Unsupervised JPEG Domain Adaptation for Practical Digital Image Forensics

Rony Abecidan, Vincent Itier, Jérémie Boulanger, Patrick Bas

Unsupervised JPEG Domain Adaptation for Practical Digital Image Forensics

Rony Abecidan
Univ. Lille, CNRS, Centrale Lille,
UMR 9189 CRIStAL,
F-59000 Lille, France
Email: rony.abecidan@univ-lille.fr

Vincent Itier
IMT Lille-Douai, Institut Mines-Télécom,
Centre for Digital Systems,
Univ. Lille, CNRS, Centrale Lille,
UMR 9189 CRIStAL,
F-59000 Lille, France
Email: vincent.itier@imt-lille-douai.fr

Jérémie Boulanger and Patrick Bas
Univ. Lille, CNRS, Centrale Lille,
UMR 9189 CRIStAL,
F-59000 Lille, France
Email: {jeremie.boulanger,patrick.bas}
@univ-lille.fr

***Abstract*—Domain adaptation is a major issue for doing practical forensics. Since examined images are likely to come from a different development pipeline compared to the ones used for training our models, that may disturb them by a lot, degrading their performances. In this paper, we present a method enabling to make a forgery detector more robust to distributions different but related to its training one, inspired by [1]. The strategy exhibited in this paper foster a detector to find a feature invariant space where source and target distributions are close. Our study deals more precisely with discrepancies observed due to JPEG compressions and our experiments reveal that the proposed adaptation scheme can reasonably reduce the mismatch, even with a rather small target set with no labels when the source domain is properly selected. On top of that, when a small portion of labelled target images is available this method reduces the gap with mix training while being unsupervised. All our experiments are available at** *https://bit.ly/UJDA-WIFS2021* .

I. Introduction

In the growing context of fake news, digital images are easily tampered in order to change their meaning. Main malicious image manipulation are copy-paste, copy-move and inpainting. Copy-paste detection focuses on retrieving two different noise distributions which are dependent of the image acquisition chain. Basic inpainting methods tend to not reproduce the image intrinsic noise. Whereas, copy-move operation duplicates a part of the image. Nevertheless, it may leave traces due to resampling, rescaling or rotation transformation. Usually, after forgery, some post processing operations are done. The goal is twofold, make the detection more arduous, for instance with smoothing or sharpening, and share them easily *i.e.* with compression. Lossy compression, by nature, is seen as a counterattack upon detection. State-of-the-art methods for forgeries detection [2], [3], [4] rely on deep learning which requires a huge amount of annotated data. These kind of dataset are very cumbersome to produce or to annotate.

If multimedia forensics schemes can be very effective at detecting image tampering or localizing tampered areas [3], [4], they are very often extremely sensitive to the very nature

WIFS'2021, December, 7-10, 2021, Montpellier, France. XXX-X-XXXX-XXXX-X/XX/$XX.00

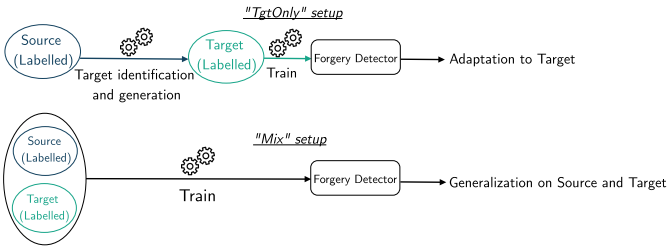

Fig. 1. Principles of the *TgtOnly* and *Mix* approaches (supervised).

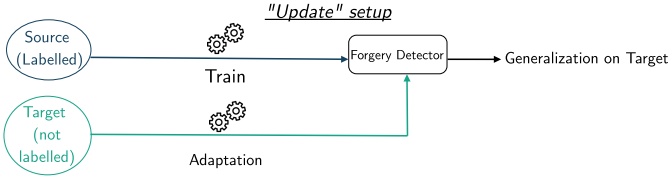

Fig. 2. Principle of the *Update* approach (unsupervised).

of the analyzed signal. For image forensics schemes relying on machine learning, this means that if the image database used for training (a.k.a. the ***source***) does not exactly undergo the same development pipeline as the scrutinized test images (a.k.a. the ***target***), the performance of the detector can then be jeopardized.

To illustrate this point, we implemented a forgery detector largely inspired by [4] and observed its performances in different contexts. To construct our source and target domains, we considered two independent set of images coming from the splicing category of the database *DEFACTO* [5]. All the details about our training protocol are explained in III. We notably trained this detector on different sources before evaluating it on a common target so that, we can disclose the influence of the training from the source on the performances on the target.

Table I shows the performances of our forensics detector on images compressed at a JPEG quality factor of 5% when the training set comes either from uncompressed images or from images also compressed at a JPEG quality factor of 5%. On this example, we can see that the impact on the performance

	Training QF5	Training uncompressed
Test QF5	82.9%	65.7%

TABLE I
EFFECT ON ACCURACY FROM SOURCE AND TARGET DOMAIN MISMATCHES FOR TWO DIFFERENT JPEG QF (THE FORENSICS SETUP IS DESCRIBED IN SECTION III).

is substantial with a loss of accuracy of more than 17%.

This problem of mismatch, a.k.a. *cover-source mismatch* is known in steganalysis [6], [7], and it can be mitigated using different strategies that depend of the practical context. We describe them in what follows using the same terminology proposed in [8], [9], see also Figs 1 and 2:

- If the set of images from the target can be associated with another training-set containing both genuine and forged images originating from the same distribution, then the learning problem becomes supervised. Usually in forensics, this scenario is not very realistic since it can be extremely time-consuming to generate forged images. In this supervised context, *TgtOnly* setup, *i.e.* re-training using the specific target set instead of using the source (coined as the *SrcOnly* setup), can be performed. If the size of the Target set is small, weak supervised learning can also be applied [10].

- Another strategy can also be used to counter the effect of heterogeneity if the number of sources is sufficiently small, denoted as the *mix* setup, *i.e.* training with all the different sources from the beginning or to augment the dataset using adversarial contents.

Note that these two strategies have been used for example for practical steganalysis, forensics or computer vision in order to cope with numerous processes [7], [11], [12], [13].

- However, if the test images cannot be associated with any training set, which is the case when the development pipeline cannot be identified, then the problem is unsupervised and unsupervised domain adaptation [14] has to be considered. We believe that this class of methods is particularly suited for the forensics analyst since the practitioner usually has to analyze a possibly small set of images belonging to the same unknown domain. In this case, the classifier trained on the source domain needs to be directly modified to consider the target domain, we call it the *Update* setup (see Fig. 2).

We review briefly few schemes that consider unsupervised domain adaptation for IFS related problems.

To detect face spoofing when the heterogeneity of the domain is due to acquisition and illumination, [15] proposes to perform subspace alignment, i.e. to compute the linear mapping between the eigenvectors of the source and target domain by minimizing the Maximum Mean Discrepancy [16]. A similar idea is proposed in [17] when we want to detect image resizing operations. Here a set of mappings computed by Riemannian Procrustes Analysis [18] is derived to transform the covariance matrices derived from mixture of Gaussian models. For audio steganalysis, the use of adversarial training was proposed to perform domain adaptation [19] when the recording is performed by different devices. Following the methodology of [20], a discriminator differentiating the source and target domain is trained from the training and testing sets and then used during the learning process. Here again, the number of test contents needed to train the discriminator is substantial.

This paper proposes to perform unsupervised domain adaptation for deep-learning based image forensics schemes, updating the network trained on the source set using a back-propagation mechanism. Similarly to what was proposed in [21] but in an unsupervised setting, the investigated source and target come from JPEG coding. Note that double compression artefacts prevent the automatic update from the source domain to the target domain. The adaptation scheme, relying on [1], updates the weights of the network by minimizing conjointly the binary cross entropy loss computed with the source labels and, the MMDs between the embeddings of the source and target distributions at each dense layer. One benefit of this approach is that the adaptation of the network does not rely on an additional network to be trained, but solely on a few adaptation parameters σ_l (see Section II).

Section III presents the forensics setup, *i.e.* the CNN detector and the different databases. Finally, section IV presents different results, including an analysis of the effect of the adaptation parameter, the impact of the size of the target domain, and the choice of the source domain. A comparison with supervised approaches is also performed.

II. UNSUPERVISED DOMAIN ADAPTATION

We consider a source dataset $\mathcal{D}_s = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^{n_s}$ made of n_s labeled observations and a target dataset $\mathcal{D}_t = \{\mathbf{x}_j^t\}_{j=1}^{n_t}$ made of n_t unlabeled observations. The distribution that generated the observations from the source dataset is denoted p_s and the distribution that generated the observations from the target dataset is denoted p_t. We suppose that $p_s \neq p_t$ while being "reasonably" close. A common goal in domain adaptation consists in designing a classifier f :

- able to embed the two domains in a feature space $\mathcal{F}$ within which we can't consider them as coming from two different distributions. In that case, we say that the features obtained are domain invariant,
- which minimizes the target risk:

$$\begin{aligned}\mathcal{R}_t(f) &= \mathbb{E}_{(\mathbf{x},y)\sim p_t}(\mathbb{1}_{f(\mathbf{x})\neq y}) \\ &= \mathbb{P}_{(\mathbf{x},y)\sim p_t}[f(\mathbf{x}) \neq y].\end{aligned} \tag{1}$$

Let $\Phi_{\mathcal{F}}$ be a mapping which allows us to obtain relevant features from the source and target domains. Finding a proper *feature mapping* $\Phi_{\mathcal{F}}$ is not an easy task but recent approaches show that it is possible to learn it using a neural network. In this context, the feature mapping is learned so that the domain adaptation is the most effective possible.

Long *et al.* [1] proposed an architecture in which the last dense layers of a CNN are forced to embed the two domains in feature spaces where their distributions are similar using the Maximum Mean Discrepancy [16] (MMD), a famous kernel-based metric enabling to judge to what extent two distributions look similar. To enforce adaptation, their idea consists in

adding to the classification loss, an adaptation loss which is the sum of MMDs between the source and target distributions embedded at each dense layer. This additional loss can be seen as a way to regularize the model so that it does not rely too strongly on the source distribution and is defined as:

$$\mathcal{L} = \sum_{\substack{i=1,\dots,n_s \\ \text{domain=source}}} \mathcal{L}_{\text{classification}}(\Phi_{\mathcal{F}}^{final}(x_i^s), y_i^s) \qquad (2)$$
$$+ \lambda \sum_{\text{layer } l=1,\dots,\text{L}} \left[\sum_{\substack{i=1,\dots,\min(n_s,n_t), \\ \text{domain=source + target}}} \text{MMD}^2(\Phi_{\mathcal{F}}^l(x_i^s), \Phi_{\mathcal{F}}^l(x_i^t)) \right],$$

where $\Phi_{\mathcal{F}}^l$ is the feature map obtained at the level of the layer l in the network and $\lambda > 0$ is a regularization parameter. It is important to know that the MMD is a revelant metric for estimating the discrepancy between two distributions only if it is associated to a *characteristic kernel*, in which case it is minimized if and only if the two distributions are identical. That's why the authors proposed to use a convex combination of characteristic kernels for computing the MMD ensuring that their resulting kernel is also characteristic. The MMD is interesting to consider since it is easy to estimate it using samples from our distributions. Indeed, considering samples $(X_i)_{1\leq i\leq n} \sim \mathcal{P}$ and $(Y_j)_{1\leq i\leq m} \sim \mathcal{Q}$, an unbiased estimator of the MMD between $\mathcal{P}$ and $\mathcal{Q}$ is given by :

$$\widehat{MMD}_k^2(\mathcal{P},\mathcal{Q}) = \frac{1}{n(n-1)} \sum_{i\neq j=1}^{n} k(X_i, X_j) + \frac{1}{m(m-1)} \sum_{p\neq q=1}^{m} k(Y_p, Y_q) - 2\frac{\sum_{i=1}^{n}\sum_{j=1}^{m} k(X_i, Y_j)}{mn}$$

With that expression we deduce that a differentiable kernel enables to make the MMD differentiable and hence, the gradient descent algorithm can be used for backpropagation. It is precisely the case of the gaussian kernel :

$$k(X,Y) = e^{-\frac{\|X-Y\|^2}{2\sigma^2}} \text{ with } X, Y \in \mathbb{R}^d, \sigma > 0$$

The authors of [1] used a combination of gaussian kernels with varying bandwiths σ against standard domain adaptation datasets and, they obtained state of the art results for image classification .

III. Proposed method

For our experiments, we choose to use the well-known forgery detector proposed by Bayar and Stamm [4]. It is a simple architecture that is efficient on standard databases and satisfying to demonstrate the benefits of the proposed strategy. As it can be observed in Fig. 4, this detector has a classical architecture (Conv + Maxpool + Linear Layers) except for the first convolution which comply with the specific constraint :

$$\begin{cases} \boldsymbol{w}_k^{(1)}(0,0) = -1, \\ \sum_{m,n\neq 0} \boldsymbol{w}_k^{(1)}(m,n) = 1. \end{cases}$$

This constraint is applied to the very first convolutional layer and fosters the extraction of relevant low-level forensic features. The other layers are non-constrained and act as usual. The convolutions help to extract high-level manipulation features and, the linear layers generate the classification output. The architecture of our detector is presented in Fig. 3.

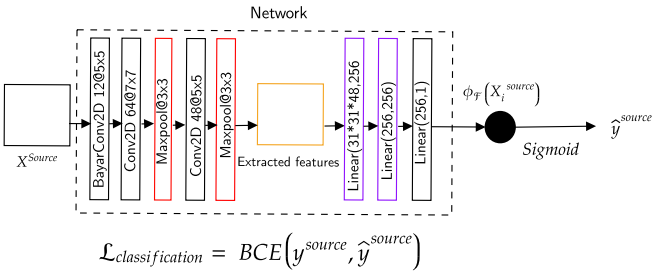

Fig. 3. Architecture of our forgery detector largely inspired by [4], in red: Batch Norm + ReLU, in orange: Flatten, in purple: ReLU + Dropout(0.5).

Concerning the domain adaptation, we compute the sum of the MMDs between the final embeddings $(\Phi_{\mathcal{F}}^l)_{l=1,\dots,3}$ of our images from the source and the target, at the level of each final layer. This metric acts as a regularization term on the cost function. Minimizing it stimulates the network to find meaningful embeddings for the forgery detection in our two domains. To keep things simple, for each experiment the same Gaussian kernel is used for the computation of the MMDs. However, we test the adaptation with different bandwidths (denoted σ) since this parameter has a great impact on the learning phase . An illustration of our domain adaptative strategy is available in Fig. 4. The regularization parameter λ from eq. (2) is set to 1.0 as its role is somehow redundant with the bandwidth σ.

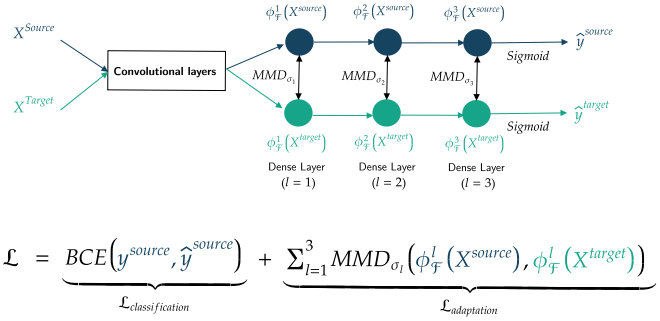

Fig. 4. Illustration of our domain adaptation strategy.

A. Construction of the databases

We worked with all the images from the "Splicing" category of the public base *DEFACTO* [5], which has been specifically designed for tampering detection. It offers a large variety of forgery and has been generated using the *MS COCO* database [22] which was designed for detection, segmentation and captioning. The main assets of the *DEFACTO* database are its large size and the high quality of the images (their size and their realism). We focus on splicing forgeries since it is the most challenging category. Spliced images are made by

inserting objects that do not depend too much of the point of view. The authors also take care of the position of the splicing to avoid objects overlapping. Finally, images are saved in TIF format that allows us to control splicing post processing such as JPEG compression. To construct the source and target bases for the following experiments, we split these images into two equally-sized independent sets . Since each image has a specific size, we cut them into 128×128 patches. This ensures a simple training phase in constant time.

For each patch from the source and target base, the "forged" class is constructed from the 128×128 patches which have a tampered surface ratio between 20% and 80% of the total surface. Selecting this range of tampered surfaces is not innocuous since on one hand, spotting a very small tampered area could be very difficult for the detector and, on the other hand, in the situation where a significant portion of the patch is tampered, it is tricky to assume any forgeries as the detector will not be able to spot two modes in the feature distribution.

Based on the number of "forged" patches, "genuine" patches are then selected in equal quantity to constitutes balanced classes. In the case where JPEG compression are applied, it is realized on the images themselves before cutting them into patches to prevent artifacts.

B. Experimental protocol

In order to obtain more accurate results about the generalization of our detector on the different bases, we design a 3-fold cross validation for each experiment. This process is applied both to the source and the target at the same time. For each fold, we have a cut for the source into train/test with the proportion $\frac{2}{3}/\frac{1}{3}$. The same processing is applied to the target base, with other patches, potentially coming from a different distribution compared to the source.

To get the best of our model and avoid overfitting, we consider for each experiment an early stopping callback based on the accuracies obtained on the source test sets and, observed the results on the target test sets. We can consider that the source test sets play the role of the traditional validation sets meanwhile, our target test sets, the role of the genuine test sets, independent of the training phase.

The following choices were made concerning the optimization strategy and the hyperparameters:

- The maximal number of epochs is fixed at 30, a reasonable amount of epochs enabling to observe a convergence in practice.
- The optimizer is the Adam one, famous for its efficiency in general.
- The batch size is fixed at 128, a reasonable size for computation on a regular GPU while ensuring a good convergence of our detector.
- The learning rate is fixed to 10^{-4}
- The initialization of our weights is the one by default on pytorch [23]. For each study, we initialized our forgery detector with the common seed 2021 in order to make our results reproductible and ensuring a fair comparison of our results.

IV. Experimental results

In this section, we present and interpret the results of various experiments to assess to what extent the presented strategy helps to achieve domain adaptation. Note that in such an unsupervised setup, we have not access to the target labels and simulate the *TgtOnly* and *Mix* setups to underline the maximal accuracies achievable on the target.

A. Image forgeries detection after JPEG compression

JPEG compression efficiency relies on the quantization of high-frequency coefficients in the DCT domain. This transformation is usually done after image forgeries to hide the malicious manipulation. In this section, we observe the impact of this type of transformation on the performance of the detector, and its capability of generalizing over different JPEG quality factors.

To illustrate that point we consider a first experiment where we train our forgery detector on our *uncompressed* source images without performing any adaptation just like in Fig. 3. Then, we look at the performance of this detector with several targets related to different quality factors.

Table II illustrates that datasets built with specific JPEG quality factors have different underlying distributions on which the detector focuses for achieving his task. We can clearly see here that applying JPEG compressions on our images is sufficient for disturbing seriously the detector and this behaviour is observed, even when we compress with a factor of 100%. That little fact on its own illustrates the need of domain adaptation.

Quality factor	QF5	QF10	QF20	QF50	QF100	None
Target accuracy	65.7%	74.9%	81.8%	84.2%	84.5%	91.8%
Standard deviation	1%	2%	1%	2%	3%	1%

TABLE II
EFFICIENCY OF OUR DETECTOR WITHIN THE SRCONLY SETUP ON SEVERAL TARGETS CREATED BY APPLYING JPEG COMPRESSIONS (3-FOLD CROSS-VALIDATION)
(TRAIN) $N_{source} \sim N_{target} \sim 18,000$ PATCHES.
(TEST) $N_{source} \sim N_{target} \sim 9,000$ PATCHES.

B. JPEG quality factor adaptation

Models built for image tampering detection are designed to look for the presence of two different distributions in the image. The feature extraction step done by the CNN part is conditioned by the JPEG quality factor as seen before.

The biggest gap observed previously is, without surprise, between an uncompressed target and a compressed target with a quality factor QF5. Hence, we decide to consider the adaptation from an uncompressed source to a compressed target with a quality factor of 5%. We present in Table III our results with our domain adaptative strategy for this precise case varying the bandwidth σ.

When $\sigma = 0.01$, the adaptation is so constraining that the forgery detector becomes inefficient. In that case, our learning phase leads the detector to predict everytime the same class so that it will perform exactly similarly on the source and the target. It is the easiest option but also the less satisfying one.

	SrcOnly	$\sigma = 0.01$	$\sigma = 8$	$\sigma = 1000$	TgtOnly
Target Accuracy	**65.7%**	56.8%	76.9%	66.2%	**82.9%**
Std	1%	1%	<1%	1%	<1%

TABLE III
ADAPTATION FROM AN UNCOMPRESSED SOURCE TO A TARGET COMPRESSED TO QF5 : STUDY OF THE IMPACT OF THE PARAMETER σ (THE BANDWIDTH OF THE GAUSSIAN KERNEL) ON ACCURACY. (3-FOLD CROSS-VALIDATION)
(TRAIN) $N_{source} \sim N_{target} \sim 18,000$ PATCHES.
(TEST) $N_{source} \sim N_{target} \sim 9,000$ PATCHES.

On the contrary, when $\sigma = 1000$, we are so flexible in the adaptation that the behaviour of the forgery detector is becoming more or less the same than when no adaptation is performed.

However, for $\sigma = 8$, the change becomes significant (+11.2%) and shows that our strategy may indeed improve performances on the target given that σ is carefully tuned. In practice we tested several σ and $\sigma = 8$ was actually the best candidate found in our case.

That being said, we see that we are far from the performance we would have achieved in a supervised setting where target labels are available. Moreover, it's important to precise that improving the performance on the target results often in a decrease in performance on the source. This can be seen on Table V.

C. Adaptation efficiency

In order to assess the full potential of our domain adaptation strategy, we observe the influence of the size of the training set from the target, on the performances observed on the whole target testing set with our domain adaptative strategy. Moreover, to compare our results with a relevant baseline, we also train our detector in a supervised setting with only $N_{train} = 1,000$ random patches from the target. It corresponds to a situation where someone had the time to label only $N_{train} = 1,000$ patches from the target. The results are presented in Table IV.

	Target Acc.	Std
SrcOnly($N_{train} \sim 18,000$)	**65.7%**	1%
TgtOnly($N_{train} \sim 18,000$) (supervised)	**82.9%**	<1%
TgtOnly($N_{train} = 1,000$) (supervised)	72.7%	1%
Mix (supervised)	**82.1%**	<1%
Update($N_{train} = 10$) (unsupervised)	73.6%	3%
Update($N_{train} = 100$) (unsupervised)	72.4%	1%
Update($N_{train} = 1,000$) (unsupervised)	74.8%	1%
Update($N_{train} \sim 18,000$) (unsupervised)	76.9%	< 1%

TABLE IV
EFFICIENCY ON THE ADAPTATION FROM AN UNCOMPRESSED SOURCE TO A TARGET COMPRESSED TO QF5 W.R.T. NUMBER OF PATCHES AVAILABLE FROM THE TARGET.

Using our strategy with only 100 unlabelled target patches for the training, we observe performances equivalent to the ones obtained after having labelled 1000 target patches. This is an important observation since it might be time consuming to label these $1,000$ patches. We can also note that with only 1000 unlabelled patches, we are not far from the results we got using 18 times more unlabelled patches in the Update setting and we deduce that the gain of performance is limited to a certain point.

D. Performance on intermediate quality factors

When we adapt our forgery detector from an uncompressed source to a strongly compressed target, we expect to perform better on our target while maintaining good performance on our source. Now, intuitively, if this adaptation is correctly achieved, we could expect a better performance on a target compressed with a quality factor higher than the one we chose for the adaptation. To confirm it, we compute the target accuracy on several targets during our experiment in IV-B. The results are presented in Table V.

	SrcOnly	$\sigma = 8$	TgtOnly	Mix
QF5	**65.7%**	76.9% (+/- 0.1%)	**82.9%**	82.1%
QF10	74.9%	81.0% (+/- 1%)	82.7%	83.6%
QF20	81.8%	82.7% (+/- 1%)	81.8%	84.7%
QF50	84.2%	83.2% (+/- 1%)	81.4%	85.4%
QF100	84.8%	83.0% (+/- 1%)	81.2%	86.2%
No comp.	91.8%	88.7% (+/- 0.1%)	81.2%	87.1%
Max std	*3%*	*1%*	*1%*	*1%*

TABLE V
RESULTS ON INTERMEDIATE QFS FOR THE EXPERIMENT IN IV-B. *Max std* DENOTES THE MAXIMUM STANDARD DEVIATION OVER ALL RUNS (3-FOLD CV).

The case where no compression is applied to the target is a specific case where the target is equivalent to the source. It enables to observe the loss in performance in the source due to the adaptation. We can see that this loss is reasonable with a bandwidth σ correctly tuned.

As expected, the adaptation achieved with $\sigma = 8$ on a target compressed to QF5 has a positive and net impact on the efficiency of the detector on targets compressed to higher quality factors until a certain point. However, we loose in efficiency within the source domain highlighting the compromise in the generalization ability of the detector.

At last, we would like to point out an odd result. It seems that, in the mix setting, the performances observed on the source are worse than the ones observed training the detector only on the source, and, it also applies for the target. We reasonably think that this observation is due to a lack of complexity of our model or a lack of data from both source and target for the training. It is also important to have in mind that we simulated this setup taking randomly half of the patches available from the source and the target to get a database comparable in term of size with the one used in the other setups.

E. Choice of the source domain

An important question when we try to do domain adaptation is the choice of our source domain. If we want to transfer the knowledge acquired from a source to a target, we expect generally that the source is richer in information compared to the target. To illustrate that point, we consider here two domain adaptations with our forgery detector where the role of the source and the target are swapped. More precisely, we compare adaptation from QF100 to QF5 to adaptation from

QF5 to QF100. The results are presented in Tables VI and VII. For this experiment, $\sigma = 6$ revealed to be a better choice than $\sigma = 8$.

As shown by these tables, we are unable to achieve a relevant domain adaptation from QF5 to QF100. In reality, when the detector is trained on QF5 without adaptation, its performances on QF5 are very similar to the ones observed on QF100. This is visible via the TgtOnly set up presented in V. We guess that this is because compressed images are made of low-level features that can be completely retrieved from images compressed with QF100. Nevertheless, in that case, we have a net discrepancy between the best results achievable on QF100 from QF5 and the best results achievable on QF100 from directly QF100. This is expected since images compressed with QF100 contains much more information compared to the images compressed with QF5.

	SrcOnly	$\sigma = 6$	**TgtOnly**	**Mix**
Target Acc.	**81.2%**	81.0%	**88.4%**	86.4%
Std	<1%	<1%	<1%	<1%

TABLE VI
ADAPTATION FROM QF5 TO QF100
(TRAIN) $N_{source} \sim N_{target} \sim 18,000$ PATCHES.
(TEST) $N_{source} \sim N_{target} \sim 9,000$ PATCHES.

	SrcOnly	$\sigma = 6$	**TgtOnly**	**Mix**
Target Acc.	**66.8%**	75.2%	**82.9%**	81.7%
Std	3%	<1%	<1%	<1%

TABLE VII
ADAPTATION FROM QF100 TO QF5
(TRAIN) $N_{source} \sim N_{target} \sim 18,000$ PATCHES.
(TEST) $N_{source} \sim N_{target} \sim 9,000$ PATCHES.

Even if the adaptation from QF5 to QF100 is a failure, we observe a success in the reverse case, from QF100 to QF5. This experiment illustrates the importance to choose properly the source domain when we can do it.

V. CONCLUSION AND PERSPECTIVES

In this paper, we propose to use domain adaptation in order to overcome the mismatch problem, which arises in digital images forensics in a blind situation where it is not possible to have access to any target label. We show that domain adaptation for different JPEG quality is possible using the MMD associated to a Gaussian kernel defined with a well chosen σ. One should prefer training its detector on a source dataset with none or low compression and adapt it to the more compressed target. The proposed *update* method neither require labelling of the target dataset nor needs a huge amount of images which may be not available in practical cases. Broadly speaking, domain adaptation should be investigated for other splicing post process.

REFERENCES

[1] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *International Conference on Machine Learning*. PMLR, 2015, pp. 97–105.

[2] Y. Rao and J. Ni, "A deep learning approach to detection of splicing and copy-move forgeries in images," in *International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2016, pp. 1–6.

[3] Y. Wu, W. AbdAlmageed, and P. Natarajan, "Mantra-net: Manipulation tracing network for detection and localization of image forgeries with anomalous features," in *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019, pp. 9543–9552.

[4] B. Bayar and M. C. Stamm, "A deep learning approach to universal image manipulation detection using a new convolutional layer," in *Proceedings of the 4th ACM workshop on information hiding and multimedia security (IH&MMSec)*, 2016, pp. 5–10.

[5] G. Mahfoudi, B. Tajini, F. Retraint, F. Morain-Nicolier, J. L. Dugelay, and M. Pic, "DEFACTO: image and face manipulation dataset," in *27th European Signal Processing Conference (EUSIPCO 2019)*, 2019.

[6] A. D. Ker, P. Bas, R. Böhme, R. Cogranne, S. Craver, T. Filler, J. Fridrich, and T. Pevnỳ, "Moving steganography and steganalysis from the laboratory into the real world," in *Proceedings of the first ACM workshop on Information hiding and multimedia security (IH&MMSec)*, 2013, pp. 45–58.

[7] Q. Giboulot, R. Cogranne, D. Borghys, and P. Bas, "Effects and solutions of cover-source mismatch in image steganalysis," *Signal Processing: Image Communication*, vol. 86, p. 115888, 2020.

[8] H. Daumé III, "Frustratingly easy domain adaptation," *arXiv preprint arXiv:0907.1815*, 2009.

[9] H. Daumé III and D. Marcu, "Domain adaptation for statistical classifiers," *Journal of artificial Intelligence research*, vol. 26, pp. 101–126, 2006.

[10] D. Cozzolino, J. Thies, A. Rössler, C. Riess, M. Nießner, and L. Verdoliva, "Forensictransfer: Weakly-supervised domain adaptation for forgery detection," 2018.

[11] S. Saxena and J. Verbeek, "Heterogeneous face recognition with cnns," in *European conference on computer vision*. Springer, 2016, pp. 483–491.

[12] W. Quan, K. Wang, D.-M. Yan, X. Zhang, and D. Pellerin, "Learn with diversity and from harder samples: Improving the generalization of CNN-based detection of computer-generated images," *Forensic Science International: Digital Investigation*, vol. 35, p. 301023, 2020.

[13] Y. Yousfi and J. Fridrich, "JPEG steganalysis detectors scalable with respect to compression quality," *Electronic Imaging*, vol. 2020, no. 4, pp. 75–1, 2020.

[14] I. Redko, E. Morvant, A. Habrard, M. Sebban, and Y. Bennani, *Advances in domain adaptation theory*. Elsevier, 2019.

[15] H. Li, W. Li, H. Cao, S. Wang, F. Huang, and A. C. Kot, "Unsupervised domain adaptation for face anti-spoofing," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 7, pp. 1794–1809, 2018.

[16] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 723–773, 2012.

[17] L. Darmet, K. Wang, and F. Cayre, "Graft: Unsupervised adaptation to resizing for detection of image manipulation," *IEEE Access*, vol. 8, pp. 55 619–55 632, 2020.

[18] P. L. C. Rodrigues, C. Jutten, and M. Congedo, "Riemannian procrustes analysis: transfer learning for brain–computer interfaces," *IEEE Transactions on Biomedical Engineering*, vol. 66, no. 8, pp. 2390–2401, 2018.

[19] Y. Lin, R. Wang, L. Dong, D. Yan, and J. Wang, "Tackling the cover source mismatch problem in audio steganalysis with unsupervised domain adaptation," *IEEE Signal Processing Letters*, 2020.

[20] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML'15. JMLR.org, 2015, p. 1180–1189.

[21] S. Mandelli, N. Bonettini, P. Bestagini, and S. Tubaro, "Training CNNs in presence of JPEG compression: Multimedia forensics vs computer vision," in *2020 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2020, pp. 1–6.

[22] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.