Detecting Deepfakes Using Error Level Analysis and Transfer Learning

Selahaddin Bayassi 2354565
Bahcesehir University
selahaddin.bayassi@bahcesehir.edu.tr

Abstract

As Generative AI grows in popularity and usage day by day, the effortless creation of hyper-realistic manipulated media, commonly referred to as 'deepfakes', presents a challenge to verifying digital authenticity today. Historically, researchers have relied on training massive neural networks to spot spatial and semantic inconsistencies directly within raw images as their training set. Although this approach worked well before for the early deepfakes, these visual clues are quickly disappearing as generative models are learning to blend images more flawlessly. In addition, relying on such massive networks creates an enormous computational burden to train on the raw RGB images.

In this study, however, I explored a fundamentally different approach to digital forensics. Instead of just analyzing the visual content looking for clues in the image, I chose to investigate the digital fingerprint left behind by the way the image was saved. To do this, I used Error Level Analysis (or ELA) to reveal hidden compression artifacts and 'residues'. When an image is manipulated or spliced, the altered regions usually have a different JPEG compression history than the untouched background. ELA highlights these subtle differences, displaying them as distinct, high-frequency visual noise.

I applied this technique to a 10,000-frame subset of the FaceForensics++ dataset by converting standard images into detailed ELA heatmaps. Recognizing the shift in domain between natural images and noise-based heatmaps, I fine-tuned an EfficientNet-B0 architecture using a strategy of varying learning rates. I completely unfroze the pretrained base, which allowed the lower-level convolutional filters to specifically adapt to the compression grids. As a result, my optimized model achieved a strong validation accuracy of 86.40% alongside an F1-score of 86.11%. Ultimately, my findings demonstrate that compression artifact patterns can serve as highly reliable and lightweight features for detecting deepfakes.

1. Introduction

The democratization of generative artificial intelligence has fundamentally altered the landscape of digital media creation. Today, producing hyper-realistic manipulated imagery, colloquially known as deepfakes, is practically effortless. While these powerful generative technologies hold tremendous potential for legitimate creative, educational, and entertainment applications, they simultaneously pose a massive threat to digital trust. We have seen deepfakes increasingly leveraged for malicious purposes, ranging from the spread of political misinformation and large-scale fraud to defamation and non-consensual content generation [2]. As the line between authentic and fabricated media continues to blur, the necessity for robust, reliable, and scalable detection mechanisms has never been more urgent.

When I began exploring this problem, I quickly noticed a prevailing trend in the digital forensics community. Most contemporary deepfake detection pipelines approach the issue by directly feeding raw RGB pixels into increasingly massive Convolutional Neural Networks (CNNs) or computationally heavy Vision Transformers (ViTs). The logic here is straightforward: the networks are tasked with learning the complex spatial and semantic inconsistencies that might betray a forgery. However, as generative architectures like diffusion models and advanced GANs continue to refine their visual blending and synthesis capabilities, these semantic "tells," such as mismatched lighting, weird textures, or impossible geometry, are becoming vanishingly rare. The generators are simply getting too good at mimicking reality. Furthermore, relying on massive models to scan every frame of high-resolution video is computationally prohibitive for real-world deployment.

Faced with these challenges, I asked myself if there was a completely different angle from which to attack the problem. Instead of forcing a network to figure out what the image portrays in the semantic space, I decided to investigate the digital fingerprint of how the image was actually constructed and saved. When a digital image is manipulated, such as when a generated face is spliced onto an authentic background, the newly inserted region rarely shares the same JPEG compression history as the original host image.

This subtle discrepancy in the compression grid is invisible to the naked eye but acts as a latent smoking gun.

To exploit this, I turned to Error Level Analysis (ELA) [1]. ELA is a forensic technique that intentionally recompresses an image at a known quality level and analyzes the pixel-wise differences. This process acts as a magnifying glass for compression inconsistencies, transforming a natural-looking image into a stark heatmap where manipulated regions light up due to differing error levels.

In this paper, I propose and detail a lightweight, highly efficient detection pipeline that bridges the gap between classic digital forensics and modern deep learning. I move away from computationally heavy RGB semantic analysis and instead feed ELA heatmaps directly into an EfficientNet-B0 [3] backbone. By doing so, I force the network to become an expert at identifying compression anomalies rather than visual semantics. Throughout my research, I confronted significant domain shifts between natural imagery and the abstract noise generated by ELA. In the following sections, I outline my journey, detailing how I adapted a pre-trained network to understand these compression grids, the challenges I faced with frozen feature extractors, and the fine-tuning strategies that ultimately led to my robust 86.40% accuracy rate on the FaceForensics++ benchmark.

2. Related Work

To fully appreciate my approach to deepfake detection, it is important to contextualize it within the broader history of digital forensics and the recent shift toward deep learning. The field has evolved dramatically from manual inspection techniques to fully automated, data-driven pipelines.

2.1. Traditional Digital Forensics

Long before the advent of Generative Adversarial Networks (GANs) and diffusion models, digital manipulation was largely a manual process performed using image editing software. Early forensic analysts relied heavily on extracting and analyzing EXIF metadata, which could easily reveal if an image had passed through a program like Adobe Photoshop. However, metadata is notoriously fragile; it can be effortlessly stripped or modified by malicious actors, rendering it an unreliable metric for absolute verification.

As a result, the community shifted its focus toward analyzing the visual content itself. Researchers developed techniques to detect physical and geometrical inconsistencies. For example, analyzing the direction of light sources and cast shadows within an image could reveal if a subject had been spliced into a scene from a different environment. Similarly, examining the reflection in a person's eyes could expose an unnatural studio setup. While these methods were mathematically elegant and physically grounded, they required significant manual intervention from expert analysts and were highly sensitive to low-resolution imagery.

2.2. Error Level Analysis

The introduction of Error Level Analysis (ELA) by Krawtz [1] marked a significant turning point in the forensics landscape. Instead of relying on metadata or high-level semantics, ELA provided a programmatic, pixel-level method for detecting tampering.

The underlying theory of ELA is based on the mechanics of the JPEG compression algorithm. The JPEG format relies on an 8×8 block discrete cosine transform (DCT) that introduces lossy artifacts. When an image is saved, these artifacts settle at a specific "error level."

If a portion of that image is later altered, perhaps by splicing in a face from a completely different and uncompressed source, and the resulting composite is saved again, the original background undergoes its second (or third) compression pass, while the newly inserted region experiences its first. ELA visualizes this discrepancy by re-saving the composite image at a known quality level (e.g., 90%) and computing the absolute pixel difference between the composite and the re-saved version.

The result is an image where areas of identical compression history appear uniformly dark, while spliced or manipulated regions pop with high-frequency, brightly colored noise. My work builds directly upon this foundational principle, taking the abstract output of ELA and feeding it into modern neural architectures.

2.3. Deep Learning for Forensics

With the explosion of deep learning in computer vision, the forensic community rapidly pivoted toward automated feature extraction. The release of the FaceForensics++ dataset [2] served as a massive catalyst for this shift. It provided researchers with tens of thousands of manipulated videos across various generation techniques (such as Deepfakes, Face2Face, and FaceSwap), establishing a much-needed standardized benchmark.

The initial wave of deep learning solutions achieved remarkable results by training massive CNNs, like ResNet or Xception, directly on the raw RGB frames from datasets like FaceForensics++. These models became incredibly adept at spotting the semantic flaws of early generators, such as blurry teeth, unblinking eyes, and unnatural skin textures. However, this arms race heavily favored the generators. As generative AI solved these semantic flaws, RGB-based detectors began to struggle with generalizability.

More recently, researchers have started exploring the frequency domain, feeding Discrete Fourier Transforms or DCT coefficients into networks to detect deepfakes. My approach shares a philosophical similarity with these frequency-based methods, but I specifically isolate the

JPEG compression grid via ELA. By marrying the traditional, targeted forensic precision of Error Level Analysis with the robust feature-extraction capabilities of a modern, parameter-efficient CNN architecture [3], I aim to create a detector that relies on the digital physics of the image format rather than its transient visual semantics.

3. Methodology

I built my detection pipeline with two primary goals in mind: maximizing my ability to visually expose digital tampering, and ensuring my classification model remained lightweight enough for practical, real-world deployment. To hit both of these targets, I designed a careful, multi-step process that combines targeted image processing with an optimized deep learning framework.

3.1. Data Curation and Preprocessing

To train and validate my model, I relied on a carefully selected subset of the FaceForensics++ dataset [2]. I pulled 10,000 static frames, ensuring I had a perfectly balanced split of 5,000 pristine, authentic images and 5,000 manipulated deepfakes. I also made sure the manipulated group included a wide variety of synthesis techniques. This variety forced my model to learn generalized compression artifacts rather than just memorizing the unique quirks of one specific generative model.

Before feeding any image into the neural network, I passed it through my custom Error Level Analysis (ELA) pipeline, which really forms the heart of my approach. Let an input image be denoted as I. I defined the re-compression operation $\mathcal{C}_q$, which compresses an image using the JPEG algorithm at a specific quality factor, q. For my experiments, I fixed q at 90, as my testing showed this value created the sharpest contrast for compression disparities. I generated the intermediate re-compressed image, I', using the following operation:

$$I' = \mathcal{C}_{90}(I) \tag{1}$$

Next, I calculated the raw ELA difference matrix, D, by taking the absolute pixel-by-pixel difference between the original image and its newly compressed counterpart:

$$D = |I - I'| \tag{2}$$

Because the raw difference values in D are usually tiny and hard for a neural network to interpret as primary features, I applied a brightness scaling multiplier. I stretched the pixel intensities to cover the full 0 to 255 color range (8-bit), which maximized the visual contrast. This step effectively strips away the natural visual content of the picture, leaving behind a stark heatmap of the underlying compression grid. In these heatmaps, heavily manipulated areas, such as a generated face, stand out as drastically brighter and noisier than the authentic background.

3.2. Network Architecture Selection

Once I had my ELA heatmaps generated, I needed a strong classification backbone. I ultimately went with the EfficientNet-B0 architecture [3], primarily because I needed something highly parameter-efficient. While models like ResNet-50 or VGG-16 are undeniably powerful, they are simply too heavy for my operational goals. EfficientNet uses a clever compound scaling method to carefully balance network depth, width, and resolution. This allowed me to achieve state-of-the-art feature extraction without the massive computational cost.

I tailored the standard EfficientNet-B0 to fit my specific binary classification task. First, I removed the original 1000-class ImageNet classification head. In its place, I built a custom top-level module: a global average pooling layer followed by a dense layer with a single linear node. Since I was treating deepfake detection as a straightforward "Real vs. Fake" problem, this single node output, combined with a Binary Cross-Entropy with Logits loss function, gave me the cleanest mathematical representation of my objective.

3.3. The Challenge of Transfer Learning on ELA

Perhaps the biggest hurdle I faced during my methodology design was navigating the severe domain shift between standard photos and my ELA heatmaps. Like many computer vision researchers, I first tried a standard transfer learning approach. I loaded the ImageNet-pretrained EfficientNet-B0 base, froze the convolutional layers to keep their learned filters intact, and only trained my newly added classification head.

This approach completely failed. The pre-trained base was optimized on ImageNet, a dataset composed of natural photos with smooth gradients, clear edges, and recognizable shapes. Consequently, its lower-level filters were completely unequipped to make sense of my data. ELA heatmaps do not look like natural photos; they resemble abstract, high-frequency, grid-like noise. The frozen filters were simply passing garbage activations up to my classification head.

To overcome this, I pivoted to full network fine-tuning paired with a differential learning rate strategy. Instead of locking down the base network, I opened it up completely. However, I knew the base still contained valuable, generalized edge-detection skills that I did not want to destroy. To protect those skills, I applied a very small learning rate to the base:

$$\alpha_{base} = 5 \times 10^{-5} \tag{3}$$

At the same time, I used a standard, much larger learning rate for the head:

$$\alpha_{head} = 1 \times 10^{-3} \tag{4}$$

4. Experiments and Results

To rigorously evaluate my Error Level Analysis and EfficientNet-B0 pipeline, I established a strict experimental protocol. I partitioned my 10,000-image dataset into an 80/20 split. This gave me 8,000 images exclusively for training and reserved 2,000 unseen images for validation. To match the input expectations of the EfficientNet architecture, I uniformly resized all images to 224×224 pixels.

4.1. Training Setup and Hyperparameters

I trained my models on a local workstation equipped with an NVIDIA RTX 3060 Ti GPU, leveraging CUDA for hardware acceleration. Given the relatively small size of my dataset, I was extremely conscious of the risk of spatial overfitting. It is very easy for a model to simply memorize the exact position of watermarks or static noise patterns in the training data.

To combat this, I aggressively employed data augmentation. During training, images were subjected to random horizontal flips and random rotations of up to 15 degrees. This forced the network to genuinely learn the fundamental texture of compression anomalies rather than relying on where those anomalies appeared on the screen.

The network was optimized using the Adam optimizer. To ensure smooth convergence and prevent the network from overshooting local minima during the final stages of training, I wrapped the optimizer in a Cosine Annealing learning rate scheduler. I ran my training loops for a maximum of 10 epochs, monitoring the validation loss at the end of each epoch so I could save the absolute best-performing model weights.

4.2. Baseline Evaluation: The Failure of Frozen Weights

As I alluded to in my methodology, my first set of experiments served as a baseline to highlight just how difficult the ELA domain shift truly is. I trained a model where the EfficientNet-B0 feature extractor was completely frozen.

The results of this baseline run were incredibly poor. The model struggled to converge, and testing on the validation set yielded an F1-score of roughly 58.37%. Given that my dataset was perfectly balanced, a 58% score is only marginally better than a random coin toss. The frozen ImageNet filters simply could not extract meaningful patterns from the high-frequency ELA noise grids. This baseline definitively proved that standard transfer learning protocols fall short when migrating from natural RGB imagery to forensic, noise-based representations.

4.3. Final Results: Full Fine-Tuning

Following the failure of the frozen baseline, I executed my primary experimental strategy: full network fine-tuning utilizing my proposed differential learning rates. The impact of unfreezing the lower-level convolutional layers was both immediate and dramatic. The model exhibited rapid convergence, demonstrating a steep decline in training loss within just the first three epochs.

By the ninth epoch, the network hit its peak performance on the validation set. The final, optimized model weights were saved, and the corresponding metrics are detailed in Table 1.

Table 1. Final Validation Metrics

Metric	Score (%)
Accuracy	86.40
Precision	88.00
Recall	84.30
F1-Score	86.11

These metrics tell a compelling story about the efficacy of my pipeline. Achieving an overall accuracy of 86.40% is highly significant. It confirms that compression artifacts alone, stripped entirely of their semantic visual context, serve as highly discriminatory features for exposing digital forgery.

Furthermore, I carefully analyzed the precision and recall scores. In the context of deepfake detection, a high false-positive rate is severely detrimental; continually flagging authentic media as fake quickly erodes user trust in the detection system. My model achieved a notably high precision score of 88.00%. This indicates a strong resistance to false positives, meaning that when my system flags an image as a deepfake, there is a very high probability that it has genuinely been manipulated. The slightly lower recall score of 84.30% suggests that the model occasionally misses highly sophisticated fakes where the generator's compression matching is exceptionally good. However, the overall F1-score of 86.11% demonstrates an excellent balance of reliability and sensitivity.

5. Conclusion and Future Work

In this study, I introduced a highly efficient, targeted approach to detecting deepfakes. By moving away from the computationally expensive trend of massive RGB semantic analysis, I showed that examining the underlying digital physics of an image is an incredibly viable alternative.

My pipeline successfully forces neural networks to act as digital forensic analysts by isolating JPEG compression inconsistencies through Error Level Analysis. That being said, teaching a modern CNN to interpret this highly abstract data was far from simple. My early setbacks highlighted the severe domain shift between standard natural imagery and abstract compression grids. To overcome this hurdle, I had to fully fine-tune the network and care-

fully apply differential learning rates. This strategy ensured that my EfficientNet-B0 architecture kept its fundamental edge-detection skills while simultaneously adapting to high-frequency noise.

Ultimately, my final results validate my core hypothesis. I successfully built a lightweight model capable of differentiating authentic media from manipulated content, achieving an impressive 86.40% accuracy rate on the FaceForensics++ dataset.

References

[1] Neal Krawtz. Error level analysis. *Hacker Factor*, 2007. 2

[2] Andreas Rossler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. Faceforensics++: Learning to detect manipulated facial images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 1, 2, 3

[3] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning (ICML)*, 2019. 2, 3