Learning Unified Deep-Features for Multiple Forensic Tasks

Owen Mayer
Drexel University
Philadelphia, PA, USA
om82@drexel.edu

Belhassen Bayar
Drexel University
Philadelphia, PA, USA
bb632@drexel.edu

Matthew C. Stamm
Drexel University
Philadelphia, PA, USA
MStamm@coe.drexel.edu

ABSTRACT

Recently, deep learning researchers have developed a technique known as *deep features* in which feature extractors for a task are learned by a CNN. These features are then provided to another classifier, or even used to perform a different classification task. Research in deep learning suggests that in some cases, deep features generalize to seemingly unrelated tasks. In this paper, we develop techniques for learning deep features that can be used across multiple forensic tasks, namely image manipulation detection and camera model identification. To do this, we develop two approaches for building deep forensic features: a transfer learning approach and a multitask learning approach. We experimentally evaluate the performance of both approaches in several scenarios and find that: 1) features learned for camera model identification generalize well to manipulation detection tasks but manipulation detection features do not generalize well to camera model identification, suggesting a task asymmetry, 2) deeper features are more task specific while shallower features generalize well across tasks, suggesting a feature hierarchy, and 3) a single, unified feature extractor can be learned that is highly discriminative for multiple forensic tasks. Furthermore, we find that when there is limited training data, a unified feature extractor can significantly outperform a targeted CNN.

KEYWORDS

Multimedia forensics, deep learning, deep features, transfer learning, multitask learning

1 INTRODUCTION

In recent years, there has been an explosion in deep learning research targeted at multimedia forensic tasks. For example, work in [11] shows that a convolutional neural network (CNN) can be built to detect whether an image patch has undergone median filtering. Other works have shown that resampling operations can be detected using deep learning methods [4, 10], as well as several approaches to classify multiple post-processing manipulations [1, 5, 12]. Research has also shown that CNNs can be used to identify the source camera model of an image patch with high accuracy [2, 3, 7, 8, 20]. To date, many multimedia forensic deep learning methods have targeted several types of manipulation detection tasks and camera model identification tasks.

IH&MMSec '18, June 20–22, 2018, Innsbruck, Austria

ACM ISBN 978-1-4503-5625-1/18/06…$15.00
https://doi.org/10.1145/3206004.3206022

More recently, research has found that techniques utilizing *deep features* are also very effective for multimedia forensics tasks [4, 6, 7, 9, 15]. Deep features are the neuron responses, at a particular layer of a CNN, induced by the feeding forward an image through the network [13, 19]. In these approaches a CNN is trained for one task, and then deep features extracted from that network are utilized for a different task. For example, work by Bondi et al. in [7] showed that deep features extracted from a closed set camera model identification CNN can be used to train an SVM that classifies a different set of camera models. Mayer and Stamm [15] showed that pairs camera model deep features can be mapped to a similarity score to identify whether two images were captured by the same camera model, even if those camera models are unknown. Additionally, Bayar and Stamm [6] proposed a technique for open set camera model identification using deep features. The findings of these works suggest that deep feature representations learned for camera model identification may generalize to many different camera models, not just those in the original training set.

In research outside of multimedia forensics, deep features have been shown to generalize to seemingly unrelated tasks. For example, deep features extracted from a CNN pre-trained for scene detection can be used to train a object detection classifier and vice versa [21]. Additionally, work in the remote sensing community has shown that object recognition deep features can be trained for land-usage classification tasks [17]. These findings suggest that perhaps features learned in one specific multimedia forensics task may be applicable to other forensic tasks.

While research has shown that features learned for camera model identification generalize to other camera models [6, 7, 15], there has been no research showing that camera model identification features generalize to manipulation detection tasks, or vice versa. Additionally, very little is understood about which image features are being captured by deep learning methods. Because of this, we are led to ask several questions:

- Are features learned for one forensic task useful for another forensic task? Is it possible that forensic tasks, which are often thought of as very different, are actually very similar?
- Does an abstract hierarchical structure exist for feature generalization? That is, is it possible that low-level features generalize well across tasks but a more task-specific representation does not?
- Does there exist a single set of universal features that work for all multimedia forensic tasks?

In this paper we address these questions and show that 1) features learned for camera model identification generalize well to manipulation detection tasks, but that features learned for manipulation detection tasks do not generalize well to camera model identification tasks, suggesting a *task asymmetry*, 2) that deeper features are more task-specific, while shallower features generalize

well across tasks, suggesting a *feature hierarchy*, and 3) that a single, unified feature extractor can be learned that is highly discriminative for multiple forensic tasks. Furthermore, we find that when there is limited training data, a unified feature extractor can significantly outperform a targeted CNN.

To do this, we adopt two strategies for learning deep feature extractors, and then experimentally evaluate the performance of each to demonstrate the properties that are enumerated above. In the first approach, we propse a transfer learning method where a CNN is initially learned for one forensic task. Then, the lower layers of the CNN are frozen, acting as a fixed feature extractor, while learning new upper layers that target a different task. We use this approach to experimentally demonstrate the transference of features between tasks. Furthermore, to evaluate the feature hierarchy, we experimentally vary the depth at which the CNN layers are frozen during retraining on the second task.

In the second approach, we propose a multitask learning approach where we train two CNNs simultaneously while constraining the lower layers of both networks to learn the same weights and biases. This approach effectively creates a single, unified feature extractor that is highly discriminative for multiple forensic tasks. Additionally, we find this approach to significantly outperform a targeted CNN when there is limited training data.

2 DEEP FEATURES OVERVIEW

In deep feature based approaches, classification of an image patch is broken down into two steps. In the first step, an image patch $X \in \mathbb{X}$, where $\mathbb{X}$ is the space of image patches, is mapped to an N-dimensional, real-valued feature space by a deep feature extractor function $\mathbf{f}(\cdot)$:

$$\mathbf{f} : \mathbb{X} \rightarrow \mathbb{R}^N. \tag{1}$$

The feature vector $\mathbf{f}(X)$ encodes high level forensic information about the image patch X. Next, a task classifier $g(\cdot)$ maps the deep feature vector into a classification decision

$$g : \mathbb{R}^N \rightarrow \mathbb{T}, \tag{2}$$

where $\mathbb{T}$ is the set of target classes (e.g. a set of camera models or a set of manipulations). Thus, the total system

$$y = g(\mathbf{f}(X)) \tag{3}$$

maps an input image $X \in \mathbb{X}$ to a classification decision $y \in \mathbb{T}$.

A diagram of the deep feature approach is shown in Fig. 1. An image patch X is input to a deep feature extractor $\mathbf{f}(\cdot)$, and the output deep features are input to task classifiers, $g_A(\cdot)$, $g_B(\cdot)$, $g_C(\cdot)$, which classify the image for the respective tasks A, B, and C.

In this work, we use a pre-trained CNN as the deep feature extractor $\mathbf{f}(\cdot)$. The features $\mathbf{f}(X)$ are evaluated by recording the neuron responses, at a specified layer, induced by the feed-forward of an input image patch X. Often, the last fully connected layer of the CNN is used for deep feature extraction [13, 19]. The power of using a deep feature approach is that

$$dimensionality\left(\mathbf{f}(X)\right) << dimensionality\left(X\right),$$

which enables a task classifier $g(\cdot)$ to be trained with relatively few training samples.

However, a deep feature approach also requires that the features output by the feature extractor $\mathbf{f}(\cdot)$ are general enough to discriminate between the classes targeted by $g(\cdot)$. Ideally, the features extracted from feature extractor encode general, low-dimensional forensic information about the image patch, which allows for training task specific classifiers on many different forensic tasks.

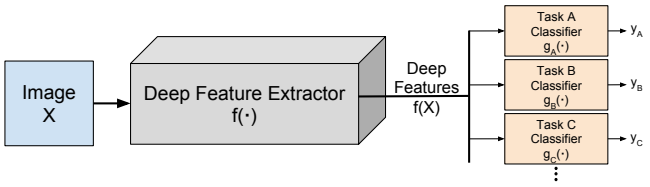


Figure 1: Diagram showing deep feature extraction and classification. A feature extractor maps an input image patch X into a deep feature space. Task classifiers then map these features into a task-specific classification decision.

3 CNN ARCHITECTURE

In this work, we use pre-trained CNNs to perform deep feature extraction. For each CNN, we use a network architecture proposed by Bayar and Stamm that has proven effective at both manipulation detection and source camera model identification [3, 5]. Briefly, the network consists of 5 convolutional layers and 3 fully connected layers. The first convolutional layer, labeled 'constr' is a set of 3 5x5 constrained filters where the central weight is constrained to be -1 and the rest of the weights sum to one, such that

$$\begin{cases} \mathbf{w}(0,0) = -1 \\ \sum_{(l,m)\neq(0,0)} \mathbf{w}(l,m) = 1 \end{cases} \tag{4}$$

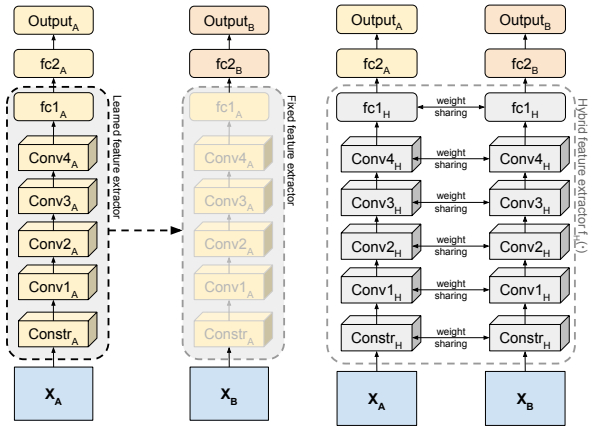
where l, m are the spatial indices of each constrained filter. These constraints are designed to suppress image content while learning salient forensic features. The remaining 4 convolutional layers, labeled 'conv1'–'conv4,' each have hyperbolic tangent activation, mini-batch normalization and pooling. The network has 3 fully connected layers. The first two are labeled 'fc1' and 'fc2', each with 200 neurons and hyperbolic tangent activation. Finally, the network has an output fully connected layer with softmax function. Further details of the baseline architecture, including filter dimensions and parameter choice motivation, can be found in [5].

4 LEARNING DEEP FEATURE EXTRACTORS

The three goals of this work are to 1) propose and evaluate techniques to develop a unified set of deep features that are discriminative for multiple forensics tasks, as well as to 2) evaluate the transference of deep features between tasks and finally to 3) identify a potential abstract hierarchy of deep features. In this section, we propose two approaches for learning deep feature extractors, i.e. the mapping $\mathbf{f}(\cdot)$ described in Eq. (1).

4.1 Transfer Learning

In the first approach, we use transfer learning to learn the deep feature extractor. In transfer learning, portions of a convolutional neural network (CNN) pre-trained for one task are used to extract features for another task [16]. That is, the knowledge (feature representations) learned for one task are "transferred" to another task. In the experimental evaluation in Sec. 5.2, we use this approach to evaluate how well the features learned from one task generalize to another task. Additionally, by varying the depth of the feature



(a) Transfer Learning (b) Multitask Learning

Figure 2: Diagrams of proposed approaches (a) transfer learning and (b) multitask learning, both using an example share depth of *fc1*. In the transfer learning approach, layers 'constr' through 'fc1' are learned for an initial task A, then 'fc2' and 'output' are retrained for a new task B. In the multitask approach, two networks are trained simultaneously on different datasets, but with layers 'constr' through 'fc1' constrained to learn the same parameters. This multitask approach creates a single, hybrid feature extractor that outputs deep features discriminative of both tasks A and B.

extractor, we use this approach to also evaluate the hierarchical nature of feature transference.

The transfer learning process is accomplished in two training phases. In the first training phase, a *source* CNN is trained for a source task A, using a baseline architecture described in Sec. 3. Training is performed with a set of training image patches X_A that are representative of classes in $\mathbb{T}_A$. The result of this training phase is a trained network that has layers $constr_A$ through $output_A$, which is depicted by the left hand side of the diagram in Fig. 2a.

In the second training phase, we first fix the source CNN learned for task A, preventing its parameters from being updated. Then, we discard the upper layers of the source CNN, above a layer called the *share depth*. Finally, we replace the discarded layers with new layers that are then learned for target task B. A diagram of this approach with a share depth of $fc1$ is shown on the right of Fig. 2a. The fixed lower layers of the source CNN through 'fc1_A' are input to new layers 'fc2_B' and 'output_B' learned for the target task B.

The term *share depth* is used to signify the layers that are shared between the two tasks. The layers below and including the share depth act as the deep feature extractor $\mathbf{f}_A(\cdot)$. The layers above the share depth act as the task specific classifier.

4.2 Multitask Learning

In our second approach, we propose a method to learn a single feature extractor that outputs deep features highly discriminative for multiple forensic tasks. To do this, we train two (or more) CNNs simultaneously on two (or more) different tasks, but constrain the lower layers of each network to learn the same parameters. This is a form of multitask learning [18]. The shared layers of these networks form a single, unified feature extractor that outputs deep features discriminative of two (or more) forensic tasks, which we call a *hybrid* feature extractor. In the experimental evaluation in Sec. 5.3, we use this approach to evaluate how well hybrid features are able to classify multiple tasks, and compare with feature extractors learned in the transfer learning approach.

A diagram of the multitask learning process is shown in Fig. 2b, which uses a share depth of *fc1*. Unlike the transfer learning process, the multitask process is accomplished in a single learning phase. One network leg has input database for task A, and the network leg has input database for task B. The bottom layers through the share depth are called *hybrid layers*. In the example shown in Fig. 2b, the hybrid layers are labeled 'constr_H' through 'fc1_H' creating a feature extractor $f_H(\cdot)$. The task A specific layers are labeled 'fc2_A' and 'output_A', and the task B specific layers are labeled 'fc2_B' and 'output_B', creating task specific classifiers $g_A(\cdot)$ and $g_B(\cdot)$.

During training, at each iteration the weights and biases of the hybrid layers are updated according to

$$w'_k = w_k + \Delta w_k, \tag{5}$$

where w_k is the original weight indexed by k, Δw_k is the weight update step, and w'_k is the updated weight. The update step is calculated by

$$\Delta w_k = \sum_{t \in T} \lambda_t \Delta w_{k,t}, \tag{6}$$

where T is the set of tasks that are being targeted, t is a specific task in that set, λ_t is a specified weight (i.e. preference) given to task t, and $\Delta w_{k,t}$ is the portion of the update step attributed to task t. In our approach, we use $\lambda_t = 1\ \forall t$, that is each task has equal weight. The task portion of the update step is calculated according to

$$\Delta w_{k,t} = -\eta \frac{\partial L_t}{\partial w_k}, \tag{7}$$

where η is the learning rate and L_t is the loss for task t. In summary, we ensure the two network's learn the same parameters by summing together the gradient for each task-specific loss and applying the same update rule to each network.

In addition, the architecture that we use employs mini-batch normalization. After each iteration, we average the normalization parameters across network legs to ensure that each network uses the same normalization.

5 EXPERIMENTAL EVALUATION

In this section, we experimentally evaluate the following: 1) how well features learned for one task transfer to another task, specifically the tasks of camera model identification and manipulation detection, 2) whether shallower features are more generic and deeper features are more task specific, and 3) whether a set of unified features can be learned that are discriminative for multiple tasks. The first two evaluations were performed using the transfer learning approach, and the third evaluation was performed using the multitask learning approach.

To do this, we created two different databases, one for each task under investigation. The first database we created was for the manipulation detection task, following the steps used in [3]. We collected 400,000 training patches and 50,000 testing patches of size 256×256 using 8334 images chosen at random from the publicly available Dresden Image Database [14]. The training and

Manipulation Detection, $\mathbb{T}_A$ =		
Unaltered	Median Filter (5x5)	Gauss. Blur (σ = 1.1)
AWGN (σ = 2)	Bilinear Interp. (x1.5)	JPEG (Q=70)
Camera Model Identification, $\mathbb{T}_B$ =		
Apple iPhone 4s*	Nikon Coolpix S710†	Rollei RCP-S7325XS†
Apple iPhone 6*	Nikon D200†	Samsung Gal. Note4*
Canon Ixus70†	Olympus mju-1050SW†	Samsung Gal. S4*
Casio EX-Z150†	Panasonic DMC-FZ50†	Samsung LZ74 wide†
FujiFilm FinePixJ50†	Pentax OptioA40†	Samsung NV15†
Kodak M1063†	Praktica DCZ5.9†	Sony DSC-T77†
LG Nexus 5x*	Ricoh GX100†	
Limited Training Data Camera Model Identification, $\mathbb{T}_C$ =		
HTC One M7*	Nikon D70†	Sony DSC-W170†
Mot. Droid Maxx*	Samsung Galaxy S2*	

Table 1: Classes for the three investigated tasks.
Camera models from our database*, and from the Dresden Image Database†

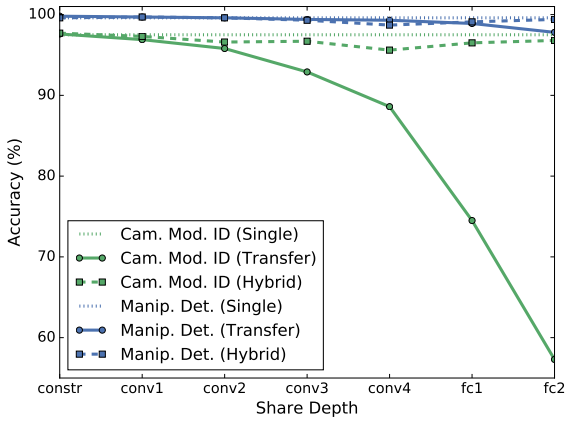


Figure 3: Accuracy, by share depth, for the single task, transfer learning, and multitask learning approaches.

testing patches were created from two separate sets of images using the nine central 256 × 256 blocks of each image. Only the green color channel was used. Next, each block was edited using the six processing operations in the set $\mathbb{T}_A$ in Table 1.

We then created a second database for the source camera model identification task. This database is composed of image patches of size 256 × 256 from the set $\mathbb{T}_B$ of 20 different camera models in Table 1. The camera models were chosen such that they had at least 2 devices. For each camera model, 20,000 non-overlapping training patches were randomly selected from among all but one device. For testing, 1500 non-overlapping patches were randomly chosen from the remaining device. The camera models were also selected to create a diversity of manufacturers and camera types (e.g. point-and-shoot, DSLR, cell phone). These images were taken from the publicly available Dresden Image Database [14], and from our own database of cameras. Camera models from our database had at least 300 images per device, with images taken in diverse scene environments. In total, 400,000 training patches and 30,000 testing patches were used.

5.1 Single task baseline

To provide a measure for comparison, we trained the baseline network for each task individually, i.e. 'Single Task' training. To do this, we trained the baseline CNN network described in Sec. 3 for each task. We performed training using stochastic gradient decent with a base learning rate of 0.001, momentum of 0.9, and batch size of 40 patches. The learning rate was halved every 3 epochs, and the network was trained for 30 epochs total.[1] Single task accuracy of 97.5% was achieved for camera model identification, and 99.6% for manipulation detection.

5.2 Feature transfer and hierarchy

In this experiment, we used the transfer learning approach outlined in Sec. 4.1 to evaluate how well features learned from one forensic task transfer to another task forensic task, as well as to evaluate a feature hierarchy.

[1]Source code for this work can be found at misl.ece.drexel.edu/downloads or gitlab.com/MISLgit/unified-features-ihmmsec2018. All experiments were conducted using caffe with an Nvidia GTX 1080 or 1080ti GPU.

To do this, we used a pre-trained single-task network, learned above in Sec. 5.1, as a deep feature extractor. Then we retrained the upper layers, above a share depth, to target the other task. We varied the share depth between the shallowest possible of 'constr' through the deepest possible of 'fc2.' This was done for both tasks. Training was performed using stochastic gradient descent with a base learning rate of 0.005, momentum of 0.9, and a batch size of 100 patches. The learning rate was halved every 3 epochs, and the network was trained for 30 epochs total, with no early stopping.

The accuracy achieved for each transfer learning scenario, by share depth, is shown in Fig. 3. Transfer of camera model features to target the manipulation detection task is shown in solid green. Transfer of manipulation features to target the camera model identification task is shown in solid blue.

When we transfered manipulation features to the camera model identification task, an accuracy of 97.5% was achieved when using the shallowest share depth of 'constr'. Accuracy monotonically decreased as share depth was increased, and achieved an accuracy of 57.8% at the deepest share depth of 'fc2.' When we transfered camera model features to the manipulation detection task, an accuracy of 99.8% was achieved when using the shallowest share depth of 'constr'. Accuracy monotonically decreased as share depth increased, and achieved an accuracy of 97.6% for at 'fc2' share depth.

An important observation from this experiment is that when we targeted the camera model identification task using manipulation features, there was a significant drop in accuracy of 39.7 percentage points relative to the single-task baseline, at a share depth of 'fc2'. However, there was only a 2.0 percentage point drop in accuracy when we targeted the manipulation detection task using camera model features up to the same share depth.

This result suggests that a *task asymmetry* exists in the generality of forensic deep features. That is, the camera model features transfer much better to the manipulation task than manipulation features transfer to camera model identification. One possible explanation for this phenomenon is that camera model features may be much more complex than manipulation features, and thus need to discriminate a greater expanse of the forensic feature space, which may include manipulation features. Another possible explanation is that manipulation features are a subset of camera model features, but that the reverse is not true.

Target Task:	Manipulation		Camera Model	
Feature Extractor	**CNN**	**ERT**	**CNN**	**ERT**
Manipulation (*A*)	99.6%	**99.8%**	57.8%	72.5%
Camera Model (*B*)	97.6%	98.9%	97.5%	**97.7%**
Hybrid (*A* + *B*, fc2)	99.4%	**99.8%**	96.8%	**97.6%**

Table 2: Accuracy comparison of extremely randomized trees (ERT) classifier on fc2 deep features versus the single task, transfer, and hybrid networks

Feature Extractor	**CNN**	**ERT**
Manipulation Detection (*A*)		75.0%
20 Camera Model (*B*)		88.9%
5 Camera Model (*C*)	86.6%	85.6%
Hybrid (*A* + *B*, fc2)		91.5%
Hybrid (*A* + *B* + *C*, fc2)	92.1%	**92.8%**

Table 3: Classification accuracy for the 5 camera model identification task using different feature extractors.

Another important observation from this experiment is that, for both tasks, the classification accuracy monotonically decreased as the share depth was increased. This result suggests a *feature hierarchy* of forensic features. That is, low-level features learned by the shallower layers of a network are general across tasks, i.e. higher level features for different forensic tasks can be learned from these low-level representations. By contrast, high-level features learned in the deeper layers appear to be more task specific. This result has significant implications for forensic investigators who use transfer learning methods, and shows that the choice of share depth is a critical one. We note that also as the share depth decreases, the parameters that must be retrained for the target task increases. Thus if the training dataset is small, it may not be practical to use a shallow share depth.

Another interesting observation is that classification accuracy actually improves to 99.8% for the manipulation detection task when using the 'constr' features transferred from camera model network, versus the classification accuracy of 99.6% achieved by the single-task CNN. Literature in multitask learning suggests that training on different tasks may prevent overfitting by enforcing generality of features [18], which perhaps happened in this case.

5.3 Unified deep features

In this experiment, we used the multitask learning approach outlined in Sec. 4.2 to evaluate how well unified features are able to discriminate multiple tasks. To do this, we trained hybrid networks using the multitask learning approach described in Sec. 4.2. Additionally, we varied the share depth from 'constr' through 'fc2'. For each hybrid network, we simultaneously trained on the manipulation detection and camera model identification datasets. Training was performed using stochastic gradient descent with a base learning rate of 0.001, momentum of 0.9, and batch size of 40 patches per task. The learning rate was halved every 3 epochs, and the network was trained for 30 epochs total with no early stopping.

The accuracy achieved for each task using this multitask learning approach, by share depth, is shown by the dashed lines with squares in Fig. 3. At a share depth of 'fc2,' the most difficult scenario for the transfer learning approach, an accuracy of 96.8% was achieved for the camera model identification task. This improved accuracy by 39.5 percentage points over the transfer learning method. For the manipulation detection, and accuracy of 99.4% was achieved using the multitask approach, a 1.6 percentage point improvement over the transfer learning method.

Notably, at all share depths, the multitask approach improved accuracy for the camera model identification task over the transfer learning method. Additionally, for the manipulation detection task, the multitask approach improved classification accuracy over the transfer learning approach at deep share depths of 'fc1' and 'fc2.'

While the multitask approach did not improve classification accuracy over the single-task baseline, it did improve classification accuracy over the transfer learning approach especially at deeper share depths. The results of this experiment demonstrate that unified features are much more effective for discriminating multiple forensic tasks than using a transfer learning approach.

5.3.1 Hybrid features with extremely randomized trees. In this experiment, we used extremely randomized trees (ERT) classifiers to improve classification accuracy of the unified features. Work in [4] found that the use of extremely randomized trees on deep features extracted from the last fully connected layer improved classification accuracy of resizing detection. We also compare to the transfer learning and single task approaches.

For each task we extracted 'fc2' deep features from 1) the baseline manipulation detection CNN (*A*), 2) the baseline camera model identification CNN (*B*), and 3) the hybrid network with share depth of 'fc2' (*A* + *B*). We then trained an ERT classifier on each set of features, using 800 estimators and a minimum of 3 samples required to split an internal node.

Results from this experiment are shown in Table 2, and are compared to the results from the previous, CNN-based experiments. In each case, the ERT classifier slightly improves the CNN classifier. Notably, in the manipulation detection case the hybrid features performed equally as well as the single task feature extractor case, both achieving 99.8% accuracy. For camera model identification, the hybrid features achieve an accuracy of 97.6%, which is very nearly the single task accuracy of 97.7%.

The results of this experiment shows that the hybrid feature extractor is able to learn a single, unified set of features that are highly discriminative of multiple forensic tasks.

5.4 Classification with limited training data

In this experiment we tested the efficacy of using unified features on a new task with limited training data. This experiment was conducted to simulate a scenario where a forensic investigator may employ a deep feature approach, i.e. a scenario where there is not enough training data to robustly train a full CNN from scratch.

To do this, we first created a third database for camera model identification of 5 camera models. This set of camera models is labeled by $\mathbb{T}_C$ in Table 1, and are disjoint from the 20 camera models used in the above experiments. For training and testing, we collected image patches using the procedure outlined for the 20 camera model identification task. However, for each camera model, only 10,000 patches were collected per class for training and testing,

resulting in 50,000 total training patches (1/8 of the 20 camera model training data), and 50,000 testing patches. To establish a baseline classification accuracy, we trained a single task CNN to target the 5 camera models using the same setup described in Sec. 5.1. The single task classifier achieved relatively poor accuracy of 86.6%.

Additionally, we trained a new hybrid network on the three training databases simultaneously, using an 'fc2' share depth. Training was performed using stochastic gradient descent with a base learning rate of 0.001, momentum of 0.9, and batch size of 40 patches per for the manipulation detection and 20 camera model tasks, and a batch size of 5 for the 5 camera model task. This was done to ensure that number of iterations per epoch was the same across all tasks. The learning rate was halved every 3 epochs, and the network was trained for 30 epochs total.

We used the pre-trained CNNs for manipulation detection (A), 20 camera model identification (B), 5 camera model identification (C), 2-task hybrid network with fc2 share depth ($A + B$), and 3-task hybrid network with fc2 share depth ($A + B + C$) to extract the 'fc2' features from the 5 camera model training data. Then, we trained an ERT classifier on each set of deep features extracted from these networks. To train each ERT, we used 800 estimators and required a minimum of 3 samples to split an internal node.

The classification accuracy achieved by each ERT is shown in Table 3. The ERT classifier achieved 85.6% when using 'fc2' features extracted from the 5 camera model CNN, which was trained with limited training data. Classification accuracy improved to 88.9% when using an ERT trained on features from the 20 camera model identification CNN. Accuracy was further improved to 91.2% when using deep features from the 2-task hybrid network. These results show that using deep features from well trained networks generalize well to new tasks, and that enforcing task generalization also improves the ability of the deep features

Furthermore, the highest classification accuracy of 92.8% was achieved when using deep features from the 3-task hybrid network. This result shows that using hybrid, unified features that also incorporate training samples from the target class improves classification accuracy over using a targeted CNN trained with limited training samples. These results show that it is important for a forensic investigator to enforce class and task generality when targeting new tasks that have limited training data.

6 CONCLUSION

In this paper, we adopted two strategies for learning deep feature extractors; a transfer learning approach, where features from one task are transferred to another task, and a multitask learning approach, where a single feature extractor is jointly optimized on multiple tasks. We experimentally evaluated their performance in several scenarios, in which we found that: 1) features learned for camera model identification generalize well to manipulation detection tasks, but features learned for manipulation detection tasks do not generalize well to camera model identification tasks, suggesting a task asymmetry, 2) deeper features are more task-specific, whereas shallower features generalize well across tasks, suggesting a feature hierarchy and 3) a single, unified feature extractor can be learned that is highly discriminative of multiple forensic tasks. Furthermore, we found that unified feature extractors outperform a targeted CNN when there is limited training data. These results highlight several critical considerations that a forensic investigator must make when using deep feature based approaches.

7 ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1553610. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Belhassen Bayar and Matthew C Stamm. 2016. A deep learning approach to universal image manipulation detection using a new convolutional layer. In *Proc. of the 4th ACM Workshop on Info. Hiding and Multimedia Security*. ACM, 5–10.
[2] Belhassen Bayar and Matthew C Stamm. 2017. Augmented convolutional feature maps for robust cnn-based camera model identification. In *Image Processing (ICIP), 2017 IEEE International Conference on*. IEEE, 1–4.
[3] Belhassen Bayar and Matthew C Stamm. 2017. Design principles of convolutional neural networks for multimedia forensics. *Electronic Imaging* 7 (2017), 77–86.
[4] Belhassen Bayar and Matthew C Stamm. 2017. On the robustness of constrained convolutional neural networks to jpeg post-compression for image resampling detection. In *ICASSP, 2017 IEEE*. IEEE, 2152–2156.
[5] Belhassen Bayar and Matthew C Stamm. 2018. Constrained Convolutional Neural Networks: A New approach Towards General Purpose Image Manipulation Detection. *IEEE Transactions on Information Forensics and Security* (2018).
[6] Belhassen Bayar and Matthew C Stamm. 2018. Towards open set camera model identification using a deep learning framework. In *Acoustics, Speech and Signal Processing (ICASSP), 2018 IEEE International Conference on*. IEEE, 1–4.
[7] Luca Bondi, Luca Baroffio, David Güera, Paolo Bestagini, Edward J Delp, and Stefano Tubaro. 2017. First Steps Toward Camera Model Identification With Convolutional Neural Networks. *IEEE Signal Processing Letters* (2017), 259–263.
[8] Luca Bondi, David Güera, Luca Baroffio, Paolo Bestagini, Edward J Delp, and Stefano Tubaro. 2017. A preliminary study on convolutional neural networks for camera model identification. *Electronic Imaging* 2017, 7 (2017), 67–76.
[9] Luca Bondi, Silvia Lameri, David Güera, Paolo Bestagini, Edward J Delp, and Stefano Tubaro. 2017. Tampering Detection and Localization through Clustering of Camera-Based CNN Features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 1855–1864.
[10] Jason Bunk, Jawadul H Bappy, Tajuddin Manhar Mohammed, Lakshmanan Nataraj, Arjuna Flenner, BS Manjunath, Shivkumar Chandrasekaran, Amit K Roy-Chowdhury, and Lawrence Peterson. 2017. Detection and Localization of Image Forgeries using Resampling Features and Deep Learning. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 1881–1889.
[11] Jiansheng Chen, Xiangui Kang, Ye Liu, and Z Jane Wang. 2015. Median filtering forensics based on convolutional neural networks. *IEEE Signal Processing Letters* 22, 11 (2015), 1849–1853.
[12] Davide Cozzolino, Giovanni Poggi, and Luisa Verdoliva. 2017. Recasting residual-based local descriptors as convolutional neural networks: an application to image forgery detection. In *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security*. ACM, 159–164.
[13] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. 2014. Decaf: A deep convolutional activation feature for generic visual recognition. In *Int. Conference on Machine Learning*. 647–655.
[14] Thomas Gloe and Rainer Böhme. 2010. The Dresden image database for benchmarking digital image forensics. *Jour. of Digital Forensic Practice* (2010), 150–159.
[15] Owen Mayer and Matthew C Stamm. 2018. Learned forensic source similarity for unknown camera models. In *ICASSP, 2018 IEEE*. IEEE, 1–4.
[16] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. 2014. Learning and transferring mid-level image representations using convolutional neural networks. In *Computer Vision and Pattern Recognition (CVPR)*. IEEE, 1717–1724.
[17] Otávio AB Penatti, Keiller Nogueira, and Jefersson A dos Santos. 2015. Do deep features generalize from everyday objects to remote sensing and aerial scenes domains?. In *Proceedings of the IEEE CVPR Workshops*. 44–51.
[18] Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098* (2017).
[19] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. 2014. CNN features off-the-shelf: an astounding baseline for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 806–813.
[20] Amel Tuama, Frédéric Comby, and Marc Chaumont. 2016. Camera model identification with the use of deep convolutional neural networks. In *Information Forensics and Security (WIFS), 2016 IEEE International Workshop on*. IEEE, 1–6.
[21] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. 2014. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*. 487–495.