of *Lecture Notes in Computer Science*, pp. 128–147. Springer Berlin Heidelberg, 2005.

[15] T. Pevny and J. Fridrich, “Detection of double-compression in JPEG images for applications in steganography,” *IEEE Trans. on Information Forensics and Security*, vol. 3, no. 2, pp. 247–258, 2008.

[16] V. Sedighi and J. Fridrich, “Histogram layer, moving convolutional neural networks towards feature-based steganalysis,” *Electronic Imaging*, vol. 2017, no. 7, pp. 50–55, 2017.

[17] M. Barni, L. Bondi, N. Bonettini, P. Bestagini, A. Costanzo, M. Maggini, B. Tondi, and S. Tubaro, “Aligned and non-aligned double JPEG detection using convolutional neural networks,” *Journal of Visual Comunication and Image Representation*, vol. 49, no. C, pp. 153–163, Nov. 2017.

[18] C. F. Tsang, J. Fridrich, “Steganalyzing images of arbitrary size with CNNs,” in *Proc. of Media Watermarking, Security, and Forensics*, 2018.

[19] D.-T. Dang-Nguyen, C. Pasquini, V. Conotter, and G. Boato, “RAISE: A raw images dataset for digital image forensics,” in *Proc. of the 6th ACM Multimedia Systems Conference*, 2015, pp. 219–224.