GCA-Net: Utilizing Gated Context Attention for Improving Image Forgery Localization and Detection

Sowmen Das†  Md. Saiful Islam¶  Md. Ruhul Amin§

†Shahjalal University of Science and Technology, Bangladesh  ¶University of Alberta, Canada  §Fordham University, USA

1sowmendipta@gmail.com  2mdsaifu1@ualberta.ca  3mamin17@fordham.edu

Abstract

Forensic analysis depends on the identification of hidden traces from manipulated images. Traditional neural networks fail in this task because of their inability in handling feature attenuation and reliance on the dominant spatial features. In this work we propose a novel Gated Context Attention Network (GCA-Net) that utilizes the non-local attention block for global context learning. Additionally, we utilize a gated attention mechanism in conjunction with a dense decoder network to direct the flow of relevant features during the decoding phase, allowing for precise localization. The proposed attention framework allows the network to focus on relevant regions by filtering the coarse features. Furthermore, by utilizing multi-scale feature fusion and efficient learning strategies, GCA-Net can better handle the scale variation of manipulated regions. We show that our method outperforms state-of-the-art networks by an average of 4.2% – 5.4% AUC on multiple benchmark datasets. Lastly, we also conduct extensive ablation experiments to demonstrate the method’s robustness for image forensics.

1. Introduction

Image manipulation is the act of altering an image’s content using various processing techniques. Recent advancements in this field have spurred many applications, e.g., content removal [29], face-swapping [27], attribute changing [46], etc. Splicing, copy-move, and inpainting are the three primary image forgery techniques [7]. In the majority of manipulation scenarios, a portion of an image (source) is copied and pasted onto another image (target) in order to alter the target image’s content, as shown in. Fig. 1. Forgy detection algorithms are used in order to determine if an image is authentic or not. This is a critical task in the age of mass social media, because any fake news can spread rapidly and be used to foment panic and propaganda.

Numerous methods have been proposed over the years for image forgery localization and detection (IFLD). The use of neural networks for this purpose has gained popularity [3] in recent years. Deep CNN architectures are capable of learning intrinsic features from images, allowing them to detect a variety of artifacts. But, unlike traditional image classification, forgery detection needs to identify hidden manipulation traces from the image rather than the apparent spatial content. Existing works largely use sequential frameworks composing of a linear encoder and a classifier for this purpose [62, 47, 20]. However, because forgery artifacts are so subtle, these linear style networks suffer from feature attenuation. Although some methods try to utilize feature hierarchies through recursive pooling [25, 2], they are mostly bound to local neighborhoods.

Furthermore, existing localization techniques primarily focus on categorizing individual pixels in order to generate a pixel-level localization map. They regard forgery localization as a binary classification of each individual pixel. Image-level classification is done depending on the percentage of detected manipulated pixels. As a result, they overlook area-specific characteristics such as noisy edges, object artifacts, and region dissimilarities. Since authentic images do not contain any manipulation traces, they highlight erroneous pixels when trying to localize untampered images; shown in Fig. 2. As a result, these methods incorrectly label real images as fake.

Contributions: In order to address the problems faced by existing networks, we propose our Gated Context Atten-
Our proposed solution intends to increase the performance of media forgery detection by combining techniques that are often applied in unrelated domains. In doing so, we are able to reduce false positives greatly and thereby improve the state-of-the-art performance for manipulation detection. To summarize the contributions of our work:

- We propose GCA-Net consisting of a novel Gated Context Attention module that enables efficient modeling of long-range dependencies and improves global context representation necessary for manipulation localization.
- We substantiate the impact of global context and non-locality in detecting image forgeries, facilitating future research in this field.
- We illustrate the effects of training strategies for reducing false positives and improving image-level predictions.
- We show that our method outperforms the state-of-the-art performance on standard datasets for image manipulation detection and localization.

2. Related Works

Depending on the type of forgery, different clues and artifacts can be found in an image to determine whether it is authentic or not. These include, compression noise [54, 1], PRNU sensor information [14, 13], camera model information [8, 41], local noise features [49, 16], etc. Earlier forensic methods used algorithms to detect these noise and signal properties using carefully designed filters. Among them, the SRM filters used in [68] have shown success in identifying hidden noise patterns from images.

Lately, many neural network architectures have been proposed for both independent forgery detection [32, 43, 37, 52], and localization [38, 9, 8, 61]. Object detection networks using RCNN, region proposal modules, and bounding box identification have shown to be effective for manipulation localization [68, 5, 57]. Consequently, segmentation networks like Unet and DeepLab have also been used in [6, 11]. Several studies have been conducted on improving input representation using constrained layers [4, 65]. These methods optimize the initial layers to minimize the effects of dominant spatial features. ManTra-Net [62] uses a simple VGG [48] network with a Z pooling method to localize anomalous features. This work was extended by SPAN [25] to further model the spatial correlation via local self-attention blocks and hierarchical pyramid propagation. However, both networks fail to utilize the correlation of global context and multi-scale features. In a recent work [33], PSCC-Net was proposed that tries to address these problems using channel features. They showed that multi-scale features and attention could be leveraged to improve manipulation detection.

While self-attention was originally introduced for language modeling [53], this seminal work has been proven to improve long-range feature representation across a vari-
3. Proposed Method

3.1. Overview

As illustrated in Fig. 3, GCA-Net is a multi-branch dense encoder-decoder network. The model is comprised of three parts: a feature encoder, a dense decoder, and two separate heads for classification and localization. For the encoder backbone, we use the EfficientNet [51] architecture. An input image is first passed through a series of content suppression layers, the outputs of which are then concatenated and sent to the backbone. The dense decoder takes in both the encoded feature generated by the backbone as well as intermediate layer outputs. The features are propagated through the decoder blocks to generate a localization map. Each decoder block is composed of a GCA layer followed by a series of convolutional layers for the purpose of accumulating and upscaling the encoded features. Additionally, the encoded features also pass through the classification head to generate an image-level probability score. Unlike traditional networks, which focus exclusively on localization, we simultaneously classify the images using a separate branch which helps the decoder in making more accurate predictions. The entire network is trained end-to-end. This joint multi-task learning approach significantly improves the overall performance of the network.

3.2. Content Suppression

While existing CNN architectures are capable of extracting features from images, they are biased towards the dominant spatial features rather than the manipulation traces. To improve detection, we add additional modules to the encoder’s first layer that extract noise level features and suppress the spatial content. For this purpose, we introduced four modules – i) the SRMConv2D [68] layer for steganalysis feature extraction, ii) the BayarConv2D [4] layer for constraining the initial values, iii) the classic convolution layer termed as RGBConv2D, and iv) our proposed Error Level Analysis (ELA) Module for extracting compression artifacts. Additional details regarding the implementation of ELA, and the effect of each layer on the network’s performance can be found in Suppl. A1.
3.3. Dense Feature Decoder

The decoder’s role is to estimate the localization mask from the encoded features. We use U-Net++ [69] which employs a bottom-up architecture. We chose this framework for two reasons: its dense interconnection between layers and its multi-scale learning capability.

Firstly, we aim to mitigate the effects of feature attenuation on subtle manipulation traces in order to improve gradient flow. The skip connections between the coarse and fine layers of the network allow the flow of global features, minimizing feature loss. This is analogous to how residual connections improve gradient flow in multi-layered deep networks. Additionally, this dense feature pooling enables us to make use of both global and local features at each stage of the decoding process. The dense connections bring the semantic level of the encoder feature maps closer to that of the decoder, facilitating optimization and faster convergence.

Secondly, since the decoder samples the intermediate encoder layers at multiple scales, it improves global representations. Features from an input $I \in \mathbb{R}^{3 \times H \times W}$ are sampled at five intermediate scales $H/k \times W/k$, where $k \in \{s, s^2, s^3, s^4, s^5\}$. The intermediate feature scales for an image $I \in \mathbb{R}^{3 \times 256 \times 256}$ and $s = 2$ are, $128 \times 128, 64 \times 64, 32 \times 32, 16 \times 16$, and $8 \times 8$. At each node, features from all lower scales are accumulated. Thus, each node can determine which features are most relevant reinforced by the finer lower level features. In contrast to standard encoder-decoder networks, this multi-scale fusion enables the decoder network to easily identify the essential attributes without relying solely on the previous layer.

Let $y^{i,j}$ denote the output of a decoder node $X^{i,j}$, where $i$ indexes the sequential down-sampling layers of the encoder and $j$ indexes a decoder block along the skip pathway at the $i^{th}$ layer. The output $y^{i,j}$ at any node is computed as,

$$y^{i,j} = C(\vartheta)$$

where $C(\cdot)$ is a series of convolution operations followed by a ReLU activation, $\Theta(\cdot)$ is the GCA layer, $\Omega(\cdot)$ is a non-local context block, $\mathcal{U}(\cdot)$ denotes an up-sampling layer, and $[\cdot]$ denotes the concatenation layer.

3.4. Gated Context Attention (GCA)

Attention mechanisms are used to modulate learned features according to their relative significance. The GCA operation shown in Fig. 4 is divided into two stages: 1) Global Context Pooling and 2) Attention Gating.

Deep convolution stacks tend to obfuscate global pixel-to-pixel relationships due to their nature of locality [55]. Non-local blocks attempt to solve this problem using attention weights and aggregating information from other points to reinforce the features of a query position. Global context modeling [10] is an improved attention framework that identifies the long range dependencies between feature maps. In the first stage, we compute the global context from the concatenated features of the same level. These are coarse, large-scale feature maps that contain a greater amount of global information than the subsequent layers. Because identification of manipulation features is based on detecting changes between a group of pixels and their surroundings, using these global contexts can help the model recognize the differences between altered regions. We can rewrite Eq. (2) as,

$$\vartheta = \Theta \left(\Omega (F_i), F_g\right) \quad (3)$$

where, $F_i = \left[y^{i,k}\right]_{k=0}^{j-1}$ is the concatenation of all the $i^{th}$ level features, and $F_g = \mathcal{U} \left(y^{i+1,j-1}\right)$ is the up-sampled feature of $(i+1)^{th}$ level. There are three steps in the context modeling framework $\Omega$ [10] — i) attention pooling, ii) feature transform, and iii) feature fusion. For the pooling step we take the layer features $F_i \in \mathbb{R}^{C_i \times H \times W}$ and pass it through a $1 \times 1$ convolution to reduce the channels to $C_i \times 1 \times 1$. It groups the features of all positions via weighted averaging to obtain the global context features. This is similar to the Global Average Pooling of Squeeze-Excitation (SE) layer [23]. The pooled features are then passed through a bottleneck block to capture the channel-wise dependencies. This is the transform step. The features are reduced and expanded by a factor $r$ to compute the importance of each channel, analogous to the excitation operation of SE block. The layer normalization acts as a regularizer to benefit generalization. Finally, the fusion step aggregates the context features to the features of each input position using a broadcast element-wise addition.

The second stage of GCA is the gated attention path. Gating is used to filtering the coarse level feature maps of the upper layers. Attention Gates (AG) identify salient image regions and prune feature responses to retain only relevant activations. As a result, feature responses in irrelevant background regions are gradually suppressed. AGs generate a coefficient matrix, $A_g \in [0, 1]$, which is multiplied with the input features to alter the scale of feature responses. Each AG learns to focus on a subset of target structures reinforced by the downstream features. The coarse maps of the encoder represent global relationships, while the downstream layers identify finer discriminating features. Gating uses these finer embeddings to disambiguate irrelevant and noisy responses within the coarse features. In order to obtain the gating coefficient via additive attention, we first transform the generated context feature $\Omega \left(F_i\right) \in \mathbb{R}^{C_i \times H \times W}$ and gating feature $F_g \in \mathbb{R}^{C_g \times H \times W}$ to an intermediate vector $F_{int} \in \mathbb{R}^{C_{int} \times H \times W}$ using $1 \times 1$ convolutions. A non-linear transformation layer $W_c = ReLU(Conv(\cdot))$ is then used to resample the in-
Figure 4: The structure of Gated Context Attention block. The blue line shows the flow concatenated features $F_l$ of $i^{th}$ decoder level and the orange line denotes the upsampled gating feature $F_g$ from $(i+1)^{th}$ level as referred in Fig. 3.

intermediate vector. This produces the final attention matrix $A_g = W_\zeta(\Omega(F_l) \oplus F_G) \in \mathbb{R}^{1 \times H \times W}$. Finally, the attention matrix is multiplied with the input feature stack $F_l$ to generate $\vartheta \in \mathbb{R}^{G \times H \times W}$, the GCA output. The resulting gated features are sent through a series of convolution and activation layers $C(\cdot)$ that perform the decoding operation at the particular node. Additional spatial and channel attention layers [44] can be added after this stage to further modulate the learnt features.

4. Experiments

4.1. Datasets

We follow the evaluation protocols in [62, 68] for training and validation. We train our model on four types of data — splicing, copy-move, inpainting, and authentic images. We use the Dresden [21] and MS COCO [31] generated synthetic datasets from [2], as well as the Defacto dataset [35], for splicing and copy-move. We also use Defacto for inpainting data. We use the IMD-Real dataset [39] for unaltered images. For both pre-trained and finetuned evaluation we use the four standard datasets: CASIAv2 [18], NIST16 [22], COVERAGE [58] and IMD-2020 [39], following the training/testing split described in [33].

In total, we used ~170k images for training, with roughly equal class distribution. This amount of data is magnitudes lower compared to existing SOTA methods like SPAN [25], MantraNet [62], and PSCC-Net [33] that use upwards of 500k~1M images. We did not use a larger dataset due to resource and accessibility constraints. The majority of our experiments and training were conducted on an Nvidia 1080 Ti GPU. However, despite the smaller train set, our model outperforms these methods in multiple experiments.

4.2. Loss Function

Our dataset consists of input images $I \in \mathbb{R}^{3 \times H \times W}$ and binary ground-truth masks $M \in [0, 1]^{1 \times H \times W}$. To train GCA-Net, we used a multi-task loss function combining detection and localization losses. Loss function plays a significant role in directing the model’s learning capabilities. Most traditional methods [62, 25, 33] train the localization network using Binary Cross-Entropy (BCE) loss. BCE works well for classification problems with lots of data and a balanced dataset because it weighs all predictions equally. The traditional BCE loss follows the equation,

$$L_{BCE} = -\frac{1}{N} \sum_{i=0}^{N} y_i \cdot \log(\hat{y}_i) \quad (4)$$

The loss function minimizes the distance between the predicted and ground truth value for all predictions. For example, if a fake pixel is predicted with a probability of 0.7, BCE will try to bring it closer to 1 in order to minimize the loss. Thus, for unbalanced datasets, predictions become skewed toward a particular class. Since we are working with a small and unbalanced dataset, we opt to use a combination of Dice [50], and Focal [30] loss. We compute the dice loss using the following equation,

$$L_{DSC}(P, G) = -\log \left( \frac{2 \cdot |P \cap G| + \epsilon}{|P| + |G| + \epsilon} \right) \quad (5)$$

For a prediction mask $P$ and ground-truth $G$ both having dimensions $1 \times H \times W$, the numerator calculates the intersection of the regions, and the denominator measures the union. Dice loss minimizes the distance between the predicted and true regions. Rather than calculating loss for each individual pixel, it improves prediction for the entire region. This is critical in lowering our false positive rate. Since BCE tries to minimize loss for each pixel independently, the network always produces incorrect predictions for a certain number of pixels.

Even though Dice loss improves false positive and region overlaps, it falls short for small forged regions. This
AUC  F1  85.60  88.51  53.46  86.92  76.00  61.56  72.39  87.74  67.72  48.91  85.2  68.4  85.2  68.4  85.2  68.4  85.2  68.4  87.1  70.5  87.9  71.7  86.3  69.4  88.3  72.6  88.3  72.6  88.9  72.5  87.3  70.4  88.1  72.0  88.3  72.6  88.3  72.6

Table 1: Localization performance of GCA-Net under different loss functions on CASIAv2 validation set.

is because as the regions become very small, \(|G|\) becomes small. So if the model does not predict anything at all, \(t.e.,\) if \(|P| \to 0\), then \(|P \cap G| \to 0\), and the total loss decreases. Thus, dice loss alone is ineffective in these instances. To overcome this problem, we combine Focal loss with Dice loss. Focal loss is an improvement over BCE loss with an additional temperature parameter to account for overconfident predictions. Focal loss is calculated as,

\[
L_{FL}(P) = -\alpha \cdot (1 - P)^\gamma \log(P) \tag{6}
\]

The temperature \(\gamma\) controls the weighting of each prediction. When \(\gamma > 0\) is used, weak predictions are weighted more heavily. So, the network focuses on improving a prediction of 0.3 towards 1 rather than trying to improve a prediction of 0.7. Since we are trying to correctly classify fake pixels, a larger prediction \(P\) results in a smaller \((1 - P)^\gamma\), thereby reducing the overall loss. So, the network will try to classify weaker predictions more accurately. Combining all of the above, our final loss function becomes,

\[
L = w_c \cdot L_{CLS} + w_d \cdot L_{DSC} + w_f \cdot L_{FL} \tag{7}
\]

\(L_{CLS} = L_{BCE}\) is the classification loss for the encoder’s prediction. By combining classification loss, the network can optimize both the encoder and decoder at the same time. Each loss parameter can be adjusted independently. We use \(\epsilon = 10^{-7}, \gamma = 2, w_c = 1, w_d = 1.10, \) and \(w_f = 1.15\) to train our final network. In Table 1 we report the results of different configuration of loss functions on the CASIAv2 validation set. To quantify localization performance, we use pixel-level AUC following previous works \([62, 68, 33]\) and Dice score \(t.e.,\) pixel F1 score.

4.3. Ablation Study

The ablation study is done on the CASIAv2 validation set, and we report the pixel-level AUC and F1 score for localization. We follow the ablation study following \([10]\) to compare change of parameters in the attention module.

**Block Design:** For the choice of long range context modeling we compare other existing frameworks - Non-local (NL) block, Simplified Non-local (SNL) block, SE block against the Global Context block, placed before every decoder node. Table 2(a) shows all context frameworks achieve better performance over baseline. NL and SNL blocks are quite similar, while GC blocks with comparably fewer parameters yield the best performance.

**Bottleneck Design:** The effects of each component in the bottleneck section are shown in Table 2(b). w/o ratio uses a single \(1 \times 1\) convolution as a transform which has higher parameters and achieves the best performance. Even though r16+ReLU has fewer parameters they are harder to optimize. Thus Layer Norm (LN) is used to ease optimization, leading to performance similar to w/o ratio.

**Bottleneck Ratio:** The bottleneck is used to reduce redundancy in parameters and provide a trade-off between parameter and performance. The bottleneck ratio \(r\) controls the amount of feature compression. Table 2(c) shows that the network’s performance improves consistently as the ratio decreases. We use a bottleneck ratio \(r = 4\) which has a good balance of performance and parameters.

**Pooling and Fusion:** The different choices for pooling and fusion are shown in Table 2(d). The results follow the similar trend as explained in \([10]\). It shows that addition is more
effective than scaling in fusion stages. Best performance comes from attention pooling combined with addition. This indicates that global context aggregation depends on how features from all positions are grouped together.

**Block Positions:** We determine whether the placement of the GCA blocks have an effect on performance. Various placement positions are illustrated in Suppl. A2. As seen in Table 2(e), placements effect the network’s performance only to a small degree. The best results are obtained by placing the attention block before each node.

Additional ablation experiments regarding backbone choice, and training specifics are provided in Suppl. A3.

### 5. Comparison and Evaluation

We evaluate the test performance against existing SOTA architectures against both pre-trained and finetuned GCA-Net. The pre-trained model was selected based on the best validation score on the training set. We report existing values as mentioned in PSCC-Net [33]. For finetuned evaluation we use unseen test splits generated following the evaluation process in [68].

| Method       | CASIAv2 | COVERAGE | NIST16 | IMD2020 |
|--------------|---------|----------|--------|---------|
| MantraNet [62] | 81.7   | 81.9     | 79.5   | 74.8    |
| SPAN [25]     | 79.7   | 92.2     | 84.0   | 75.0    |
| PSCC-Net [33] | 82.9   | 84.7     | 85.5   | 80.6    |
| GCA-Net       | 87.1   | 83.1     | 85.2   | 81.3    |

Table 3: Comparison of localization AUC against existing methods using their pre-trained models.

In Tables 3, 4 we compare GCA-Net to existing methods. Both the pre-trained and fine-tuned comparisons demonstrate that GCA-Net outperforms all other methods by a significant margin on the CASIAv2 and IMD-2020 datasets and is comparable on the NIST and COVERAGE datasets. On CASIA, we see an improvement of 5.4% and 4.46% on IMD over the current SOTA PSCC-Net. GCA-Net achieves an AUC of 95.3 on NIST-16, trailing behind PSCC-Net by only 4%. However, we surpass every network on NIST-16 in terms of F1 score. This is because we have finetuned the loss functions extensively. Pixel-level F1 score measures the region overlap of the prediction and ground-truth. Since we optimized our network using Dice loss, the network’s region identification is better than the existing models. We rank third in COVERAGE, behind SPAN and PSCC-Net. COVERAGE includes samples with very small shifts of copied regions followed by contrast correction and edge blurring. Our train data, which is composed entirely of publicly available datasets, is free of such perturbations, resulting in a difference in the train-test distribution. This limitation can be overcome by training on synthetic copy-move data supplemented with adversarial examples. For authentic image localization, GCA-Net outperforms all the methods with little to no false-positive predictions. In Fig. 6 we show the prediction masks for the three authentic samples we previously compared in Fig. 2. We can see that the predictions are virtually error-free with no false predictions. Qualitative examples of localization are shown in Fig. 5.

### 5.1. Detection Performance

To analyze the performance of GCA-Net at the image level, we compare it to the SOTA architectures using the metrics reported in [33]. We use a detection dataset developed by separating images from the CASIAv2 dataset for comparison. The set contains 511 forged images and 749 real images. We use the pretrained GCA-Net for this comparison. As can be seen from Table 5, GCA-Net significantly outperforms all other models. This is because the network is equipped with a dedicated classification head that was trained particularly to recognize photos with forged content. In comparison, existing approaches identify the image class by counting the number of manipulated pixels discovered in the localization result.

| Method            | Image-Level F1 Score |
|-------------------|----------------------|
| MantraNet         | 56.69                |
| SPAN              | 63.48                |
| PSCC-Net          | 66.88                |
| GCA-Net (pretrained) | **85.51**            |

Table 5: Comparison of image-level detection performance on CASIAv2 detection set against other methods.

### 5.2. Robustness Analysis

We examine the performance of our proposed method against various attacks/post-processing to further verify its efficacy and robustness. For this purpose, we degrade images from the NIST16 test set using the distortion settings in [33]. These include Gaussian Blur with kernel size k, JPEG Compression with a quality factor q, and Additive Gaussian Noise using standard deviation σ. The reported metrics are calculated using the pretrained GCA-Net. In Fig. 7, we can see that our model outperforms existing approaches against various post-processing attacks.
5.3. Limitations

In our experiments we found that GCA-Net might fail when the manipulation region is very large compared to authentic pixels. Fig. 8(a) contains a sample from NIST-16 where the entire white region is manipulated. Although GCA-Net could detect the discrepancy between the authentic and forged regions, it is not confident about the prediction. It highlighted the centre portion denoting that the region is different from the surrounding pixels. For a second example, we tested our network similar to [62] to check for manual assistance applicability. The initial image in Fig. 8(b) was of dimension $1024 \times 1520$ having a small forged region. The network failed to locate the region when tested with the entire image. After that, we cropped the image around the forged location and again tested the cropped image. This time the network was able to identify the manipulated region. This indicates that the network could be used as a computer-aided tool.

6. Conclusion

In this paper, we introduced a novel Gated Context Attention Network (GCA-Net) for detecting and localizing image forgeries. Our proposed network uses a gated attention block to utilize the global context features together with the region attributes to localize manipulated pixels. This paper illustrated the problems surrounding existing methods and how they might be addressed with better feature representation and training strategies. As demonstrated by our results, GCA-Net can improve forgery detection by utilizing multi-scale features through dense interconnections. Furthermore, we validated the effectiveness of our approach by outperforming existing SOTA architectures on multiple benchmark datasets. In the future, we will further explore our method’s ability to detect deep learning-based deepfake forgeries and other segmentation tasks.
A1. Content Suppression Modules

In order to improve detection of the subtle forensic features and suppress the spatial content of the image, we add additional modules to the encoder’s first layer that extract noise level features. For this purpose, we introduced four modules – i) the SRMConv2D [68] layer, ii) the BayarConv2D [4] layer, iii) the classic convolution layer termed as RGBConv2D, and iv) our proposed Error Level Analysis (ELA) Module. Fig. 9 shows the output of applying SRM and ELA on a tampered image.

ELA has previously been used for localizing compression artefacts from JPEG images [56]. It works by comparing the pixel-wise difference between an image and its compressed copy. If an image contains pixels from a different source, then the pixels of the two sources would produce different levels of compression noise. We propose to use this ELA output as a feature for the encoder. We take an input image and compress it with a reduced 90% compression factor. Then we calculate the difference between the original and the compressed image to generate the ELA output. This output ELA image is then passed through a series of convolution layers before applying activation to produce the ELA feature map.

To evaluate the effect of these modules on the encoder, we compare the detection accuracy on the CASIAv2 test set in Table 6. We can see that the choice of the first layer affects model performance by a significant amount. The proposed ELA module has a notable effect as it improves encoder accuracy by a factor of more than 3%. So, for our final encoder, we select a combination of the four layers. The input images pass through all of them simultaneously, then the outputs are concatenated and sent to the backbone. This additional compression and steganalysis feature help the network to detect the traces of the boundary regions. Moreover, the encoder becomes more robust to post-processing operations as it learns to detect and correlate the multi-domain artefacts with other spatial features.

| 1st Conv Layer     | #Filters, Kernel Size | Detection Accuracy (%) |
|--------------------|-----------------------|------------------------|
| RGBConv2D          | 16, k=(3,3), p=1, x2  | 87.60                  |
| SRMConv2D          | 3, k=(5,5)            | 89.13                  |
| BayarConv2D        | 3, k=(5,5)            | 88.75                  |
| ELA Module         | 32, k=(3,3), p=1, x2  | 90.47                  |
| Combined           | 54, -                 | 92.71                  |

Table 6: Results of using additional feature extraction layers for the 1st layer of the encoder with an EfficientNet-B4 backbone.

A2. Additional Ablation Experiments

A2.1 Block Positions

Previously we had talked about the effects of placing the GCA block in different positions within the network. Fig. 10 shows these placement positions. The blue squares represent the encoder layer, green circles are the decoder nodes, and the red rectangles denote the GCA block.

To evaluate the effect of these modules on the encoder, we compare the detection accuracy on the CASIAv2 test set in Table 6. We can see that the choice of the first layer affects model performance by a significant amount. The proposed ELA module has a notable effect as it improves encoder accuracy by a factor of more than 3%. So, for our final encoder, we select a combination of the four layers. The input images pass through all of them simultaneously, then the outputs are concatenated and sent to the backbone. This additional compression and steganalysis feature help the network to detect the traces of the boundary regions. Moreover, the encoder becomes more robust to post-processing operations as it learns to detect and correlate the multi-domain artefacts with other spatial features.

A2.2 Backbone Choice

There are no dominant network architectures proven to be useful for IFLD tasks. XceptionNet has been shown to
perform well for DeepFake detection, and media forgeries [45]. DenseNet also showed promise in detecting camera model features [42], which has relevant implications for manipulation identification. We test multiple such backbone networks to test their efficacy for manipulation detection. We trained and tested these baseline models using the CASIAv2 [18] dataset. Since we are evaluating the encoder performance only, we perform these tests as a classification task without the decoder and compare the detection performance. From Table 7 we see that EfficientNet performs the best. Additionally, it uses an inverse bottleneck convolution with channel attention making it the lightest of all the networks with only 19.34 million parameters.

### A.3. Implementation Details

In order to tackle the challenge of low data and improve generalizability, all images were augmented using Flipping, Random Rotations, Optical and Grid Distortions, and Gaussian Blur, each with a probability of 30% - 50%. We trained the model with the encoder pre-loaded with ImageNet weights using Adam optimizer with a learning rate of 0.00001 and a weight decay of 0.00005. Learning rate scheduling was done using Reduction on Plateau by a factor of 0.25. All models were trained for 60 epochs, with Early-Stopping patience of 20 epochs. The model was implemented using PyTorch. For the EfficientNet backbone we used the implementation from Timm PyTorch Image models [59].

### A.4. False Positive Evaluation

One of the primary contributions of our proposed method was the reduction of false positives in authentic images. In order to evaluate the degree of improvement against other networks, we calculate $R_{FPR} = -\log(FPR)$ for both authentic and manipulated images. $FPR$ or False Positive Rate quantifies the proportion of data in a sample that is incorrectly identified. In our case, a false positive prediction occurs when a pixel that should be authentic, i.e., 0, is classified as fake, i.e., 1. $FPR$ measures the fraction of incorrect predictions made against the entire set of pixels in an image.

$$FPR = \frac{\text{False Positive Pixels}}{\text{Total Number of Pixels}}$$ (8)

However, because the number of incorrect predictions is relatively small in comparison to the total number of pixels in an image, this value can become extremely small and difficult to interpret. Thus, we use the $-\log(FPR)$ as our evaluation metric. Increased values of $R_{FPR}$ indicate better performance and lower false positives.

### Table 7: Baseline test accuracy of different CNN architectures for binary classification on CASIAv2.

| Model                  | #Params (M) | Test Accuracy (%) |
|------------------------|-------------|-------------------|
| XceptionNet [15]       | 22.86       | 78.03             |
| DenseNet-161 [26]      | 28.68       | 83.56             |
| ResNeXt-50 [63]        | 30.42       | 82.29             |
| SEResNeXt-50 [24]      | 27.56       | 85.81             |
| EfficientNet-B4 [51]   | 19.34       | 87.65             |

### Table 8: Comparison of false positive, $R_{FPR}$, for authentic and manipulated images from CASIAv2 and IMD2020 test sets. Higher values denote less false positives.

|                      | ManTraNet | GCA-Net |
|----------------------|-----------|---------|
| CASIA                |           |         |
| Authentic (A)        | 5.39      | 9.88    |
| Tampered (T)         | 4.77      | 5.09    |
| Combined (A+T)       | 4.95      | 5.97    |
| IMD                  |           |         |
| Authentic (A)        | 4.57      | 6.10    |
| Tampered (T)         | 4.43      | 5.65    |
| Combined (A+T)       | 4.66      | 6.93    |
| Authentic (CASIA + IMD) | 4.94  | 8.83    |
| Tampered (CASIA + IMD)| 4.56  | 5.30    |

To conduct the evaluation, we created a test set using 200 authentic and 100 tampered images taken independently from both CASIAv2 and IMD2020 datasets. We used the pretrained model for GCA-Net, and the publicly available implementations for ManTraNet. As illustrated in Table 8, GCA-Net consistently outperforms the other models by a significant margin. For authentic images in both CASIA and IMD, GCA-Net’s score is almost double that of the other methods. Although the difference is relatively small for tampered images, GCA-Net still outperforms these existing state-of-the-art models. Thus, we can see that our proposed framework is more accurate at identifying authentic images and generates the fewest false positive predictions. Further qualitative comparison for various images are illustrated in Fig. 11.

### References

[1] Neda Alipour and Alireza Behrad. Semantic segmentation of jpeg blocks using a deep cnn for non-aligned jpeg forgery detection and localization. Multimedia Tools and Applications, 2020.

[2] Jawadul H Bappy, Cody Simons, Lakshmanan Nataraj, BS Manjunath, and Amit K Roy-Chowdhury. Hybrid Lstm and encoder-decoder architecture for detection of image forgeries. IEEE Transactions on Image Processing, 2019.

[3] Z. J. Barad and M. M. Goswami. Image forgery detection using deep learning: A survey. In 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS), pages 571–576, 2020.

[4] B. Bayar and M. C. Stamm. Constrained convolutional neural networks: A new approach towards general purpose image manipulation detection. IEEE Transactions on Information Forensics and Security, 2018.
Figure 11: Qualitative comparison of GCA-Net and ManTraNet for various tampered and authentic images.

[5] T. Aaron Gulliver & Saif alZahir Belal Ahmed. Image splicing detection using mask-rcnn. Signal, Image and Video Processing, 2020.

[6] X. Bi, Y. Wei, B. Xiao, and W. Li. Rru-net: The ringed residual u-net for image splicing forgery detection. In 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pages 30–39, 2019.

[7] Gajanan K. Birajdar and Vijay H. Mankar. Digital image forgery detection using passive techniques: A survey. Digital Investigation, 10(3):226 – 245, 2013.

[8] L. Bondi, S. Lameri, D. Güera, P. Bestagini, E. J. Delp, and S. Tubaro. Tampering detection and localization through clustering of camera-based cnn features. In 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pages 1855–1864, 2017.

[9] Jason Bunk, Jawadul H. Bappy, Tajuddin Manhar Mohammed, Lakshmanan Nataraj, Arjuna Flenner, B. S. Manjunath, Shivkumar Chandrasekaran, Amit K. Roy-Chowdhury, and Lawrence Peterson. Detection and localization of image forgeries using resampling features and deep learning, 2017.

[10] Yue Cao, Jiarui Xu, Stephen Lin, Fangyun Wei, and Han Hu. Gcnet: Non-local networks meet squeeze-excitation networks and beyond, 2019.

[11] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, 2017.

[12] Xinnru Chen, Chengbo Dong, Jiaqi Ji, Juan Cao, and Xirong Li. Image manipulation detection by multi-view multi-scale supervision, 2021.

[13] Giovanni Chierchia, Giovanni Poggi, Carlo Sansone, and Luisa Verdoliva. Prnu-based forgery detection with regularity constraints and global optimization. In 2013 IEEE 15th International Workshop on Multimedia Signal Processing (MMSP), 2013.

[14] Giovanni Chierchia, Giovanni Poggi, Carlo Sansone, and Luisa Verdoliva. A bayesian-mrf approach for prnu-based image forgery detection. IEEE Transactions on Information Forensics and Security, 9, 2014.

[15] François Chollet. Xception: Deep learning with depthwise separable convolutions, 2017.

[16] Davide Cozzolino and Luisa Verdoliva. Noiseprint: a cnn-based camera model fingerprint, 2018.

[17] Hao Dang, Feng Liu, Joel Stehouwer, Xiaoming Liu, and Anil K. Jain. On the detection of digital face manipulation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2020.

[18] Jing Dong, Wei Wang, and Tieniu Tan. Casia image tampering detection evaluation database. pages 422–426, 07 2013.

[19] P. Ferrara, T. Bianchi, A. De Rosa, and A. Piva. Image forgery localization via fine-grained analysis of cfa artifacts. IEEE Transactions on Information Forensics and Security, 7(5):1566–1577, 2012.

[20] Aurobrata Ghosh, Zheng Zhong, Terrance E Boult, and Maneesh Singh. Spliceradar: A learned method for blind image forensics, 2019.

[21] Thomas Gloe and Rainer Böhme. The 'dresden image database' for benchmarking digital image forensics. Association for Computing Machinery, 2010.

[22] H. Guan, M. Kozak, E. Robertson, Y. Lee, A. N. Yates, A. Delgado, D. Zhou, T. Kheyrykh, J. Smith, and J. Fiscus. Mfc datasets: Large-scale benchmark datasets for media forensic challenge evaluation. In 2019 IEEE Winter Applications of Computer Vision Workshops (WACVW), pages 63–72, 2019.
[23] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. Squeeze-and-excitation networks, 2019.

[24] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. Squeeze-and-excitation networks, 2019.

[25] Xuefeng Hu, Zhihan Zhang, Zhenye Jiang, Syomantak Chaudhuri, Zhenheng Yang, and Ram Nevatia. Span: Spatial pyramid attention network for image manipulation localization. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, Computer Vision – ECCV 2020. Springer International Publishing, 2020.

[26] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks, 2016.

[27] Iryna Korshunova, Wenzhe Shi, Joni Dambre, and Lucas Theis. Fast face-swap using convolutional neural networks, 2016.

[28] N. Krawetz. A picture ‘s worth . . . digital image analysis and forensics version 2. 2007.

[29] Jingyuan Li, Fengxian He, Lefei Zhang, Bo Du, and Dacheng Tao. Progressive reconstruction of visual structure for image inpainting. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), October 2019.

[30] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection, 2018.

[31] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2014.

[32] Huajian Liu, Martin Steinbach, and Katrin Schölei. Improved manipulation detection with convolutional neural network for jpeg images. pages 1–6, 08 2019.

[33] Xiaohong Liu, Yaojie Liu, Jun Chen, and Xiaoming Liu. Pscc-net: Progressive spatio-channel correlation network for image manipulation detection and localization, 2021.

[34] Babak Mahdian and Stanislav Saic. Using noise inconsistencies for blind image forensics. Image Vision Comput., 27(10), 2009.

[35] Gaël MAHFOUDI, Badr TAJINI, Florent RETRAINT, Frédéric MORAIN-NICOLIER, Jean Luc DUGELAY, and Marc PIC. Defacto: Image and face manipulation dataset. In 2019 27th European Signal Processing Conference (EUSIPCO), 2019.

[36] Gas ¨el MAHFOUDI, Badr TAJINI, Florent RETRAINT, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. 2017.

[37] Ghazal Mazaheri, Niluthpol Chowdhury Mithun, Jawadul H. Bappy, and Amit K. Roy-Chowdhury. A skip connection architecture for localization of image manipulations. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, June 2019.

[38] Aniruddha Mazumdar, Jaya Singh, Yosha Singh Tomar, and Prabin Kumar Bora. Universal image manipulation detection using deep siamese convolutional neural network, 2018.

[39] G. Muzaffer and G. Ulutas. A new deep learning-based method to detection of copy-move forgery in digital images. In 2019 Scientific Meeting on Electrical-Electronics Biomedical Engineering and Computer Science (EBBT). 2019.

[40] Ozan Oktay, Jo Schlemper, Loic Le Folgoc, Matthew Lee, Mattias Heinrich, Kazunari Misawa, Kensaku Mori, Steven McDonagh, Nils Y Hammerla, Bernhard Kainz, Ben Glocker, and Daniel Rueckert. Attention u-net: Learning where to look for the pancreas, 2018.

[41] Abdul Muntakim Rafi, Uday Kamal, Rakibul Hoque, Abid Abrar, Sowmitra Das, Robert Laganètre, and Md. Kamrul Hasan. Application of densenet in camera model identification and post-processing detection, 2019.

[42] Abdul Muntakim Rafi, Jonathan Wu, and Md. Kamrul Hasan. L2-constrained remnet for camera model identification and image manipulation detection, 2020.

[43] Yuan Rao and Jiangqun Ni. A deep learning approach to detection of splicing and copy-move forgeries in images. pages 1–6, 12 2016.

[44] Abhijit Guha Roy, Nassir Navab, and Christian Wachinger. Recalibrating fully convolutional networks with spatial and channel ‘squeeze & excitation’ blocks, 2018.

[45] Andreas Rössler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. Faceforensics++: Learning to detect manipulated facial images, 2019.

[46] Yujun Shen, Jinjin Gu, Xiaouou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing, 2020.

[47] Z. Shi, X. Shen, H. Kang, and Y. Lv. Image manipulation detection and localization based on the dual-domain convolutional neural networks. IEEE Access, 2018.

[48] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[49] Xunyu Pan & Xing Zhang Siwei Lyu. Exposing region splicing forgeries with blind local noise estimation. International Journal of Computer Vision, 2014.

[50] Carole H. Sudre, Wenqi Li, Tom Vercauteren, Sebastien Ourselin, and M. Jorge Cardoso. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. Lecture Notes in Computer Science, page 240–248, 2017.

[51] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2020.

[52] Kun Ji Bihari Meena & Vipin Tyagi. A hybrid copy-move image forgery detection technique based on fourier-mellin and scale invariant feature transforms. Multimedia Tools and Applications, 2020.

[53] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. 2017.

[54] Vinay Verma, Deepak Singh, and Nitin Khanna. Block-level double jpeg compression detection for image forgery localization, 2020.

[55] Xiaoalong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018.

[56] N. B. A. Warif, M. Y. I. Idris, A. W. A. Wahab, and R. Salleh. An evaluation of error level analysis in image forensics. In
[57] Liqing Wei, Wu, Dong, Zhang, and Sun. Developing an image manipulation detection algorithm based on edge detection and faster r-cnn. Symmetry, 11:1223, 10 2019.

[58] B. Wen, Y. Zhu, R. Subramanian, T. Ng, X. Shen, and S. Winkler. Coverage — a novel database for copy-move forgery detection. In 2016 IEEE International Conference on Image Processing (ICIP), pages 161–165, 2016.

[59] Ross Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019.

[60] Yue Wu, Wael AbdAlmageed, and Prem Natarajan. Deep matching and validation network – an end-to-end solution to constrained image splicing localization and detection, 2017.

[61] Yue Wu, Wael AbdAlmageed, and Prem Natarajan. Buster-net: Detecting image copy-move forgery with source/target localization. In European Conference on Computer Vision (ECCV). Springer, 2018.

[62] Y. Wu, W. AbdAlmageed, and P. Natarajan. Mantra-net: Manipulation tracing network for detection and localization of image forgeries with anomalous features. In 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 9535–9544, 2019.

[63] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks, 2017.

[64] Langqing Xue, Xiaopeng Li, and Nevin L. Zhang. Not all attention is needed: Gated attention network for sequence data, 2019.

[65] Chao Yang, Huizhou Li, Fangting Lin, Bin Jiang, and Hao Zhao. Constrained r-cnn: A general image manipulation detection model, 2020.

[66] Jiiani Zhang, Xingjian Shi, Junyuan Xie, Hao Ma, Irwin King, and Dit-Yan Yeung. Gaan: Gated attention networks for learning on large and spatiotemporal graphs, 2018.

[67] Peng Zhou, Bor-Chun Chen, Xintong Han, Mahyar Najibi, Abhinav Shrivastava, Ser Nam Lim, and Larry S. Davis. Generate, segment and refine: Towards generic manipulation segmentation, 2019.

[68] Peng Zhou, Xintong Han, Vlad I. Morariu, and Larry S. Davis. Learning rich features for image manipulation detection, 2018.

[69] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: A nested u-net architecture for medical image segmentation, 2018.