参赛队员姓名：袁昊、张宝瑾

中学：深圳市南山区荟同学校

省份：广东省

国家/地区：中国

指导教师姓名：李正 、孟鑫

指导教师单位：深圳市南山区荟同学校

论文题目：F-SegNet: Image Forgery Detection based on Instance Segmentation and ELA-NMA Features

F-SegNet: Image Forgery Detection based on Instance Segmentation and ELA-NMA Features

Hao Yuan, Baoqi Zhang
Mars Laboratory, Whittle School & Studios Shenzhen Campus
Email: {pyuan, bzhan871}@whittleschool.org

Abstract

As technology develops, many people now have the access to image manipulation software. Numerous fake images flood towards the internet everyday. Traditionally, social media platforms hire a large amount of human workers to perform image verification, which is an expensive and laborious task. The society urgently needs an image forgery detection assistant that is more cost-efficient.

This paper proposes Forgery Segmentation Network (F-SegNet), a self-supervised-based verification model that integrates Mask R-CNN, image analysis methods, and self-supervised learning to check image forgeries. The main work and contribution of this research are as follows:

- Provides **Auto-Labeled Image Splicing (ALIS) dataset**, which comprised 224,388 spliced images. Traditional image verification datasets usually lack pixel-level labels for forgery instances. By using image segmentation techniques, ALIS dataset automatically generates spliced images with pixel-level ground truth masks.
- Converts the original **image verification problem** into a **forgery instance segmentation problem**. Comparing to other pixel-level forgery detection models that generate blurry results, adapting **Mask R-CNN**, an image segmentation model, enabled F-SegNet to label forged areas with clean outlines and thus help observers locate forged area better.
- Implements **Image Analysis Methods (IAMs)** in Mask R-CNN to improve the model's ability in detecting forgeries. This paper combines **Error Level Analysis (ELA)** and **Normal Map Analysis (NMA)** with Mask R-CNN to help F-SegNet locate forged areas more precisely.
- Integrates **MoCo-v2, a self-supervised learning method,** with Mask R-CNN. The implementation of a self-supervised learning model allows F-SegNet to utilize limited training data better and achieve a better performance with less training time.

After training on ALIS dataset, F-SegNet achieves an average AP score of 88.090 and 89.062 on Bbox detection and segmentation tasks, respectively. Both the implementation of IAMs and self-supervised learning are proven to be valid methods in improving the model's performance.

This project is made open-source on: https://github.com/Merxon22/F-SegNet. ALIS dataset is uploaded on (VPN required): Dropbox.

Keywords: *Instance Segmentation, Self Supervised Learning, Image Verification, Error Level Analysis, Normal Map Analysis, Image Splicing*

Contents

1 Introduction

1.1 Background

In our modern world, image editing tools such as Adobe Photoshop are becoming more accessible to the public. Almost anyone with a smart device is able to modify images and upload them to the internet. Fig.1 (a) is a picture of our research team standing with Andrew Ng, a world-famous computer scientist. But is Andrew Ng really our project tutor? The answer is no. People edit photos for various reasons: some people do it without harmful intentions, but some do it purposely to achieve their malicious goals.

One example of online content forgeries is fake parkour photos and videos. Parkour has become a more and more popular sport among teenagers in recent years. Traceurs[1] from all over the world upload astonishing images and video clips of themselves running, jumping, and traversing around the urban environment. However, some of the contents are discovered to be forged. Fig.2 shows parkour images of people running on the edge of buildings or jumping from a high place. These traceurs often use techniques like copy-move, trimming, and video cuts to fake the audience. This makes the people believe that traceurs can easily perform dangerous tasks without any protection. These forged contents pose a serious threat to teenagers' safety because young people who lack safety awareness might imitate those dangerous actions and result in injuries or even deaths.

Fake images do not only affect people's daily life, but also heavily impact global news and politics. After the US president election in 2020, a Trump campaign spokesperson posted a manipulated image of a newspaper on Twitter, which claimed that "Florida pushes President Gore over the top with bare majority"[2]. However, it was soon discovered that the news publisher had never featured a headline about "President Gore" and the image was forged from another newspaper. Similarly, during the Zhengzhou flood in 2021, a photo of a humpback whale swimming on the street got spread crazily, saying that animals from the aquarium had escaped[3]. Soon, this news was proven fake and the humpback whale was manually added to the original photo using image editing software. Countless images on the internet are forged, such as "Failed rocket

Figure 1: *Is Andrew our real project tutor? The answer is no. F-SegNet is capable of identifying a forged person instance from a group of people standing together (right).*

[1]Traceurs: a practitioner of parkour activities (https://en.wikipedia.org/wiki/Parkour)
[2]https://twitter.com/i/events/1325531738655793153
[3]https://baijiahao.baidu.com/s?id=1705875863773834699wfr=spiderfor=pc

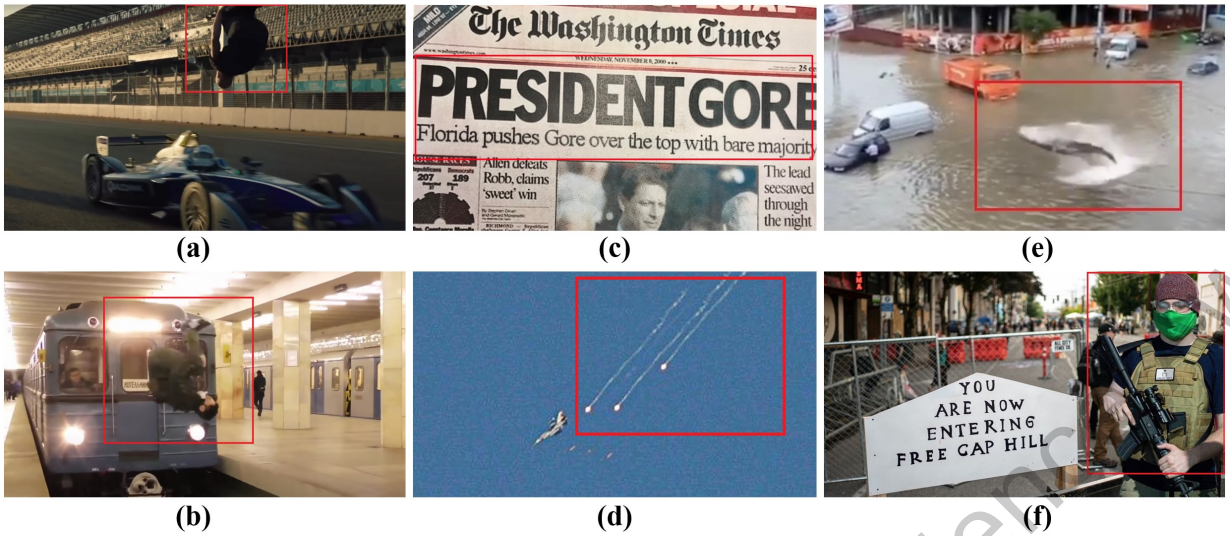


Figure 2: *Fig. (a-b) demonstrate fake images of traceurs jumping in dangerous places. Fig. (c-f) show fake images on social networks which are forged by using image editing software.*

launcher in Iran's show of military got removed", "News publisher runs digitally altered images in coverage of Seattle's protests', and "Forged Israeli fighter attacking photo" (Fig.2). These images can lead to results such as misinformation, the increase of public hatred, and the spread of fake news.

1.2 Typical Manipulation Methods

In the past century, image modification technology has developed rapidly. We classify major image manipulation methods into four categories: Copy-Move, Color Enhancement, Embellish, and Image Splicing.

1.2.1 Copy-Move

As the most simple yet effective method, copy-move is being widely used many image manipulators. To perform copy-move manipulation, a selected object will be duplicated on the same image for one or multiple times. As shown in Fig.3, a picture of news coverage of Iran's military exercises, a malfunctioning launcher was removed and media reporters copied and pasted the launched missiles to non-overlapping positions. This results in the effect of exaggerating

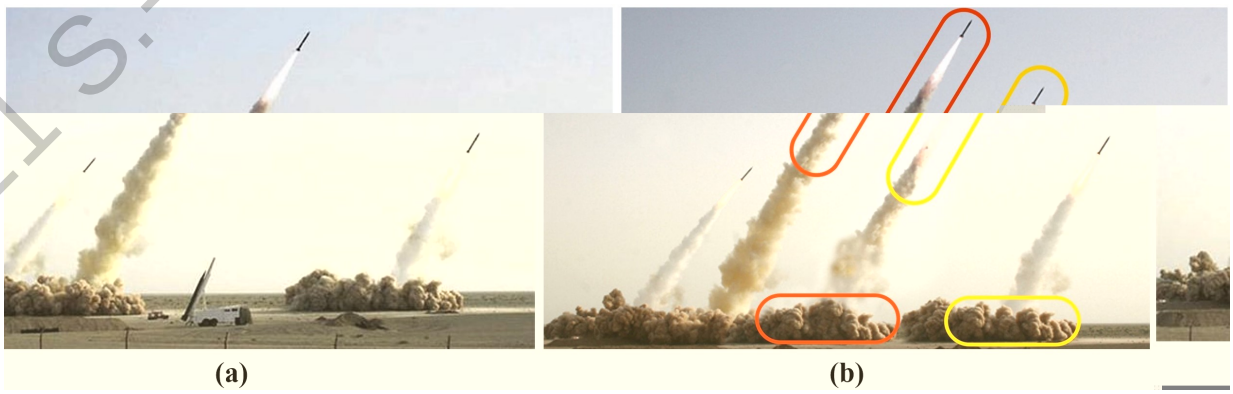


Figure 3: *In photo (a), a failed launcher is replaced by another missile in the image (b).*

(a) **(b)**

Figure 4: *In image (a), the original color of the leaves is changed from green to orange (b). People might interpret this photo as taken during autumn, which is actually taken during spring.*

military capabilities, which greatly misled the judgment of military strength.

1.2.2 Color Enhancement

Color enhancement is also an effective approach to modify images. Based on the situation, people may choose to either modify the color of the entire image or specific objects. Color enhancement manipulation often involves adjusting the contrast, brightness, saturation, and hue of the picture. In most cases, the purpose for modifying the color is to change the semantics information contained in the picture. As shown in Fig.4, the leaves in the original image are green. After modifying the hue, the season represented in the image becomes autumn.

1.2.3 Embellish

The use of image embellish rose with the rapid development of social media. As one of the most commonly used image modification methods on the internet, image embellish is widely applied to make oneself look more attractive. Image embellish manipulation often involves techniques such as liquify and reshaping. When performing liquify, the human facial muscles are indented through liquefaction to make the face look thinner and younger. Such beautification can be seen frequently on social platforms. Another tampering method is skin grinding, which is also known as image polishing. Skin grinding is usually performed by reducing noises presented in one's forehead cheek, or chin, so that one's skin appears to be smooth. This common cosmetic method can remove wrinkles and swelling on the original face. In most cases, users who upload their photos will use modified photos in order to attract attention. This may result in people using beautified photos to deceive social media users ad perform illegal acts such as defrauding money.

1.2.4 Image Splicing

The last commonly used image manipulation method is image splicing, which is moving a section of an image into another image. This simple but effective image stitching manipulation is enough to change the original meaning of the image. As shown in Fig.5, pets are not supposed to be brought on subways. However, we added a cat next to a man and made him seems to be breaking the rules by bringing his pet cat onto the subway. This simple modification cost us less

(a) **(b)**

Figure 5: *Image splicing modification example. A cat is added to the spliced image (b).*

than three minutes but can completely change the image. Image splicing can be used in any circumstances: ranging from generating fake evidence in a criminal case to illegally creating forged documents. Considering how easy it can be performed but the large impact it may potentially result in, image splicing will be the main focus of this research.

1.3 Related Works

In recent years, researchers tried to detect image forgery using various approaches. In 2018, Mingyoung Huh et al. [1] used features such as EXIF data to verify an image's consistency. This self-consistency-based model was trained on real photos and was able to learn from unary and pairwise methods. By extracting the metadata from the image, this model can find areas that are inconsistent with other reference parts, and thus localize the modified portion. However, since the model uses the consistency of the entire image, it cannot effectively detect copy-move image manipulations (the modified part comes from the same image, meaning that it has the same consistency). Also, over-exposed and under-exposed regions are sometimes marked as inconsistent because these areas share a uniform color. Lastly, the self-consistency model does not perform very well on locating small objects.

In 2019, Sheng-Yu Wang et al. [2] proposed a method that detects photoshopped faces. This model specifically focuses on detecting image warping that is conducted by Adobe Photoshop. The team first scripted the Face-Aware Liquify (FAL) tool in Photoshop to generate manipulated human faces and experimented using a dilated residual network variant model. This method was able to detect face warpings made by photo editing software and undo the changes. Although the FAL-detector has achieved significant results in human face detection, it is still limited to warping manipulations. Furthermore, it cannot detect color enhancements or faces of other animals.

ManTra-Net [3], a manipulation tracing network that can localize forged image regions was

proposed by Yue Wu et al. in 2019. This model comprises two sectors: image manipulation trace feature extractor and local anomaly detection network. In the first part, ManTra-Net subdivides image manipulation into 385 fine-grained types and thus enhances the test accuracy with the backbone architecture developed from VGG [4], ResNet [5] and DnCNN [6]. Then, the manipulation tracing analysis is passed to the anomalous feature extraction, where the model will holistically evaluate whether a pixel is modified or not. Despite the model integrating various image manipulation methods and showing an outstanding performance in image verification tasks, it still persists several limitations, namely detecting completely machine-generated images, detecting images contaminated with high correlated noise, and detecting images with multiple manipulations. At the same time, ManTra-Net is not able to detect images that have a bit depth of 32 bit (4-channel photos such as PNG and TIFF images).

When looking towards the fake news detection field, other approaches such as Ti-CNN [7] combines both textual inputs and image inputs to verify the contents in a news. Fakeddit [8] is a dataset established by collecting posts and images from Reddit, one of the world's biggest online forum. All samples in Fakeddit are categorized into six groups: true, satire, misleading content, manipulated content, false connection, and imposter content. Similar to Fakeddit, CASIA2.0 and PS-Battle dataset [9] are both image datasets dedicated to image manipulation detection. CASIA2.0 dataset is established for image forgery detection tasks. It contains 5123 tampered images in JPG or TIFF format, which includes 3274 copy-move images and 1849 spliced images. PS-Battle dataset is constructed based on the "photoshopbattles" subreddit, where large amount of professional artists post manipulated images regularly.

Besides focusing on manipulated images on social media, researches such as [10, 11, 12] also stated that the number of image forgery in research papers has increased over the years. In "The prevalence of inappropriate image duplication in biomedical research publications" [10], researchers pointed out that most image forgeries in academic fields are conducted through transformation, cropping, duplication, and image enhancements. These misleading images are often caused by carelessness, but are sometimes modified intentionally. Several image verification methods for academic research papers have been proposed by the researchers. An automatic detection framework for image manipulations [11] used software operations to "crop" images from the research paper and feed them to several image checking software to perform the verification. Another forgery detection method [12] uses Scale-Invariant Feature Transform (SIFT) [13] keypoint detection algorithm, RanSac algorithm, and other image manipulation detection methods with the combination of human labels to identify suspicious images. However, these methods are not suited for large amount of images because they require human to be involved during the verification process.

Typically, different model focuses on detecting different kinds of image manipulations. Many researches such as [14, 15, 16, 17, 18] performs image verification on copy-move regions. Such tasks often use a method like level set approach, SIFT, cellular automata and local binary patterns, and patch match. Other researches like [19, 20] detects image splicing manipulation and regions with forged contents, where wavelet decomposition and polar harmonic transform was used. In "An Evaluation of digital image forgery detection approaches" [21], it is concluded that most image verification models focus on detecting copy-move, forged image region, tampered region, and image splicing. Among all the detection types, copy-move detection is the most common one.

1.4 Motivation and Contribution

Considering the negative impact forged images can deal to our society, this research paper proposes F-SegNet, an self-supervised-learning-based image forgery detection framework that integrates Mask R-CNN [22], ELA [23], NMA [24], and MoCo-v2 [25]. This paper has four major contributions:

- Creates a self-labeling image dataset that quantifies image verification accuracy.
- Novelly converts image verification problem into forgery instance segmentation problem to raise F-SegNet's accuracy.
- Integrates multiple Image Analysis Methods (IAMs) with Mask R-CNN to improve the detection efficiency and precision.
- Combines Mask R-CNN with self-supervised learning method to better utilize numerous unlabeled data and thus mine semantics knowledge to improve the model's performance.

The rest of the paper is organized as follows: Section 2 introduces the auto-labeling dataset we provide for image splicing detection. Section 3 explains the architecture of our image verification algorithm. Section 4 compares the experiment results. Finally, Section 5 concludes the paper.

2 ALIS Dataset

In image verification field, most datasets lack pixel-level labels that indicate the actual modified area in the picture because marking these areas is a very labor-intensive job. Both CASIA2.0 and PS-Battle datasets do not include the localized "truth mask" for image manipulation. A fully labeled dataset with localized object mask is essential for the training and evaluation of image verification networks. In this section, we established Auto-Labeled Image Splicing (ALIS) dataset, which can automatically generate spliced images with pixel-level truth masks.

2.1 Image Collection

ALIS dataset is constructed based upon three image datasets: COCO, ImageNet, and Crowd Human. COCO [26] is a large-scale image dataset designed for object detection and image

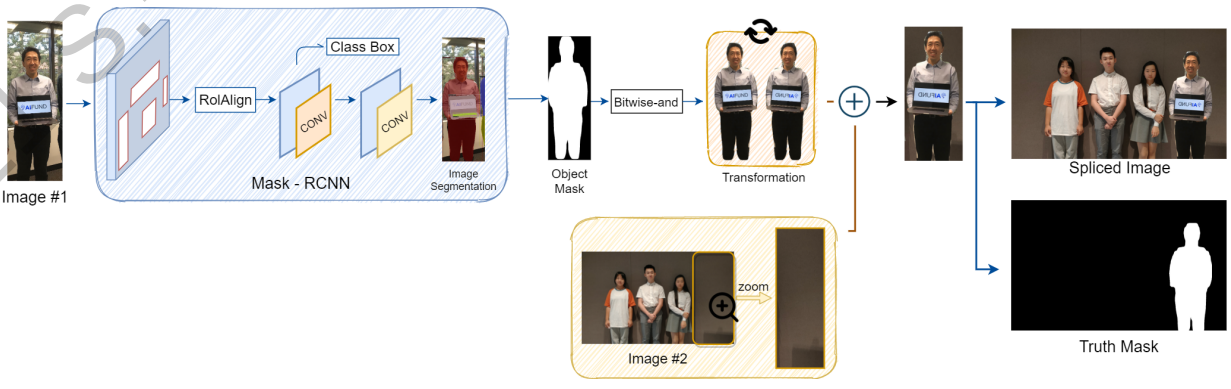


Figure 6: *ALIS dataset generation pipeline.*

segmentation tasks. ImageNet [27] is another large image dataset that contains high-definition pictures for object recognition and other visual studies. Eventually, Crowd Human [28] is a dataset comprised of photos that contain multiple human instances that can be used to enhance our model's performance when detecting human forgeries.

2.2 Mask R-CNN based Auto-Labeling Pipeline

ALIS dataset uses Mask R-CNN [22], an instance segmentation approach, to generate spliced images with clean outlines. A detailed discussion of Mask R-CNN can be found in Section 3.1. Since the entire process is conducted by machine, the labeling of the modified area will be automatically recorded and saved with the final output as well.

To perform the generation of forged images, ALIS dataset takes two inputs: an image that provides elements for image splicing (image 1) and a second image (image 2) that we want to "transplant" the element from image 1 onto (Fig.6). To perform the generation of ALIS dataset, Mask R-CNN model will first generate a prediction result based on all the existing instances in image 1. Instead of object detection approach which generates only a rectangle prediction box, image segmentation approach was used in order to ensure that the least amount of undesired background will be added into image 2. The object with the highest confidence score will be selected and then be pasted onto image 2 after random transformation. The ground truth mask for this operation will be saved while producing the forged image as well.

Algorithm 1 ALIS dataset generation process

Require: img_1 as input image 1; img_2 as input image 2; M as default Mask R-CNN model

1: $seg\_results = M_{segmentation}(img_1)$
2: $spliced\_object = max_{confidence}(seg\_results)$
3: $spliced\_area = img_1 \cap spliced\_object_{instance\_mask}$
4: $transformed\_spliced\_area = random\_transformation(spliced\_area)$
5: **Output**$(transformed\_spliced\_area + img_2)$ as spliced image
6: **Output**$(transformed\_spliced\_area \cap img_2)$ as truth mask

2.3 Annotation Quality and Data Distribution

During the experiment, however, we raise a concern that a model trained on ALIS dataset can only learn the ability to segment objects instances such as people and cars but not forgery instances. Therefore, three datasets are used to ensure that multiple objects exists in one image. By doing so, a person instance can be added into a group of people (Fig.6), so that the generated image will resemble real-world images better and reinforce the learning difficulty. Since a large amount of computer vision models use datasets in COCO format [26], we convert ALIS dataset into this style to ensure the universality of our dataset. Besides all the image files, datasets in COCO format contain a JSON file that stores the image information such as file name, file size, object mask, and object class. In ALIS dataset, only one class persists among all the images, which is the "forged" class. To create a JSON file that includes our dataset information, pycococreator [29], a library that converts image datasets into COCO format, is used to accomplish the task. Together, ALIS dataset provides a total of 224,388 machine-generated spliced images with their corresponding ground truth masks. We focus on generating only 5 categories of spliced

Figure 7: *Image (a) is the input image and image (b) is the truth mask for splicing area. Other pixel-level forgery detection models might sometimes generate a meaningless mask (c), while F-SegNet can generate a clean segmentation outline for suspicious region (d).*

images: Person, People, Car, Cat, and Chair, because they can represent some of the most commonly seen objects in our daily life. However, any researchers can further extend the dataset according to their need by easily using our image generation algorithm provided above.

3 F-SegNet Architecture

The traditional approach to perform image verification is to calculate every single pixel's probability that is being forged. This sometimes results in a output mask that look blurry and meaningless (Fig.7.c), which cannot effectively help the observer locate forged areas. In comparison, this paper converts an image verification problem into a forgery instance segmentation problem, considering the fact that both tasks require computer models to generate a predicted mask of a specific object. In F-SegNet, we wish to generate mask on the "forged" class. Therefore, Mask R-CNN [22], a popular and effective model in image segmentation field, is adapted into the model. Since such models consider the forged area as a whole object, F-SegNet can output a much cleaner mask with clearer outline comparing to other methods (Fig.7.d).

3.1 Backbone: Mask R-CNN

Mask R-CNN is an instance segmentation model based on Faster R-CNN [30]. It generates object mask for every individual object in the image while performing classification tasks. As shown in Fig.9, Mask R-CNN is divided into multiple stages to achieve this work. It typically uses ResNet [5] as its first-level feature extractor. ResNet, also known as Residual neural network, is aimed to solve the "degradation problem" found when performing training on deep-level neural networks. In ResNet, results from previous layers are allowed to affect the following layers. By doing so, ResNet significantly improved training loss in deep-level neural networks. In Mask R-CNN's feature extractor, the beginning layers will extract simpler features such as corners and edges while deeper layers will extract features that are more complex, such as a person or a car. During this process, a Feature Pyramid Network (FPN) [31] is implemented to allow higher level features to be passed down to the lower level layers directly and thus improve the

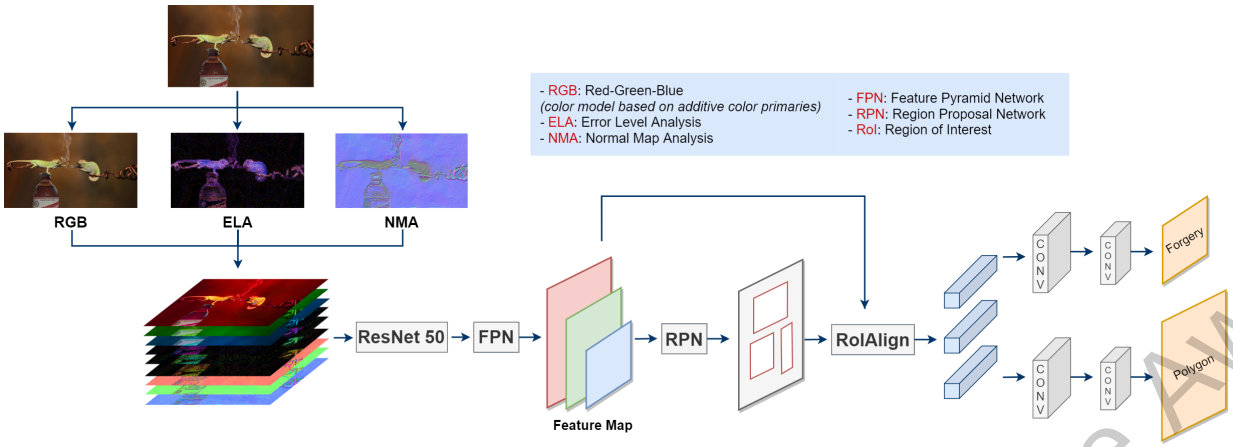
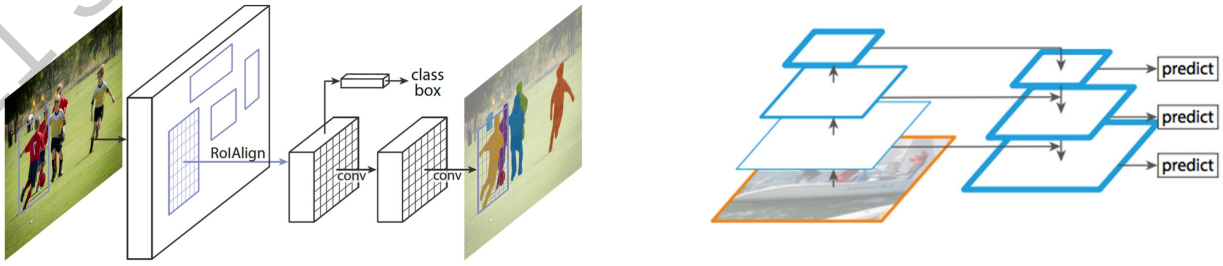


Figure 8: *The main architecture of F-SegNet, which is adapted from Mask R-CNN. The input is converted into a 9-channel image and the default backbone (ResNet-50) is replaced with model weights trained from MoCo-v2. Detailed description for IAM adaptation and self-supervised learning can be found in Section 3.2 and Section 3.3.*

extraction process (Fig.9). Eventually, the 1024×1024×3 input image will be converted into a 32×32×2048 feature map.

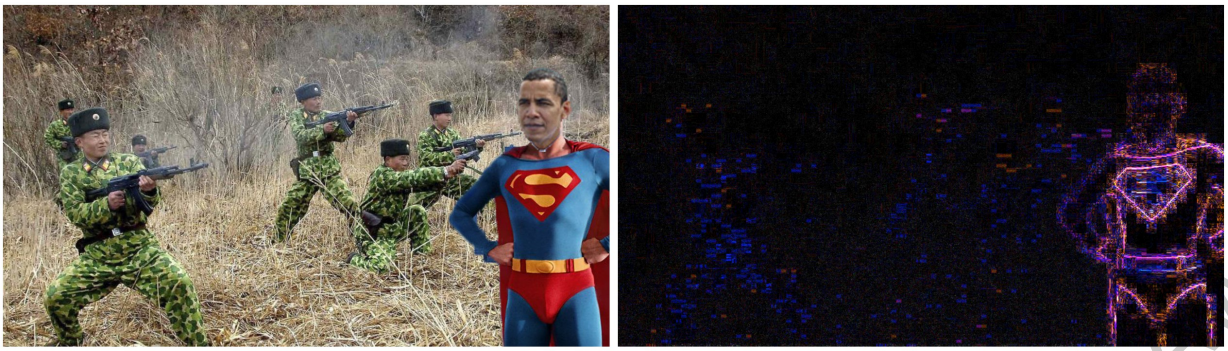
After the extraction of key features in the original image, a Region Proposal Network (RPN) will scan the image based on the "anchor points" in the image and propose regions that are likely to contain objects. This process is usually time-efficient because RPN scans over the generated feature map instead of the original image. The outputted region of interests (ROIs) will be passed down to an ROI classifier that categorize the class of the object and further adjust the bounding box of the instance. During the ROI classification process, ROI pooling and ROI alignment methods are introduced. ROI pooling is applied to ensure that the classifier can perform categorization on different image sizes by scaling the original input into a fixed size (7×7). However, when the output size passed down from the previous level is not a multiple of the classifier's input size, non-integer strides will occur. Rounding the stride to the nearest integer will make some information from the original image being discarded. To solve this problem, ROI alignment is applied, where the stride is remained as a non-integer value and the feature map is being sampled using bilinear sampling.

Eventually, to generate the object mask for instance segmentation, the pooled feature map will be passed to a fully convolutional network (FCN). FCNs are implemented because they retain spacial orientation, which is crucial for location-specific tasks such as generating object masks. The output mask of FCN is a 28×28 soft mask layer represented with float numbers, which will be eventually up-scaled to the original image size.



(a) *(b)*

Figure 9: *Mask R-CNN framework [22] (a) and FPN architecture [31] (b).*



(a) **(b)**

Figure 10: *ELA analysis extracts JPEG compression differences by re-saving an image. In the analysis result (b), Obama and his outfit is in high intensity values, indicating that this area has a different error level comparing to the entire image.*

3.2 Image Analysis Methods

Since many photo editors are able to minimize the trace of manipulation with image processing skills, it is difficult to detect forgeries with the information presented in the image only. Instead, we should unearth more hidden data and features through different image analysis methods.

Platforms such as FotoForensics[4] and Forensically[5] are both online image forgery detection tools that utilizes different IAM algorithms. As described by in [12], other researchers also proposed pipelines that detect image forgeries in research paper. However, all the methods mentioned above require human observer to perform the verification task and are not integrated with an automatic-detection model. In some circumstances, the results produced by such methods might confuse the observer because they output meaningless analysis results. In order to provide an effective and also efficient forgery detection framework, this paper combined two IAMs, ELA and NMA, into F-SegNet to further improve the model's performance.

3.2.1 Error Level Analysis (ELA)

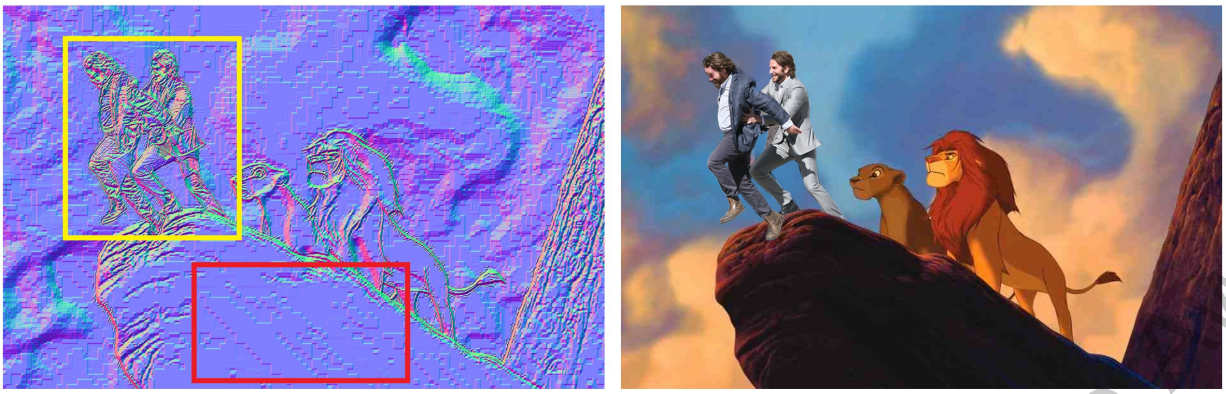
ELA [23] is an image analysis method based on the fact that JPEG is a lossy compression method. When every JPEG image is being re-saved, some of its original information will be discarded by the compression algorithm. Eventually, an image after many re-saves will reach an error rate where nearly no information will be further discarded. However, if a region is added to the original image after a few times of compressions, it might have a different error rate comparing other parts in an image. In ELA process the original image is re-saved at a known error rate, and the result are obtained by subtracting these two images:

$$ELA \leftarrow (img - img \cdot c\%) \cdot s \tag{1}$$

where *img* is the original input image, *c* is the compression ratio, and *s* is the error enhancement scale. If an image undergoes no modification, the ELA outcome should have similar intensity consistency for every object in the image. However, if an image is manipulated, ELA might find

[4]FotoForensics: http://fotoforensics.com/

[5]Forensically: https://29a.ch/photo-forensics/forensic-magnifier



(a) **(b)**

Figure 11: *Normal map analysis (b) captures the gradient of light intensity in an image. Real images usually contain areas with complex details due to random noises (yellow area). Computer-generated regions usually appear in flat or boxy patterns (red area).*

areas that have a different error rate, which are often indicated as high intensity output (Fig.10). Since ELA is based on the compression of images, it is good at capturing JPEG modification artifacts done by image editing software.

For image formats that undergo lossless compression (such as PNG images), the image information will not be discarded if every re-save is performed at a 100% quality level. Theoretically, ELA should find very little differences in the error rate among these lossless images. In our experiment, however, it is discovered out that ELA is still having a decent performance on PNG images. This is probably due to the fact that every photo taken by a camera is originally outputted as JPEG format, and then being converted into other formats during the image editing process. This characteristic allowed ELA to find image compression artifacts in almost every image disregarding the image file type.

3.2.2 Normal Map Analysis (NMA)

NMA [24] calculates the surface normal direction for every pixel in the image based on the light direction. Most of the time, light intensity is not evenly distributed in the image. Sections closer to the light source will have a higher intensity value while sections in the shadow will have a lower intensity value. In our model, normal map analysis uses the sobel operator [32] to compute the image gradient on both x and y axis and combine them into a vector representing the surface normal of each pixel. The red, green, and blue (RGB) value in a normal map indicates the values on the X, Y, and Z axis respectively.

NMA is particularly effective in detecting computer-generated regions. If an image is taken by a camera, random noises should appear on the entire image, even the area is visually smooth to human eyes. Therefore, when conducting NMA, the algorithm should find random gradient noises on flat surfaces. However, if a surface appears to be flat in normal map analysis, it probably means that this region is generated by a computer software, since image editing software usually does not take camera noises into account (Fig.11).

3.2.3 IAM Integration

To perform the integration of IAMs with Mask R-CNN, the input of ResNet-50 backbone has to be modified. In a typical ResNet-50 backbone, a 3-channel (RGB) image is passed into the neural network. In F-SegNet, the image array is resized into a 9-channel image, where the first three channels contain the original RGB information, the fourth to sixth channels contain ELA results, and the last three channels contain NMA results (Fig.8).

Algorithm 2 IAM adaptation process

Require: *img* as input image, in numpy array format
1: tmp_img = resave *img* at 90% error rate
2: ela_img = *img* - tmp_img
3: $Sobel_x, Sobel_y$ = compute sobel operation for *img* on both x and y axis
4: $nma\_img = Compute\_normal\_map(Sobel_x, Sobel_y)$
5: $output\_img$ = numpy array of size[$img.height, img.width, 9$]
6: **for** i in range 3 **do**
7: $output\_img[i] = img[i]$ //channel 0-2 represents original image
8: $output\_img[i+3] = ela\_img[i]$ //channel 3-5 represents ELA image
9: $output\_img[i+6] = nma\_img[i]$ //channel 6-8 represents NMA image
10: **end for**
11: **Output** $output\_img$

3.3 Self-Supervised Learning

As described in Section 2, image manipulation datasets often have labels only on whether the image is being forged. Most datasets do not have a pixel-level label on the forgery mask because locating forged areas is an expensive task. Therefore, self-supervised learning approach is implemented into F-SegNet. Comparing to supervised learning which requires external label data, self-supervised learning uses labels and information that are contained in the data itself and convert the problem into a supervised learning. Therefore, it can achieve a better performance on a limited dataset size. Moreover, when sufficient labeled data is provided, self-supervised learning can improve the model's performance to a further extent.

There are two types of self-supervised learning methods that are widely being used: pretext-based learning, which was previously used before, and contrastive learning, which is currently a more popular approach. Contrastive learning learns through comparing positive samples and negative samples. Comparing to other self-supervised learning methods, contrastive learning focuses more on abstract representation of the image instead of the pixel-level details. During training, a contrastive learning model will generate both positive and negative samples to learn with. The neural network will gradually develop a function to maximize the gap between two

Figure 12: *Implementation of DINO in a video clip [33].*

different types of samples.

One implementation of contrastive self-supervised learning is SimCLR [34]. For every image input, SimCLR will generate two correlated learning samples. Contrastive Loss Function, which reduces the gap between correlated samples and enlarges the gap between uncorrelated samples, will further adjust the model. During contrastive learning, however, models such as SimCLR requires a relatively large number of batch size. Therefore, Kaiming He et al. proposed Moco [35], also known as Momentum Contrast. In 2020, Xinlei Chen et al. improved MoCo and proposed MoCo-v2 [25]. Comparing to other contrastive learning methods, MoCo uses two encoders and updates network parameters during the process of training: one is a regular encoder and the other is a momentum encoder. As a result, MoCo has a less requirement on training device and can be implemented on more situations.

Another implementation of self-supervised learning is DINO [33]. DINO is a self-supervised developed by Facebook and is well-suited for training on random and unlabeled data (Fig.12). In DINO training, the model uses a "self-distillation" process, where a supervising teacher network and a learning student network are introduced. Both networks contain a Vision Transformer (ViT), and the teacher's momentum is set to an exponentially weighted average of the student ones:

$$\theta_t \leftarrow \lambda \theta_t + (1-\lambda)\theta_s \tag{2}$$

where θ_t represents teacher weights, θ_s represents student weights, and λ follows a cosine schedule during training process.

4 Experiments

4.1 Training Environment

The experiment is conducted on an 80-cores machine with 128GB memory installed and 2 RTX-6000 GPUs mounted. Ubuntu 18.04 LTS is installed as the operating system, and python 3.7 is used as scripting language. Unless specified, all trainings are performed with an iteration of 100,000, learning rate of 0.00025, and configured with a default ResNet-50 backbone. In addition, Detectron2 [36] is used as our framework and Colab is used as the visualization tool for training analysis. All the evaluations below follow COCO evaluation metrics[6].

4.2 Experiment Results

Table 1 gives the AP score of Bbox detection and segmentation of Basic Mask R-CNN after 20,000 iterations, where no pretrained weights are used. As presented in the table above, the evaluation score did not reach our expectation. By comparing the results after 10,000 and 20,000

Method		Bbox			Segmentation		
		AP	AP50	AP75	AP	AP50	AP75
Average	10000 Iter	4.654	16.749	0.873	4.908	14.622	2.272
	20000 Iter	9.697	28.706	3.049	13.668	28.756	11.505
	delta	+5.043	+11.957	+2.176	+8.760	+14.134	+9.233

Table 1: *The comparison of Basic Mask R-CNN model after 10,000 and 20,000 iterations.*

[6]COCO detection evaluation metrics: https://cocodataset.org/#detection-eval.

Method		Bbox			Segmentation		
		AP	AP50	AP75	AP	AP50	AP75
Person	Basic	39.656	74.468	38.259	45.520	73.546	48.827
	IAM	45.568	80.752	45.697	52.040	79.518	57.544
	delta	+5.912	+6.284	+7.438	+6.520	+5.972	+8.717
People	Basic	31.369	66.033	26.392	39.294	65.237	38.296
	IAM	46.834	82.934	47.638	54.595	82.536	61.022
	delta	+15.465	+16.901	+21.246	+15.301	+17.299	+22.726
Car	Basic	51.310	82.456	58.368	67.073	84.980	74.407
	IAM	54.611	84.984	63.864	71.111	87.548	77.709
	delta	+3.301	+2.528	+5.496	+4.038	+2.568	+3.302
Cat	Basic	49.721	82.511	54.765	62.359	83.249	70.534
	IAM	53.313	85.775	59.877	67.296	86.881	75.842
	delta	+3.592	+3.264	+5.112	+4.937	+3.632	+5.308
Chair	Basic	46.153	79.645	48.595	54.095	78.120	60.344
	IAM	52.103	84.360	58.263	60.127	83.450	67.009
	delta	+5.950	+4.715	+9.668	+6.032	+5.330	+6.665
Average	Basic	43.642	77.023	45.276	53.668	77.026	58.482
	IAM	50.486	83.761	55.068	61.034	83.987	67.825
	delta	+6.844	+6.738	+9.792	+7.366	+6.960	+9.344

Table 2: *Comparison between Basic Mask R-CNN and IAM-implemented Mask R-CNN after 100,000 iterations of training.*

iterations of training, it is shown that the AP score of 20,000 iterations has increased considerably. The result suggests that the low performance of the Basic Mask R-CNN model dose not comes from overfitting, and we had to seek other ways to improve the model.

As described in Section 3.2, image analysis methods (IAMs) can provide extra information to the neural network. Therefore, two popular IAMs, ELA and NMA, are implemented into the basic Mask R-CNN model and the training is extended to 100,000 iterations. Table 2 gives the AP scores of the IAM-Adapted Mask R-CNN model, where no pretrained weights are used and the input was converted into a 9-channel image. After the implementation of IAMs, the model's performance increases in every aspect. The average AP score for Bbox detection and segmentation has increased 6.844 and 7.366, respectively. The image analysis methods being used in the model gives additional information to F-SegNet and thus can strengthen its ability to locate forged areas. Therefore, combining IAMs with Mask R-CNN is a valid method to improve model performance. However, the model's AP score after the integration of IAMs is still lower than our expectation.

Next, the backbone of Mask R-CNN model is replaced with model weights trained from MoCo-v2 [25], a self-supervised learning model, to let F-SegNet utilize limited data better. Table 3 compares the performance of IAM-Adapted Mask R-CNN and MoCo-v2+IAM-Adapted Mask R-CNN, where the IAM model uses a default ResNet-50 model as its backbone and the

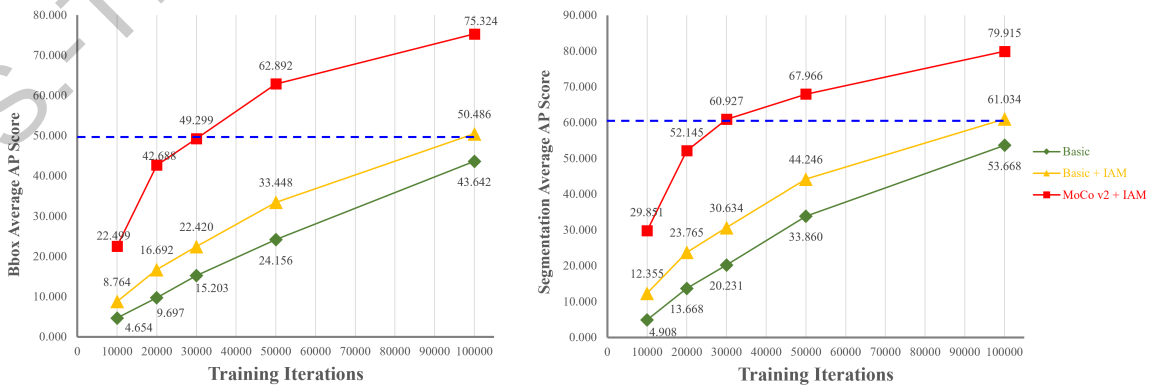


Figure 13: *Comparison between Basic Mask R-CNN, IAM-Adapted Mask R-CNN, and MoCo-v2+IAM-Adapted Mask R-CNN on Bbox detection (a) and segmentation (b).*

Method		Bbox			Segmentation		
		AP	AP50	AP75	AP	AP50	AP75
Person	IAM	45.568	80.752	45.697	52.040	79.518	57.544
	MoCo+IAM	72.560	95.873	83.899	74.594	95.329	86.435
	delta	+26.992	+15.121	+38.202	+22.554	+15.811	+28.891
People	IAM	46.834	82.934	47.638	54.595	82.536	61.022
	MoCo+IAM	71.808	95.179	82.866	74.211	94.468	85.900
	delta	+24.974	+12.245	+35.228	+19.616	+11.932	+24.878
Car	IAM	54.611	84.984	63.864	71.111	87.548	77.709
	MoCo+IAM	78.954	95.223	89.315	87.576	95.772	93.185
	delta	+24.343	+10.239	+25.451	+16.465	+8.224	+15.476
Cat	IAM	53.313	85.775	59.877	67.296	86.881	75.842
	MoCo+IAM	77.490	95.569	87.953	84.356	96.272	92.328
	delta	+24.177	+9.794	+28.076	+17.060	+9.391	+16.486
Chair	IAM	52.103	84.360	58.263	60.127	83.450	67.009
	MoCo+IAM	75.806	95.613	86.758	78.839	95.058	88.357
	delta	+23.703	+11.253	+28.495	+18.712	+11.608	+21.348
Average	IAM	50.486	83.761	55.068	61.034	83.987	67.825
	MoCo+IAM	75.324	95.491	86.158	79.915	95.380	89.241
	delta	+24.838	+11.730	+31.090	+18.881	+11.393	+21.416

Table 3: *Comparison between IAM-Adapted model and MoCo-v2+IAM-Adapted model after 100,000 iterations of training.*

MoCo+IAM model uses model weights from MoCo-v2 as its backbone. With the implementation of self-supervised learning, the performance score is boosted significantly. Comparing to the IAM model, MoCo-v2+IAM model shows a 24.838 and 18.881 increase in the average AP score of Bbox detection and segmentation, respectively.

Fig.13 compares the three models' average AP result during the 100,000 iterations of training. The MoCo-v2+IAM-Adapted model has demonstrated an overall leading performance throughout the entire training process. Moreover, as shown by the blues lines, the implementation of self-supervised learning allows F-SegNet to achieve a better performance with less training: a MoCo-v2+IAM model that is trained for only 30,000 iterations is capable of competing with a IAM-Adapted model which had been trained for 100,000 iterations.

Eventually, the training on MoCo-v2+IAM-Adapted Mask R-CNN model is extended to 500,000 iterations. Fig.14 (a) demonstrates the total loss and Fig.14 (b) shows average AP scores for Bbox detection and segmentation. After roughly 2 days of training on our machine, F-SegNet has reached a total loss of 0.088 and achieved average AP scores close to 90. Among all the five categories in ALIS dataset, "Car" and "Cat" subcategories has received the highest AP scores. Considering the outstanding performance of F-SegNet, it can be potentially be deployed into certain image verification fields (Fig.15).

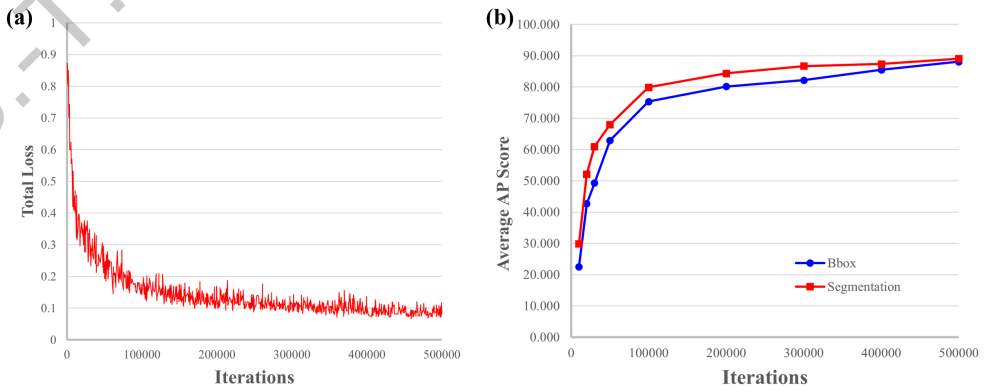


Figure 14: *Model total loss (a) and Bbox detection and segmentation average AP score (b) after 500,000 iterations of training.*

Methods	Bbox			Segmentation		
	AP	AP50	AP75	AP	AP50	AP75
Person	87.197	98.565	94.490	85.379	98.654	95.614
People	85.913	98.408	94.217	84.542	97.653	94.431
Car	88.650	98.406	94.305	94.550	98.626	98.275
Cat	89.848	98.769	95.625	92.930	98.779	97.772
Chair	88.840	98.442	94.452	87.909	97.662	95.442
Average	88.090	98.518	94.618	89.062	98.275	96.307

Table 4: *Detailed AP score for Bbox detection and segmentation tasks after 500,000 iterations of training.*

5 Conclusion

In this paper, we proposed ALIS dataset and F-SegNet. ALIS dataset is an auto-labeled dataset that automatically generates spliced images with pixel-level truth masks. By combining three well-known image datasets: COCO, ImageNet, and Crowd Human, we together generated 224,388 images for "Person", "People", "Car", "Cat", and "Chair" categories. This dataset is eventually converted into COCO format and uploaded on Dropbox[7] to be easily accessible to any researcher.

F-SegNet is an image forgery detection framework that combines Mask R-CNN, modern image analysis methods (ELA and NMA), and self-supervised learning to improve the model's performance. By using Mask R-CNN, a popular image segmentation model, we converted a forgery detection problem into instance segmentation problem. We also replaced the backbone of Mask R-CNN with model weights from a MoCo-v2 model, which allowed F-SegNet to achieve the same outcome with only 30% of the training time (Fig.13). Eventually, after training for 500,000 iterations, F-SegNet received an average AP score of 88.090 and 89.062 in Bbox detection and segmentation tasks, respectively. As shown in Fig.15, the distinctive performance of F-SegNet has allowed it to perform certain forgery detection tasks on spliced images.

Figure 15: *F-SegNet performance on spliced images.*

[7]https://www.dropbox.com/sh/r94z9f7ov66gj3i/AACLXFgDuogrSK-jiMJPJ9YFa?dl=0

References

[1] M. Huh, A. Liu, A. Owens, and A. A. Efros, "Fighting fake news: Image splice detection via learned self-consistency," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 101–117, 2018.

[2] S.-Y. Wang, O. Wang, A. Owens, R. Zhang, and A. A. Efros, "Detecting photoshopped faces by scripting photoshop," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10072–10081, 2019.

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

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

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[6] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE transactions on image processing*, vol. 26, no. 7, pp. 3142–3155, 2017.

[7] Y. Yang, L. Zheng, J. Zhang, Q. Cui, Z. Li, and P. S. Yu, "Ti-cnn: Convolutional neural networks for fake news detection," *arXiv preprint arXiv:1806.00749*, 2018.

[8] K. Nakamura, S. Levy, and W. Y. Wang, "r/fakeddit: A new multimodal benchmark dataset for fine-grained fake news detection," *arXiv preprint arXiv:1911.03854*, 2019.

[9] S. Heller, L. Rossetto, and H. Schuldt, "The ps-battles dataset-an image collection for image manipulation detection," *arXiv preprint arXiv:1804.04866*, 2018.

[10] E. M. Bik, A. Casadevall, and F. C. Fang, "The prevalence of inappropriate image duplication in biomedical research publications," *MBio*, vol. 7, no. 3, pp. e00809–16, 2016.

[11] E. M. Bucci, "Automatic detection of image manipulations in the biomedical literature," *Cell death & disease*, vol. 9, no. 3, pp. 1–9, 2018.

[12] D. E. Acuna, P. S. Brookes, and K. P. Kording, "Bioscience-scale automated detection of figure element reuse," *BioRxiv*, p. 269415, 2018.

[13] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[14] D. Tralic, S. Grgic, and B. Zovko-Cihlar, "Video frame copy-move forgery detection based on cellular automata and local binary patterns," in *2014 X International Symposium on Telecommunications (BIHTEL)*, pp. 1–4, IEEE, 2014.

[15] K. Sudhakar, V. Sandeep, and S. Kulkarni, "Speeding-up sift based copy move forgery detection using level set approach," in *2014 International Conference on Advances in Electronics Computers and Communications*, pp. 1–6, IEEE, 2014.

[16] K. Sudhakar, V. Sandeep, and S. Kulkarni, "Shape based copy move forgery detection using level set approach," in *2014 Fifth International Conference on Signal and Image Processing*, pp. 213–217, IEEE, 2014.

[17] D. Cozzolino, G. Poggi, and L. Verdoliva, "Copy-move forgery detection based on patchmatch," in *2014 IEEE international conference on image processing (ICIP)*, pp. 5312–5316, IEEE, 2014.

[18] S.-Y. Liao and T.-Q. Huang, "Video copy-move forgery detection and localization based on tamura texture features," in *2013 6th international congress on image and signal processing (CISP)*, vol. 2, pp. 864–868, IEEE, 2013.

[19] A. Kashyap, B. Suresh, M. Agrawal, H. Gupta, and S. D. Joshi, "Detection of splicing forgery using wavelet decomposition," in *International Conference on Computing, Communication & Automation*, pp. 843–848, IEEE, 2015.

[20] L. Zhong and W. Xu, "A robust image copy-move forgery detection based on mixed moments," in *2013 IEEE 4th International Conference on Software Engineering and Service Science*, pp. 381–384, IEEE, 2013.

[21] A. Kashyap, R. S. Parmar, M. Agrawal, and H. Gupta, "An evaluation of digital image forgery detection approaches," *arXiv preprint arXiv:1703.09968*, 2017.

[22] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

[23] C. G. Sri, S. Bano, T. Deepika, N. Kola, and Y. L. Pranathi, "Deep neural networks based error level analysis for lossless image compression based forgery detection," in *2021 International Conference on Intelligent Technologies (CONIT)*, pp. 1–8, IEEE, 2021.

[24] C. Han, B. Sun, R. Ramamoorthi, and E. Grinspun, "Frequency domain normal map filtering," in *ACM SIGGRAPH 2007 papers*, pp. 28–es, 2007.

[25] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *arXiv preprint arXiv:2003.04297*, 2020.

[26] 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*, pp. 740–755, Springer, 2014.

[27] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.

[28] S. Shao, Z. Zhao, B. Li, T. Xiao, G. Yu, X. Zhang, and J. Sun, "Crowdhuman: A benchmark for detecting human in a crowd," *arXiv preprint arXiv:1805.00123*, 2018.

[29] P. Wspanialy, "pycococreator," May 2018.

[30] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, pp. 91–99, 2015.

[31] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125, 2017.

[32] Z. Jin-Yu, C. Yan, and H. Xian-Xiang, "Edge detection of images based on improved sobel operator and genetic algorithms," in *2009 International Conference on Image Analysis and Signal Processing*, pp. 31–35, IEEE, 2009.

[33] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[34] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*, pp. 1597–1607, PMLR, 2020.

[35] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," *arXiv preprint arXiv:1911.05722*, 2019.

[36] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2." `https://github.com/facebookresearch/detectron2`, 2019.

Appendix A: Experiment Record

Methods		Bbox			Segmentation		
		AP	AP50	AP75	AP	AP50	AP75
10000 Iter	Person	2.753	10.333	0.338	2.192	7.247	1.203
	People	2.675	10.126	0.633	2.349	7.709	1.166
	Car	6.477	22.989	1.410	9.090	24.279	5.069
	Cat	5.747	20.780	0.811	5.797	18.815	1.703
	Chair	5.620	19.519	1.174	5.113	15.062	2.219
	Average	4.654	16.749	0.873	4.908	14.622	2.272
20000 Iter	Person	5.350	17.167	1.919	6.027	15.365	3.807
	People	4.132	14.062	1.157	5.088	13.259	3.155
	Car	13.930	39.228	2.944	23.604	43.745	22.572
	Cat	13.433	38.380	5.033	18.066	38.552	14.928
	Chair	11.638	34.694	4.190	15.556	32.861	13.062
	Average	9.697	28.706	3.049	13.668	28.756	11.505
30000 Iter	Person	9.296	27.409	3.540	11.065	24.930	8.079
	People	6.887	22.043	1.690	8.607	20.980	5.362
	Car	22.294	52.749	13.341	33.344	55.517	34.380
	Cat	20.217	50.138	10.838	25.718	49.416	23.750
	Chair	17.320	46.270	7.898	22.420	43.483	21.108
	Average	15.203	39.722	7.461	20.231	38.865	18.536
50000 Iter	Person	20.801	51.419	11.635	25.803	49.688	23.514
	People	14.282	39.780	6.024	19.767	40.894	17.106
	Car	29.474	62.868	24.487	46.722	68.230	50.404
	Cat	28.925	62.955	22.289	41.217	65.470	44.274
	Chair	27.300	60.609	20.345	35.791	59.024	36.719
	Average	24.156	55.526	16.956	33.860	56.661	34.403
100000 Iter	Person	39.656	74.468	38.259	45.520	73.546	48.827
	People	31.369	66.033	26.392	39.294	65.237	38.296
	Car	51.310	82.456	58.368	67.073	84.980	74.407
	Cat	49.721	82.511	54.765	62.359	83.249	70.534
	Chair	46.153	79.645	48.595	54.095	78.120	60.344
	Average	43.642	77.023	45.276	53.668	77.026	58.482

Table 5: *Training record for Basic Mask R-CNN throughout 100,000 iterations.*

Methods		Bbox			Segmentation		
		AP	AP50	AP75	AP	AP50	AP75
10000 Iter	Person	5.302	17.512	1.483	5.763	15.125	3.268
	People	3.637	12.177	1.099	4.179	10.989	2.303
	Car	13.781	38.950	5.115	23.043	42.974	22.132
	Cat	10.397	31.499	2.950	14.138	31.507	11.149
	Chair	10.705	33.189	3.278	14.653	31.642	11.779
	Average	8.764	26.665	2.785	12.355	26.447	10.126
20000 Iter	Person	11.860	33.608	5.064	14.681	32.636	10.894
	People	9.916	28.505	3.936	13.071	28.108	10.548
	Car	23.904	57.107	14.539	38.503	61.289	40.456
	Cat	18.944	47.512	10.656	26.575	48.221	25.789
	Chair	18.835	48.821	10.175	25.994	47.622	25.374
	Average	16.692	43.111	8.874	23.765	43.575	22.612
30000 Iter	Person	18.209	46.628	9.517	22.277	45.038	19.225
	People	16.363	43.074	7.919	21.398	42.132	19.223
	Car	28.406	63.702	20.456	43.940	67.442	47.774
	Cat	24.452	58.112	14.890	34.411	58.705	35.631
	Chair	24.669	57.655	16.572	31.146	54.751	31.827
	Average	22.420	53.834	13.871	30.634	53.614	30.736
50000 Iter	Person	27.075	60.588	19.748	32.819	58.746	32.257
	People	27.906	63.658	18.876	35.792	62.637	36.561
	Car	40.751	76.220	40.572	57.400	79.546	63.665
	Cat	36.297	73.308	32.102	50.028	74.287	56.334
	Chair	35.209	71.363	30.152	45.193	70.587	49.273
	Average	33.448	69.027	28.290	44.246	69.161	47.618
100000 Iter	Person	45.568	80.752	45.697	52.040	79.518	57.544
	People	46.834	82.934	47.638	54.595	82.536	61.022
	Car	54.611	84.984	63.864	71.111	87.548	77.709
	Cat	53.313	85.775	59.877	67.296	86.881	75.842
	Chair	52.103	84.360	58.263	60.127	83.450	67.009
	Average	50.486	83.761	55.068	61.034	83.987	67.825

Table 6: *Training record for IAM-Adapted Mask R-CNN throughout 100,000 iterations.*

Methods		Bbox			Segmentation		
		AP	AP50	AP75	AP	AP50	AP75
10000 Iter	Person	16.749	41.428	9.882	19.013	38.722	16.847
	People	14.295	36.789	9.186	17.368	35.995	15.016
	Car	29.220	62.431	23.287	45.136	65.990	50.502
	Cat	27.001	58.595	19.454	35.712	59.775	38.459
	Chair	25.228	57.034	18.016	32.028	55.108	32.589
	Average	22.499	51.255	15.965	29.851	51.118	30.683
20000 Iter	Person	37.378	74.071	31.685	44.335	73.027	47.579
	People	35.468	72.420	29.521	43.049	71.787	45.441
	Car	49.541	82.052	55.099	64.049	84.228	71.699
	Cat	45.965	80.519	47.793	56.843	80.271	65.159
	Chair	45.086	79.690	46.300	52.447	78.546	58.179
	Average	42.688	77.750	42.080	52.145	77.572	57.611
30000 Iter	Person	48.515	85.283	50.341	56.602	85.661	64.261
	People	41.582	77.261	40.313	50.082	78.241	55.242
	Car	51.504	83.063	57.748	69.767	87.244	76.428
	Cat	52.164	85.781	57.780	66.639	87.241	75.228
	Chair	52.732	86.811	58.798	61.544	87.132	69.303
	Average	49.299	83.640	52.996	60.927	85.104	68.092
50000 Iter	Person	58.681	88.781	67.156	61.733	87.999	71.401
	People	58.319	87.688	66.654	60.810	86.887	70.685
	Car	68.114	90.465	79.554	76.425	91.811	85.354
	Cat	66.474	91.646	77.434	73.866	91.800	84.410
	Chair	62.870	90.083	72.425	66.995	89.487	76.406
	Average	62.892	89.733	72.645	67.966	89.597	77.651
100000 Iter	Person	72.560	95.873	83.899	74.594	95.329	86.435
	People	71.808	95.179	82.866	74.211	94.468	85.900
	Car	78.954	95.223	89.315	87.576	95.772	93.185
	Cat	77.490	95.569	87.953	84.356	96.272	92.328
	Chair	75.806	95.613	86.758	78.839	95.058	88.357
	Average	75.324	95.491	86.158	79.915	95.380	89.241
200000 Iter	Person	78.323	97.138	88.941	79.678	97.207	91.536
	People	77.809	97.138	89.139	79.282	96.671	90.938
	Car	82.524	97.206	92.039	91.083	97.652	95.397
	Cat	81.838	97.229	91.849	88.328	97.395	95.074
	Chair	80.394	96.955	90.349	83.317	96.901	92.227
	Average	80.178	97.133	90.463	84.338	97.165	93.034
300000 Iter	Person	81.232	97.753	91.413	82.698	97.734	92.831
	People	80.989	97.652	91.594	82.394	97.575	93.078
	Car	82.492	96.931	91.269	92.357	97.756	96.518
	Cat	84.017	97.323	92.906	90.820	98.081	96.166
	Chair	82.129	97.468	91.333	85.217	97.412	93.483
	Average	82.172	97.425	91.703	86.697	97.712	94.415
400000 Iter	Person	84.174	98.301	92.989	83.769	98.317	94.277
	People	82.923	97.998	93.022	82.644	97.376	93.183
	Car	86.958	98.118	92.970	93.188	98.404	97.228
	Cat	87.658	97.798	94.332	91.396	98.569	96.735
	Chair	85.813	98.178	93.024	85.774	97.504	93.926
	Average	85.505	98.079	93.267	87.354	98.034	95.070
500000 Iter	Person	87.197	98.565	94.490	85.379	98.654	95.614
	People	85.913	98.408	94.217	84.542	97.653	94.431
	Car	88.650	98.406	94.305	94.550	98.626	98.275
	Cat	89.848	98.769	95.625	92.930	98.779	97.772
	Chair	88.840	98.442	94.452	87.909	97.662	95.442
	Average	88.090	98.518	94.618	89.062	98.275	96.307

Table 7: *Training record for MoCo-v2+IAM-Adapted Mask R-CNN throughout 500,000 iterations.*

Appendix B: F-SegNet Prediction Outcome

Figure 16: *Demonstration of F-SegNet prediction outcome on spliced images.*

Acknowledgement

In recent years, more and more fake pictures have been uploaded to social media platforms. Our parents and grandparents who are exposed to these new technologies may not have a very good understanding on how to identify fake images. As a result, they are vulnerable to deception and frauds. In order to prevent this from happening, we want to design an image verification assistance that can help our society identify fake images, so that our online community can become a better place for people to share and learn.

During the research of F-SegNet, the training device and GPU platform are supported by Mars Laboratory in Whittle School. We, Hao Yuan and Baoqi Zhang, collaborated and finished this research paper together by ourselves. Special thanks are also given to our schoolmates and teachers who inspired us and offered us help during this project.

We first want to express our sincere thanks to Mr.Li Zheng and Mr.Meng Xin, our project instructors from Mars Laboratory of Whittle School & Studios. Together, they taught us research ideologies and tutored us through all the challenges we encountered during our project. Our knowledge was greatly expanded under their kind guidance.

We would also like to thank Sophia Li, Geo Park, Haoran Tang and Doris Jiao, our schoolmates who helped us during this research project. Sophia Li is our junior schoolmate who helped us improve our graphical assets. With her efforts and time, we are able to produce quality images and figures. Geo park is one of our classmates who provided suggestions on improving our writing level. Haoran Tang is one of our friends who helped us in filming our experiment procedure. Doris Jiao is also one of our classmates. She took a group photo for our research team and it is later shown in the form of Fig.1 in our paper. At the same time, Doris also helped us overcome challenges by supporting us with companionship and alleviating our pressure.

Moreover, we want to express our thanks to GitHub contributors, Stack Overflow users, and all the other people on the internet who kindly helped us solve our problems. These online platforms hugely helped us improve our skills and we completed our research successfully with their generous assistance.

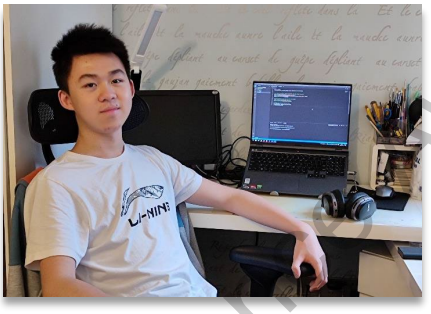
At last, we want to express our gratitude to all the people who helped us during our research once again!

袁昊　HAO YUAN

Whittle School & Studios Shenzhen Campus

+86 19924538239　pyuan@whittleschool.org

For the past several years, I devoted myself in learning computer hardware and scripting programs. **C#**, **Python**, and **JavaScript** are all my favorite languages that I used in **game development** and **webapp production**. Recently, I built a web server by myself and initiated **"Whittle Share"** project that is aimed to **provide internet file-sharing platform for our school**.



HONORS & ACEDEMICS:

TOEFL iBT score: 116/120 — *Mar 2021*

Won Waterloo math contest, Grade 11 Hypatia — *Apr 2021*

School Volleyball team member — *Sep 2021*

SELECTED PUBLICATIONS:

"An Efficient Attention Based Image Adversarial Attack Algorithm with Differential Evolution on Realistic High-Resolution Image"

ACIS 2021 FALL

Hao Yuan, Shaofei Li, Wanzhen Sun, Zheng Li, Xin Steven

ACTIVITIES & PROJECTS:

Independently programed two computer games, played by 40,000+ players from 130+ countries, receiving "Very Positive" reviews — *Nov 2019 ~ Jan 2021*

Frontend JavaScript developed of "Jupinpin" webapp — *May 2021*

Co-leader of Computer Science club — *Sep 2020 ~ current*

Initiator of "Whittle Share" web sharing service — *July 2021 ~ Current*

ABILITIES & STRENGTHS

Game programming with C# and Unity Engine

AI research and software development with Python

Frontend development with JavaScript React

MySQL database scripting

张宝瑾　BAOQI ZHANG

Whittle School & Studios Shenzhen Campus

+86 15999591278　bzhan871@whittleschool.org

After several years of study in **computer science,** I have established my ambition to develop in this direction. At the same time, I invented the **automatic positioning wheelchair,** which uses ROS function and infrared sensor to get rid of the inconvenience of the traditional wheelchair. This design **won the first prize in the competition.** In addition, I have a unique interest in designing games. That I have made an RPG type computer game.

HONORS & ACEDEMICS:

Gold medal of the 10th China Singapore International Music Competition *2016*

Second prize of 2019 Vienna World Orchestra competition *2019*

Junior software programming test system (Python) level 1 *Jul 2020*

First prize in Novelty Originality Creativity Finals, interactive programming, junior high school group *Dec 2020*

First prize in The Nineteenth Guangdong children's invention award Shenzhen trial *May 2020*

ACTIVITIES & PROJECTS:

Design article age grading function and display *Apr 2020*

Design RPG type computer game with RPG Maker MV *Jun 2020*

Design foot power generator *Dec 2020*

Design a new type of wheelchair with automatic positioning *Apr 2021*

Co-leader of Computer Science club *Sep 2021 ~ Current*

ABILITIES & STRENGTHS

Programming with Python & C

Proficient in game compilation software

Proficient in playing 6 kinds of instrument

Received all A and A* during Junior high school and senior high school English and computer science classes