Detecting Digital Photo Tampering with Deep Learning

Mitchell B. Singleton and Qingzhong Liu(✉)

Department of Computer Science, Sam Houston State University, Huntsville, TX 77341, USA
{mitchellsingleton, liu}@shsu.edu

Abstract. The tools to tamper with digital photographs have become easier to use by the lay person and techniques have evolved that make detection of tampering more difficult, there is a clear need to not only discover novel ways to distinguish the difference between real and fake, but also to make the detection quicker and easier. Using content-aware resizing, which is also known as image retargeting, seam carving, content-aware rescaling, liquid rescaling, or liquid resizing, allows for changing the resolution of an image while keeping the content unchanged in the image. In this paper, the retraining of Inception ResNet v2, one of the best object detection deep learning classifiers, is undertaken to look at classifications of untampered versus tampered photos via seam removal.

Keywords: Photography · Detection of tampering · Detection of modification Detection of falsification · Deep learning

1 Introduction

The ease of manipulating an image hasn't caused people to rely less on images as a source of truth. The reasons for the manipulation span the spectrum from what can seem benign (removing pimples, whitening teeth, or removing wrinkles in a school photo) all the way to intentionally malicious (falsified images passed as real, fear mongering, or deception).

Image manipulation tools have become widely available that allow anyone to perform robust image manipulation to the point where it is difficult to determine if an image has been falsified. Coupling that with an issue of the public becoming desensitized to the importance of image manipulation. The task for an average person to determine if an image has been tampered becomes herculean.

Previous investigations into manipulation of images have taken place looking at artifacts of specific file formats that occur after editing is done on an image. A very common standard to store digital images was created by the Joint Photographic Experts Group and the format became the known by the acronym JPEG and due to the popularity is an important file format to focus on for detection of tampering. To manipulate an image stored using the JPEG format there are a few basics operations; image scaling, rotation, copy and pasting, and double compression artifacts, are used and the detection of each have been well studied [1–7].

The need to be able to show image content over a range of resolutions has led to the creation of and current use of content-aware scaling. Shai Avidan of Mitsubishi Electric Research Labs (MERL) and Ariel Shamir of the Interdisciplinary Center and MERL, designed what they termed "seam-carving". This is a method that allows changing the size of an image by "carving-out or inserting pixels in different parts of the image." Seams are connected pixels from the top to the bottom or from the side to side of an image. Seam carving is done with the help of an energy function that is then used to identify the order of seams from lowest energy to highest energy [8]. The type of energy function used and the order that seams are removed can influence the resulting picture. Additionally, certain areas of the image can be artificially marked higher or lower in the energy function to protect from or ensure that certain areas are removed. An example of directed removal can be seen in Figs. 1, 2, and 3. Figure 1 is the original image. Figure 2 is the image with the marked pixels that will be protected (the man is highlighted in green) and removed (the man is highlighted in red). Figure 3 is the result of the seam carving.

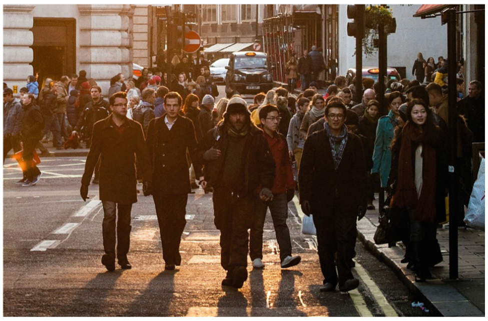


Fig. 1. Original image

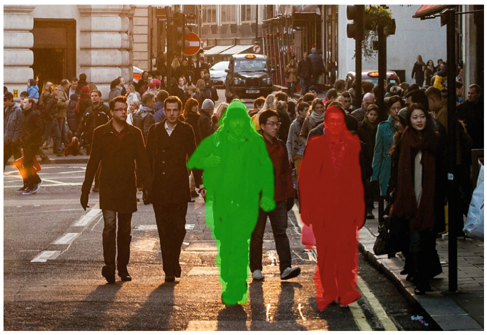


Fig. 2. An example image with overlays marking pixels in red and green. (Color figure online)

Fig. 3. An example of using seam carving to remove the red marked pixels while protecting the green marked pixels. (Color figure online)

2 Deep Learning

Deep learning is a specific type of machine learning that uses multiple layers of processing to find features of a data set and that categorizes the data into certain groupings based on the features. Most current deep learning models are based on an artificial neural network.

A new convolutional neural network (CNN) was released in 2017 named Inception-ResNet v2 by Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi from Google Incorporated [9]. The new CNN builds upon the previous version of inception v1 [10], v2 [11] and v3 [12] along with the use of optimized residual connections based on residual connections presented by He, Zhang, Ren, and Sun [13]. The diagram for this network can be seen in Fig. 2.

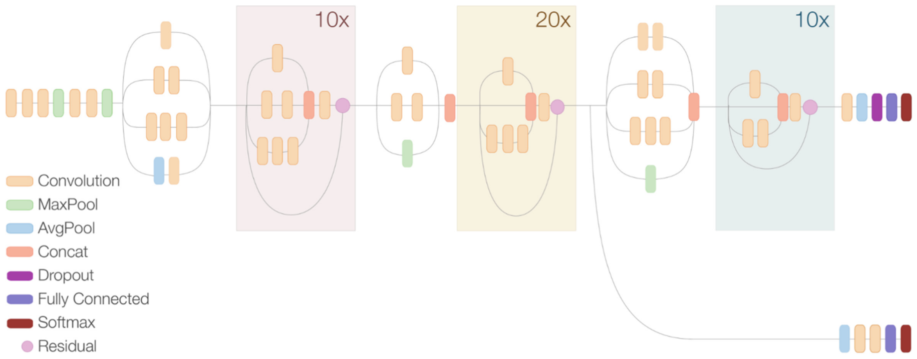


Fig. 4. Diagram of Inception ResNet v2 [14]

The layers that make up Inception ResNet v2 are convolution, pooling (max and average), concatenation, dropout, fully connected, softmax and residual. The convolution layer is where the tensor input is changed using a filter of a certain size. This filter then outputs the resulting tensor. The convolution process allows for extraction of features. Pooling allow shrinking the dimension of an input by looking at specific bundles of pixels and then using max or average on the bundle to determine a representative amount that then becomes the new value moving towards the input of the next layer. This is used to shrink the number of features from a convolution. A concatenation layer combines the tensors along a specified axis into a larger tensor. A dropout layer randomly drops data from a tensor into an output tensor that is the same size based on an entered probability. A fully connected layer does the classification on the features and every node in the layer is connected to a node in the preceding layer. Lastly a residual layer allows the skipping of certain layers to re-enforce the learning by addition of the tensor that skipped the layers with the output tensor that when through the layers.

In this paper, the Inception ResNet v2 network is fine-tuned to classify JPEG images that have been tampered, using seam carving, versus untampered. This paper seeks to study if Inception ResNet v2 can reliably determine if an image has been tampered via seam carving.

Recent research into using deep learning to detect and localize image tampering by way of resampling was shown by Bunk et al. [15] to be successful for copy-clone, removal, and splicing that was in the NIST Nimble 2016 dataset. This dataset contains images that were tampered in an advanced way to beat current state of the art detection techniques. Two methods of deep learning (CNN and LSTM) were used on Radon transformed data.

The application of external metadata allowed detection of tampering of images by Chen et al. [16]. They viewed images containing weather clues and compared them to the weather service for the same date and time. The correlation of the information was used to identify anomalies. The process could be further applied to also aid in detecting manipulations by comparing if clothing in images was weather appropriate for the date and location.

Deep learning was used by Shan [17] to learn from pathology patches to identify micro aneurysm in fundal images. They fine-tuned the training through a final iteration that allowed the accuracy results to move from high 80 s to low 90 s. The fine tuning used a stacked sparse autoencoder to do the deep learning. Autoencoders were part of early machine learning.

Deep learning was shown in research by Kalpana and Amritha to be able to identify “manipulations like median filtering, Gaussian blurring, resizing and cut and paste forgery..with… an average accuracy of 97%.” [18] Their approach used the addition of a prediction error filter to get at the relationship of the pixels and not the content of the picture.

This paper uses the novel approach of re-tuning an object recognition model, Inception ResNet v2, to detect JPEG image tampering by vertical seam carving. The hypothesis is that the object detection training will allow the model to re-train to be able to learn to find images that have been tampered via vertical seam carving. Maybe through identification of objects that don’t appear as they should or finding discrepancies in the colors or energy.

3 Methodology

A dataset was constructed from the 2017 ImageNet challenge test dataset containing 5500 images [19]. The first step cropped out a random selection from the original image with a size of 358 × 299. The untampered image is produced by cropping out a 299 × 299 selection randomly within the x axis of the original selection. Figure 3 shows an example untampered image.

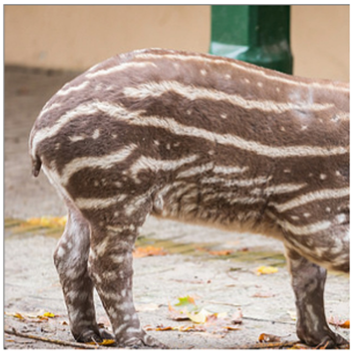


Fig. 5. Untampered image

The original selection is then evaluated for the seams that will be removed. Figure 4 shows the original selection with green lines showing the seams that will be removed from the image.

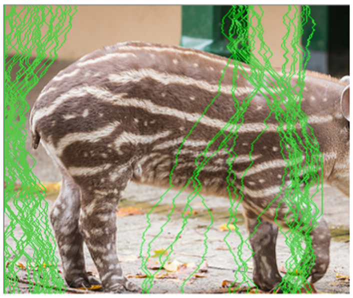


Fig. 6. Expanded selection with seams indicated in green (Color figure online)

The last step creates the tampered image using the seam removal process, removing 59 vertical seams, which are marked in green in Fig. 4, from the width of the image, resulting in a 299 × 299 image. Figure 5 shows what an example tampered image looks like with the seams removed. Notice that the tail, all four legs, and the ear can be seen and that there is no noticeable change to the pattern of the fur.

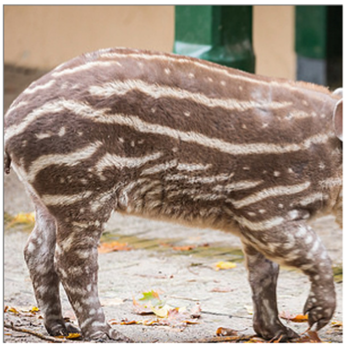


Fig. 7. Tampered image with seams removed

This process led to a dataset with 5500 tampered and 5500 untampered images of size 299 by 299. The Python function, imwrite, writes the JPEG files at a quality of 95 (https://docs.opencv.org/3.0-beta/modules/imgcodecs/doc/reading_and_writing_images.html#imwrite). This size was chosen to match the input size of the Inception ResNet v2 classifier. The image classifier will be retrained into a binary classification of tampered or untampered using TensorFlow (Fig. 7).

To convert the tampered and untampered images from JPEG format into tfrecords, required for the retraining method, the download_and_convert_data.py script was used from https://github.com/tensorflow/models/tree/master/research/slim. Through this process 1,000 images were reserved in a validation set and the other 10,000 images were placed into a training set.

The TensorFlow library was used to create a convolutional neural network (CNN) allowing training on the data set for identification of tampered images and identification of the method used. TensorFlow is an open source library that was created by Google for numerical manipulation within data flow graphs where the nodes are mathematical operations and the edges are where the multidimensional data arrays (tensors) are transferred between. TensorFlow allows building deep neural networks easier than coding them from scratch for every use case. TensorFlow can be included in many different programing languages and for the purposes of this proposal will be used is in Python.

The retraining script and evaluation script were used from here, https://github.com/kwotsin/transfer_learning_tutorial. These scripts output the streaming accuracy metric into a summary scalar (accuracy and validation_accuracy) which allows easy tracking throughout in TensorBoard.

4 Results

The training accuracy is charted in Fig. 8 with the y axis the accuracy percentage and the x axis the number of global steps in the training and Fig. 9 shows the text of the x and y coordinates of the point at the rightmost position. Once the retraining ended at 10 epochs, the accuracy percentage had grown to 68%.

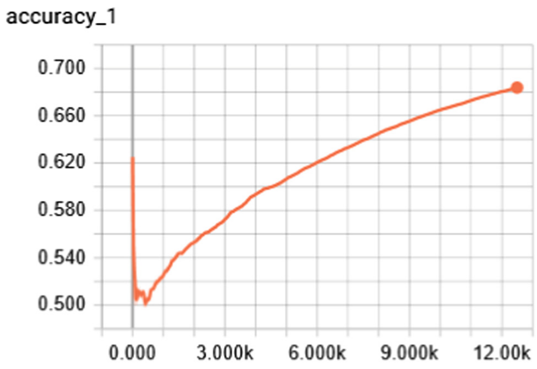
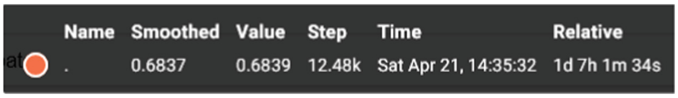


Fig. 8. Plot of training accuracy during training. 10 epochs and 1 day and 7 h later



	Name	Smoothed	Value	Step	Time	Relative
●	.	0.6837	0.6839	12.48k	Sat Apr 21, 14:35:32	1d 7h 1m 34s

Fig. 9. Details for the right-most point of the right-most point in Fig. 8 - 68%

The validation script accuracy is charted in Fig. 10 with the y axis the validation accuracy percentage and the x axis the number of global steps in the validation and Fig. 11 shows the textual information of the x and y coordinates of the point at the rightmost position. After 10 epochs, the validation accuracy stabilizes around 82%.

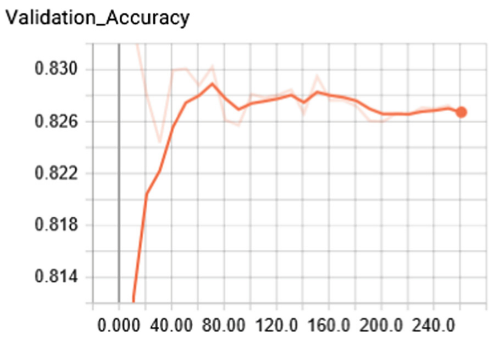
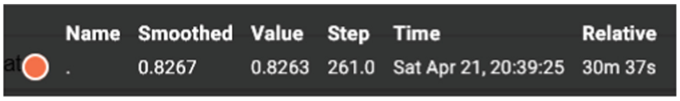


Fig. 10. Plot of validation accuracy against the number of steps in the validation. 1 epoch and 30 min later



	Name	Smoothed	Value	Step	Time	Relative
●	.	0.8267	0.8263	261.0	Sat Apr 21, 20:39:25	30m 37s

Fig. 11. Details for the right-most point of the validation accuracy plot in Fig. 10 - 82%

There can be seen a discrepancy between the resulting training accuracy and the validation graphs and the numbers found. This can be explained due to the training not running until an equilibrium is reached. In Fig. 12, a longer running training can be seen – it ran for 3 days and 21 h compared to Fig. 6 which ran for 1 day and 7 h. The accuracy moved from 68% to 76% between the shorter run to the longer training session.

The y axis of Figs. 8, 10, and 12 is the accuracy percentage and the x axis shows the number of steps. Each step is a batch of images that have been processed. The number of steps directly correlates to the amount of time that was required and the number of images that was in each image set to do the training or validation in Figs. 6 (12.48k steps), 8 (261 steps), and 10 (37.49k steps) (Fig. 13).

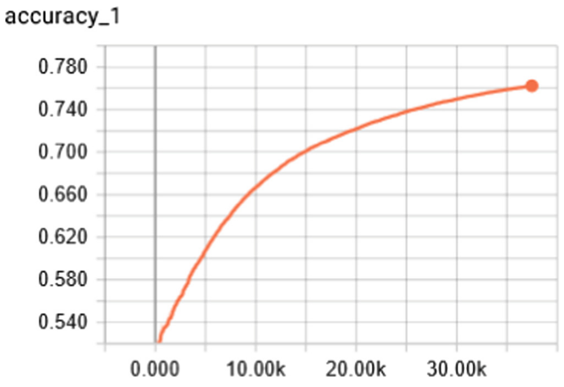
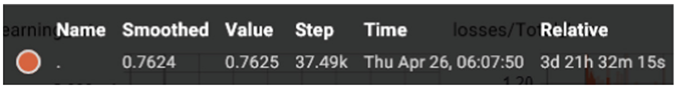


Fig. 12. A plot of training accuracy against the number of steps during training; 30 epochs and 3 days and 21 h later



earnin	Name	Smoothed	Value	Step	Time	losses/To	Relative
●	.	0.7624	0.7625	37.49k	Thu Apr 26, 06:07:50		3d 21h 32m 15s

Fig. 13. Details for the right-most point of the plot in Fig. 12 - 76%

5 Conclusions

The Inception ResNet v2 model, the best at object detection across multiple classifications, does perform well when it is retrained for binary classification of tampered images. Figure 6 demonstrates that as training went on, the accuracy kept climbing and combined with Fig. 8, it is expected that an accuracy in the mid-80 percent isn't unreasonable if the training is left to continue or additional images were provided.

The validation percentage of around 80% confirms that the retraining of the model isn't overfitting to the data set. It also supports the idea that the training accuracy would continue to move higher until it was close or above the validation accuracy.

After reviewing the false images, there is a higher incidence of false negatives than false positives. This supports a suspicion that there may be image manipulation being done during the training that may have altered the truthfulness of the label and thus caused confusion with the model. For example, if the tampering in an image was localized and that area wasn't included in a crop distortion then the model would be given essentially an untampered photo but falsely labeled as tampered.

6 Future Work

This paper restricted the seam removal to only vertical seams and future work should confirm that expanding the seam removal to seams going in any number of directions has the ability detect image tampering. For example, image preparation with horizontal seams or diagonal seams would allow broadening the usefulness and applicability in the real world.

For future experimentation, a test to check if the model is looking at the image and not the format of the image. This may allow expanding the ability to detect seam carving in various file formats and even screen shots of tampered images where the artifacts of the file format and compression may be lost or hidden.

Reducing the image size for the training may allow a higher accuracy and a reduced amount of false predictions. This would ensure that any tampering that was done to the image would be included in the training and potentially help with the accuracy of false negatives where the tampered part of the image wasn't looked at during the training.

A real-world application of this model into a browser or email plugin or social media API would allow watermarking a suspected image to grab an end-user's attention to the possibility of image tampering. This may help to reduce false information being presented as pictures from making the leap from entertainment to fake news.

Acknowledgments. We highly appreciate the support for this study from the National Science Foundation under Award #1318688 and from the SHSU Office of Research and Sponsored Programs under an Enhanced Research Grant.

References

1. Hsu, C.-M., Lee, J.-C., Chen, W.-K.: An efficient detection algorithm for copy-move forgery. Presented at the May (2015)
2. Chen, C., Shi, Y.Q., Su, W.: A machine learning based scheme for double JPEG compression detection. In: 2008 19th International Conference on Pattern Recognition. ICPR 2008, pp. 1–4. IEEE (2008)
3. Chen, Y.-L., Hsu, C.-T.: Detecting recompression of JPEG images via periodicity analysis of compression artifacts for tampering detection. IEEE Trans. Inf. Forensics Secur. **6**, 396–406 (2011). https://doi.org/10.1109/TIFS.2011.2106121
4. Dirik, A.E., Memon, N.: Image tamper detection based on demosaicing artifacts. Presented at the November (2009)

5. Farid, H.: Image forgery detection. IEEE Signal Process. Mag. **26**, 16–25 (2009). https://doi.org/10.1109/MSP.2008.931079
6. Liu, Q.: An approach to detecting JPEG down-recompression and seam carving forgery under recompression anti-forensics. Pattern Recogn. **65**, 35–46 (2017)
7. Liu, Q.: An improved approach to detecting JPEG seam carving under recompression. IEEE Trans. Circuits Syst. Video Technol. (in Press)
8. Avidan, S., Shamir, A.: Seam carving for content-aware image resizing. ACM Trans. Graph. **26**, 10 (2007). https://doi.org/10.1145/1239451.1239461
9. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.: Inception-v4, inception-ResNet and the impact of residual connections on learning. arXiv:160207261 Cs. (2016)
10. Szegedy, C., et al.: Going deeper with convolutions. Presented at the June (2015)
11. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv:1502.03167 (2015)
12. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. Presented at the June (2016)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. Presented at the June (2016)
14. Alemi, A.: Improving inception and image classification in TensorFlow, https://research.googleblog.com/2016/08/improving-inception-and-image.html
15. Bunk, J., et al.: Detection and localization of image forgeries using resampling features and deep learning. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 1881–1889. IEEE (2017)
16. Chen, B.-C., Ghosh, P., Morariu, V.I., Davis, L.S.: Detection of metadata tampering through discrepancy between image content and metadata using multi-task deep learning. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 1872–1880. IEEE (2017)
17. Shan, J., Li, L.: A deep learning method for microaneurysm detection in fundus images. Presented at the June (2016)
18. Kalpana, K., Amritha, P.P.: Image manipulation detection using deep learning in TensorFlow. Int. J. Control Theory Appl. **9**(40), 221–225 (2016)
19. Russakovsky, O., et al.: ImageNet large scale visual recognition challenge. arXiv 1409 (2014)