参赛学生姓名：刘厚杰

中学：中国人民大学附属中学

省份：北京市

国家/地区：中国

指导老师姓名：卢婧华，万丹

指导老师单位：中国人民大学附属中学

论文题目：GFPC: Reliability-Aware Gated Fusion of Artifact and Semantic Evidence for Trustworthy AI-Generated Image Detection

GFPC: Reliability-Aware Gated Fusion of Artifact and Semantic Evidence for Trustworthy AI-Generated Image Detection

Houjie Liu
The High School Affiliated to The Renmin University
Beijing, China
liuhoujie@hotmail.com

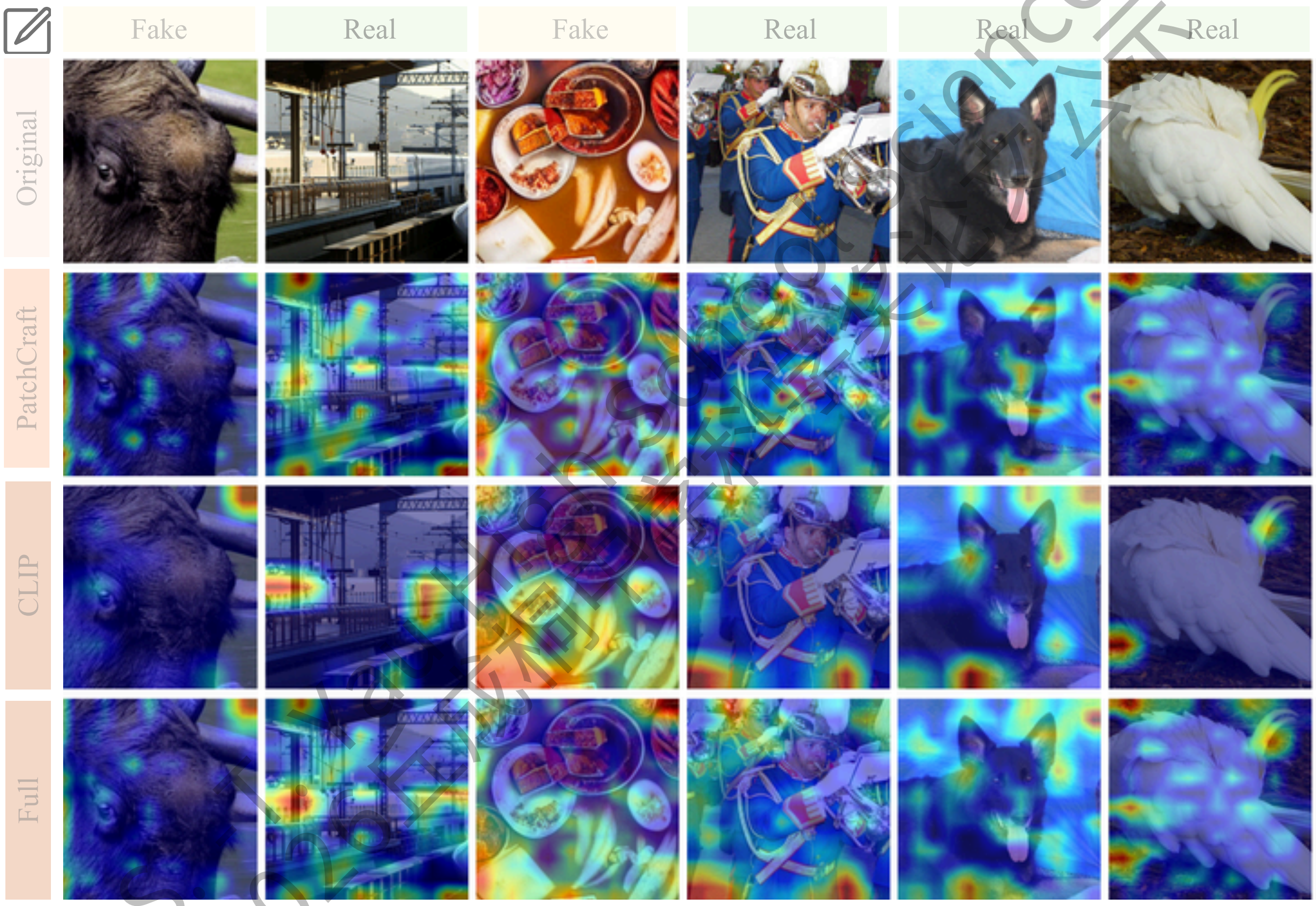


Fig. 1: Grad-CAM visualizations of the PatchCraft branch, CLIP branch, and full dual-domain model on six images from the SD V1.5-generated test set (two AI-generated, four real), all correctly classified. The full model's attention maps integrate the localized artifact cues highlighted by the PatchCraft branch with the broader semantic regions attended to by the CLIP branch.

***Abstract*—The fast growth of AI image generation models means AI-produced images are getting harder to catch, and most existing detectors still struggle to hold onto three things at once: generalizing across generators they weren't trained on, staying robust when an image gets compressed or blurred, and explaining why they decided what they decided. This work proposes GFPC, a dual-domain framework that pairs a PatchCraft branch, which reads low-level artifact evidence out of patch-texture contrast, with a CLIP branch that reads high-level semantic evidence, and fuses the two through a gate that weighs each branch by its own calibrated confidence rather than by raw features. On a leave-one-generator-out evaluation across seven GenImage generators, this reaches 83.0% mean accuracy, 7.1 points above the strongest single-domain baseline, and holds close to its clean-image accuracy (98.6%) under JPEG compression (89.0%) and downsampling (91.2%), with Gaussian blur the one condition**

that reduces the performance quite larger (79.2%). On images from generators the model never trained on, GPT Image 2 and Nano Banana Pro, it still generalizes reasonably well, 84.7% and 71.5%, but it does not always beat its own stronger branch tested alone.

To explain this gap, we tried to formalize it. A branch's confidence, $c_b(x) = \max_y p_b(y \mid x)$, is not the same quantity as its actual reliability, $R_b(D)$, on a given condition D, and once you separate the two you can define a reliability gap $\Delta_R(D)$ between the branches and a gate gap $\Delta_G(D)$ that says which way the gate actually leans; a gate worth trusting should keep $\mathrm{sign}(\Delta_G(D)) = \mathrm{sign}(\Delta_R(D))$ and lean harder as $|\Delta_R(D)|$ grows. Across eleven such conditions, it mostly does not: the gate's weighting sits inside a narrow 0.43–0.50 band no matter how lopsided the underlying reliability gap gets, and four alternative confidence-based rules we tried instead, capping confidence, reweighting by entropy, two flavors of disagreement fallback, either collapse back to nearly the same decisions or just trade one condition's accuracy for another's. An oracle upper bound then lets us split the failures that remain into two kinds worth telling apart: *selection-limited* conditions, where one branch already has the right answer and the gate simply is not listening to it, and *evidence-limited* conditions, where neither branch has anything useful to offer and no amount of smarter routing will change that. Grad-CAM examples, shown in correct/incorrect pairs, make this distinction visible on single images rather than only in aggregate numbers. Put together, confidence-weighted fusion works well on average here, but confidence and reliability turn out not to be the same signal, and that's a gap current post-hoc fusion rules do not close, one we think future gate designs need to be trained against directly.

***Index Terms*—AI-generated image detection, dual-domain fusion, interpretability, Grad-CAM, CLIP**

I. INTRODUCTION

WITHIN the last ten years, the increasingly fast progress of artificial intelligence has resulted in a whole class of generative applications, such as image generation or content making. It has brought about considerable gains in both efficiency and creativity for such areas as art and design or entertainment, and for education [1]. Still, among the many advantages there are now growing concerns about the misuse of generative models for deceptive purposes. One particularly concerning type of misuse is the insertion of false facts into images that look almost like real photographs. Examples include captions or explanatory material, false though they may be, superimposed upon realistic human faces [2]. If the visual part is taken as authentic by a viewer, it is natural to carry that judgment over to the associated false information; such images thus gain persuasive power not available to isolated false texts. The tactic in question has already been deployed in many contexts. Within commercial environments, AI-generated images are used for commercial manipulation of users. Some companies now employ AI-generated images to catch people's attention on social media, and to send them to fake websites; DiResta and Goldstein (2024) cite 125 examples of this kind of post, with combined millions of views and replies [3]. For politics too the consequences have proved very grave. According to Recorded Future, 82 public deepfake images of political persons surfaced from 2023 to 2024, involving 38 separate countries [4]. AI-generated fake pictures have now begun to quietly appear all about us, though they are difficult for most people to tell from real ones. To find methods capable of reliably distinguishing AI-generated images from genuine ones is therefore an urgent task to address.

The problem has been treated by many researchers from various technical angles, who have proposed a variety of AI-based image detection techniques. Earlier work on detection made extensive use of manually defined statistical indicators, including recognition of the upsampling defects caused by Generative Adversarial Network (GAN) generated images in the frequency domain [5], a method that worked well for certain generators yet lagged behind the fast development of generative models. As deep learning has developed, baseline models like ResNet-50 or the Transformer derived DeiT have yielded almost 90% accuracy for training and test sets that come from one common generative model, yet they only reach about 50% accuracy when tested under cross-generator conditions, which is essentially the same effectiveness as random guessing [6]. Such a generalization gap motivated the emergence of methods based on reconstruction, such as DIRE, which assigns classes to images by means of the discrepancy between the image and its diffusion model reconstruction and reaches almost 99% accuracy in both single-generator and cross-generator contexts [7]. An independent and quite long-standing limitation is the question of robustness to image degradation: the existing detectors with few exceptions were made and tested with clean, unaltered images, without allowance for compression, for resizing, or for re-encoding typical of online transmission scenarios. With a view to solving this gap, methods such as DCPT now use both clean and degraded data for training and consider degradation losses [8]. Yet even this focused treatment leads to about 75% accuracy on average for degraded inputs, far below that achievable with unmodified examples. Many methods address robustness and generalization separately, and meeting simultaneously the two requirements remain challenging. One is strong cross-generator generalization and stable operation. The other is robustness under the realistic degradation conditions of social media.

Building on the two gaps identified above, this project proposes a dual-domain framework that not only closes them but also brings a further, previously overlooked issue to light: the reliability of each evidence channel is not fixed, but shifts with the generator and degradation an image has passed through. The contributions of this work follow accordingly. First, we propose a dual-domain fusion framework that achieves strong cross-generator generalization and degradation robustness, outperforming existing single-domain and fusion baselines on both fronts. Second, we identify that the fusion weight assigned by the gate does not always track each branch's actual reliability on a given input, and we quantify a case where this gap is large enough to collapse detection accuracy. Third, we design a reliability-aware gating mechanism that estimates each branch's confidence explicitly before fusion, evaluated against both the original gate and an oracle upper bound, and pair it with a Grad-CAM-based visualization that makes each branch's contribution to a given prediction directly inspectable. Taken together, these contributions reframe the central question for this task: not simply how to combine artifact and semantic evidence, but how a detector should decide, on a per-input basis, which evidence to trust before combining it at all.

II. RELATED WORK

A. Traditional artifact detection based on spatial domain and texture

Regarding the issue of detecting whether an image is AI-generated, the researchers transitioned from initial methods of detection based on spatial domain and texture to more sophisticated model frameworks. In 2015, Kaiming He et al. proposed the Deep Convolutional Neural Network ResNet, which is a powerful model to extract features from images [9]. Subsequently, based on this ResNet network, GenImage tested the ability of ResNet-50 to detect whether an image is AI-generated in 2023 [6]. It achieves around 99 % accuracy when it is trained and tested on the same generation model. This shows that it extracts the features of images generated by the model very well. Meanwhile, if the training and testing sets are generated by different generative models, the ResNet-50 does not perform well. For example, when the ResNet-50 is trained on Stable Diffusion V1.4, its accuracy on detecting Midjourney images is only 54.4 % [6]. Moreover, its average accuracy on the six image-generation models (all different from SDV1.4) is about 62.3%. This further shows the limitations of the generalization ability of CNN-based detectors. In order to detect more general characteristics and longer-range relationships, Transformer-based image detection models such as ViT were proposed [10]. UnivFD was subsequently developed as a model that was fine-tuned on the encoder blocks of CLIP using ViT [11]. Cozzolino et al. also tested UnivFD's performance in determining the image to be AI-generated or real. It performed quite well in both single-model and cross-domain testing, with an accuracy higher than 85 % [12]. However, we still cannot understand what features UnivFD was examining and studying, limiting its interpretability. These detection models have evolved over time, solving many previous issues such as lack of generalizability, but there still exist some fundamental challenges for models that are based only on spatial domain and texture.

B. Frequency-domain and PatchCraft artifact detection

Although these traditional models can achieve quite high accuracy in in-domain detection, their cross-generator generalization ability remains limited because of their reliance on spatial-domain features. This limitation suggests that these traditional models may be more likely to extract and detect surface-level characteristics rather than deeper, more architecture-based features, which are more useful and reliable in detection. Therefore, one method that aims to extract the deeper characteristics (artifacts) is proposed, which is based on Fast Fourier Transform (FFT). Although previous studies have shown that upsampling operations in GAN-based image generation cannot accurately reproduce the frequency spectra of real images [5], suggesting that this method may do well in detecting images generated by GAN, the performance of FFT-based detectors on images generated by Stable Diffusion is only around 63.1%, a lot lower than the GAN's 90.9% [13]. Therefore, FFT-based methods can be highly effective when a generator leaves pronounced spectral fingerprints, but their dependence on generator-specific frequency patterns limits their universality and cross-generator generalization. To overcome the dependence on generator-specific frequency patterns, PatchCraft was introduced, shifting the focus from the frequency domain to local texture. Zhong et al. discovered that local texture patches tend to preserve more generation artifacts than the global semantic features of an image [14]. To be specific, AI-generated images tended to perform poorly in rich texture area because it is very difficult for the generators to accurately recreate the relationships between patches in real images. In contrast, poor texture areas tend to be easier for generators to simulate, so the difference in this area between AI-generated and real will be significantly smaller than the difference in rich texture area. Therefore, dividing each image's patches into rich-texture and poor-texture groups, creating rich-texture images and poor-texture images and calculating the residual is a solid method to extract local texture. In such cases, this method will be especially robust against cross-generator testing because it does not only rely on one generator's features. For example, PatchCraft achieved 89.85% accuracy while trained on ProGAN-generated images and tested on other generation models' images, significantly higher than 63.28% accuracy achieved by FreDect, a model based on frequency domain. Nevertheless, this method still has some limitations; for example, the detection may be misled and biased when many rich-texture patches are added to the dataset.

C. Multi-domain feature fusion for AIGC detection

As discussed in Sections II-A and II-B, the majority of detection methods rely on information from only a single domain, which only tends to focus on single-domain features, leading to poor generalization abilities. Therefore, to improve the models' generalization abilities, some fusion methods to integrate different domain models together have been proposed. One of the simplest and most direct fusion methods is the direct feature-level fusion. In this fusion method, features are extracted independently from different domains and concatenated before being passed to the classifier. For example, Man et al. proposed a multi-domain model that extracts features from the spatial domain, the discrete cosine transform (DCT) domain and the discrete wavelet transform (DWT) domain [15]. Afterwards, these features are fed into the transformer network to detect whether an image is authentic or AI-generated. This method achieved 96% accuracy on GAN-generated images, which is higher than both UnivFD and FreqNet [15]. Therefore, this fusion method is quite effective in cross-domain settings because it enables features from different domains to interact. Moreover, another fusion method known as dynamic weighted fusion serves to extend future-level fusion. In real-world scenarios, it is common that some features should receive greater emphasis and some should not, indicating that merely assigning them equal weights may fail to underscore some features. Therefore, the dynamic weighted fusion is introduced to dynamically assign weights to different features. The fused representation is shown in Equation 1:

$$\mathbf{f}_{\text{fused}} = \alpha_1 \mathbf{f}_1 + \alpha_2 \mathbf{f}_2 + \cdots + \alpha_n \mathbf{f}_n, \qquad \sum_{i=1}^{n} \alpha_i = 1 \tag{1}$$

Even though this feature fusion appears to account for the relative importance of different features, there are still some limitations that cannot be addressed by dynamic weighting alone. For example, Zhou et al. pointed out that different features such as frequency and semantic features reside in heterogeneous representation spaces that differ widely [16]. Frequency-domain encoders generally capture more regional and low-level features, whereas semantic encoders often capture more general and abstract features. Simply concatenating them together may lead to performance degradations, motivating the design of cross-domain projection and gated fusion. To address the mismatch, cross-domain projection maps all features into a shared representation space, while gated fusion adaptively controls the information retained from each domain. Nevertheless, effective integration of complementary features alone does not guarantee that a detector will remain reliable under real-world image degradations.

D. *Evidence reliability estimation*

Currently, the probability outputs of deep learning networks cannot always reflect clearly the real probability. Therefore, these networks are still sometimes overconfident on their outputs, meaning that high confidence does not necessarily imply high results. This mismatch motivates the use of reliability estimation, which means that the detection models will add some process to calculate the reliability of the predictions. For example, Guo et al. showed that temperature scaling can boost the calibration of predicted probabilities [17]. This can improve the alignment between the confidence and real accuracy, boosting the model's trustworthiness.

Moreover, reliability estimation can also lead to the integration of information from multiple views or domains. For instance, Han et al. proposed a trusted multi-view classification algorithm, using information from each view to estimate the uncertainty and reliability using a Dirichlet distribution [18]. The framework then uses Dempster's rule of combination to merge each view's reliability together and involve them in the prediction process. In this way, the confidence and belief can help the model to reduce reliance on a single domain or branch when the quality of the images decreases. This further suggests that this framework is a good way to be integrated in fusion module.

E. *Robustness in Degraded Scenarios*

As discussed in Section II-C, multi-domain models still lack robustness under degradation. Such robustness remains crucial in the real world since, in the real world, images are rarely available in their original, high-quality form while they are processed by detection models or perceived by humans. For example, social media users usually see images that were cropped or resized by other users or the platform itself. Therefore, the artifacts that originally appeared clearly in clean images will become blurred or even vanish in degraded images. Dang-Nguyen et al. found that social media platforms would zoom in or crop the images in some cases [19]. For instance, Facebook would zoom out the images whose dimensions exceed 2,048 pixels and images on some social-media platforms may experience compression, downscaling and recompression [19], [20]. Moreover, images often undergo multiple rounds of JPEG compression when the images are uploaded, downloaded, and re-uploaded, making the degradation even worse. Since degraded images are usually the majority of the images seen on the internet, detection methods designed to remain robust under degradation have been proposed. Some methods started by incorporating degraded images in the training and validation stages. Since the model has been trained with these degraded images, the parameters of the model will adapt to degradation patterns, improving its performance in detecting degraded images in the test phase. Wang et al. implemented Gaussian blur with ($\sigma \sim \mathcal{U}(0, 3)$) and JPEG compression with a randomly selected quality factor ranging from 30 to 100 [21]. The detection accuracy increased significantly after this implementation, changing from 72.2% to 88.2% when tested on BigGAN-generated images and from 67.0% to 98.1% when the generator is GauGAN. Therefore, this further shows that adding degraded images can not only make the model more robust to image degradation, but also decrease its reliance on fragile and localized features, boosting the model's cross-generator capability. Meanwhile, some other methods do not change the training data. Instead, they start with designing algorithms that are more robust to degradation. For example, Yang et al. proposed a model called DCPT, which constructs similar characteristics between the clean images and their degraded counterparts [8]. It defines the discrepancy between the features of the clean and degraded version as the loss term, and trains the model to minimize the loss. The accuracy of the model increased from 64.0% to 73.1%, which suggests that this method is somewhat effective. However, these methods remain limited because they typically address predefined types of degradation on a case-by-case basis. Consequently, they may fail to generalize to unseen or unanticipated degradations. Therefore, single-domain methods usually lack robustness across all degradation scenarios, so another domain needs to complement it when certain features fail.

F. *Cross-generator & Content Bias Issue*

Even though every generator may leave artifacts in generated images, most of these generators' artifacts differ significantly. Even small changes in model training, such as the use of different GAN architectures, can have very different artifacts [22]. Therefore, if the models only learn some specific

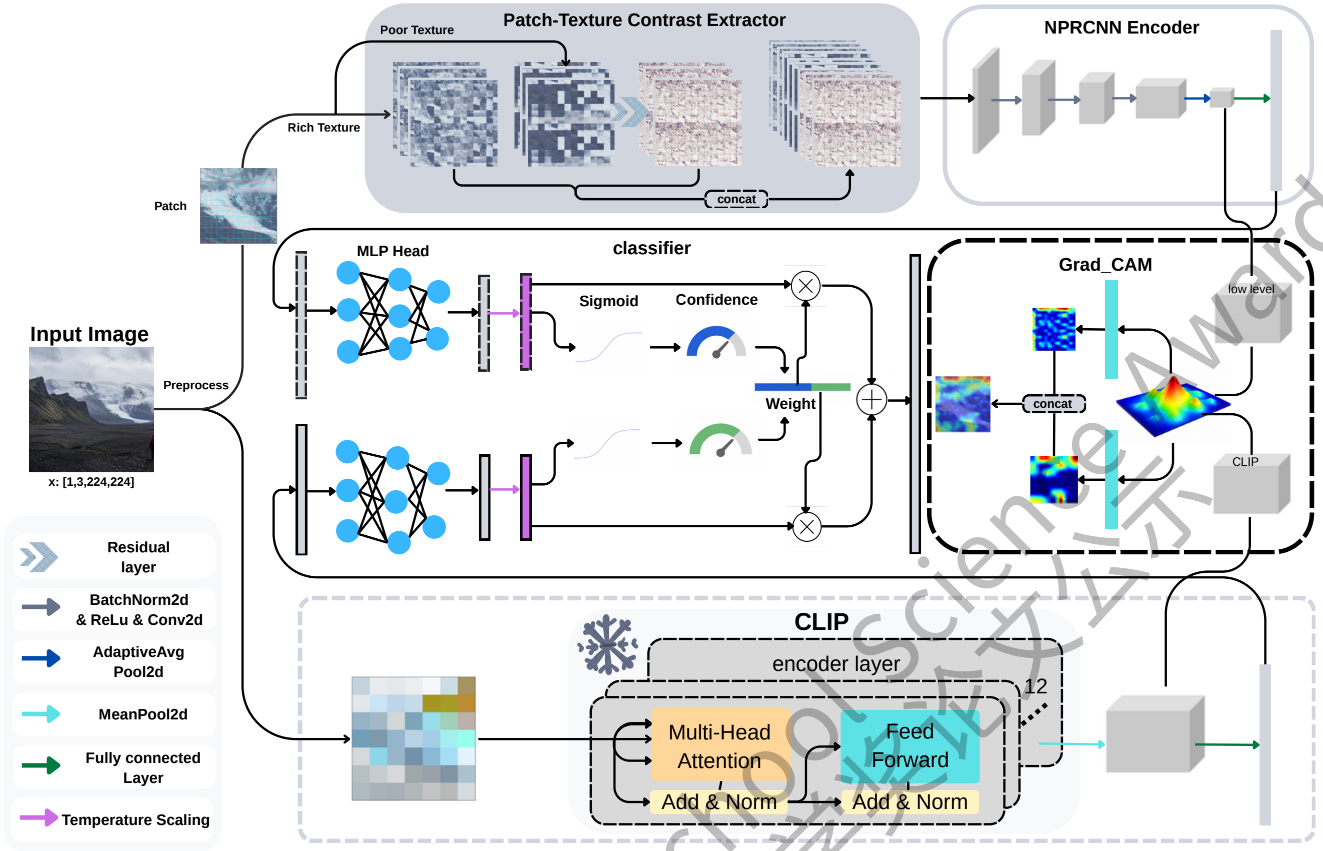


Fig. 2: Overview of the proposed dual-domain framework for AI-generated image detection. The PatchCraft branch extracts low-level artifact features, while the CLIP branch extracts high-level semantic features. The two branches obtain temperature-scaled logits and confidence score, and the final probability is determined by the fusion of confidence score with features. Afterwards, Grad-CAM is applied to visualize the discriminative regions.

artifacts associated with specific generators, but not the shared artifacts between multiple models, the models will not perform well when they are tested on images produced by other models. For example, just as mentioned in II-A, the CNN-based detectors, such as ResNet50, only achieved 54.4% in cross-generator scenes. Therefore, ResNet50 only extracts artifacts specific to one model, like Stable Diffusion, but these artifacts may not be shared by other generators, such as Midjourney. Moreover, even if the generator is the same, the model may learn certain biases that only exist in certain images, although these biases are not artifacts of the generator. For example, the model may learn differences in image content, instead of learning the true artifacts between the real and the fake images. In such cases, the bias will cause the models to achieve low accuracy on test images even if they have achieved high accuracy on the training data. Gye et al. have pointed out that some detectors may rely on semantic features that vary across content categories even within the same generator [23]. Therefore, if the images produced by the same generator have different contents, the accuracy may decrease significantly. In both cases, detectors mentioned above mostly rely on end-to-end training, learning the shortcuts during the training phase instead of the intrinsic artifacts.

G. *The interpretability of the model*

In the traditional deep learning neural network, such as CNN, the models often lack the ability to be interpreted, which means that the relationship between these parameters and the model's final decisions is often difficult to interpret. Therefore, researchers may not know whether the models are credible. This often has more important consequences in real industry. For instance, as mentioned in Section II-F, a detector might achieve high accuracy by exploiting shortcuts of content bias rather than genuine artifacts. Without interpretability, researchers may fail to know whether the features learned by a detector correspond to generator artifacts or content bias. Therefore, the detectors that lack interpretability may fail in cross-generator settings, making their real-world applications unstable. In such cases, researchers start to look for the architectures and techniques that provide greater interpretability. One of the models is the Transformer, which employs multi-head self-attention. These heads can improve the interpretability of the models because the attention distribution can sometimes show the semantic and syntactic relationships between tokens [24]. The weights of the attention can also be determined by similarity between the query and key representations, and these attention weights and representations can be inspected by the researchers. Another model is the Grad-CAM, which aims to find where in the images detectors determine whether the images are real or generated [25]. Grad-CAM enhances the interpretability by back-propagating the target class score to the last convolutional layer, thereby creating the heatmap of which regions are responsible for detectors' decisions. Therefore, the advantage of the model to be interpretable is that its underlying decision process, such as weights and features, can be perceived by the researchers. In this way, regions in the images that are more likely produced by AI can be seen directly, helping them make further decisions such as image replacements.

III. METHOD

A. Overview

To combine the two sources of evidence, this project uses a reliability-aware fusion strategy. The motivation is that the usefulness of artifact and semantic evidence is not fixed: a branch that works well for one generator or image condition may become less informative for another. Instead of directly learning a fusion weight only from the feature representations, each branch first produces its own prediction. The model then uses the branch's temperature-adjusted confidence as a lightweight estimate of how trustworthy that evidence may be for the current input. These confidence signals determine the relative contribution of the two branches to the final prediction.

In this way, the model does not only ask how artifact and semantic features should be combined. It also considers which source of evidence deserves more weight for the current image. Finally, Grad-CAM is used to visualize the regions that contribute to the decisions of the two branches and the fused model. The overall framework is shown in Figure 2.

B. Dataset

This project is based on the GenImage dataset, which was introduced by Huawei Noah's Ark Lab [6]. This project uses a subset of $28,000$ images, consisting of $14,000$ real images and $14,000$ AI-generated images. These AI-generated images were produced by seven different generators, including several state-of-the-art models at the time of the dataset's release in 2023. The seven generators are Midjourney, Stable Diffusion V1.5 (SD V1.5), ADM, GLIDE, Wukong, VQDM and BigGAN. These generators cover the most prevalent families of generative models, including diffusion-based models, GAN-based models, and commercial image-generation systems such as Midjourney. Besides, real and fake images have aligned class distributions overall. Meanwhile, the real images are provided by the ImageNet. Some example images are shown in Figure 3. Since the images in the dataset vary in size, central cropping was applied instead of random cropping. This approach reduces the risk of removing important artifacts from AI-generated images while ensuring a consistent aspect ratio. After cropping, all images were resized to a uniform resolution of 224×224 pixels. The GenImage images are used as the main training and benchmark data in this study. Image degradation is introduced later as part of the training augmentation and robustness evaluation rather than as a permanent modification of the dataset itself. JPEG compression, Gaussian blur, and downsampling are considered because they represent common transformations that may occur when images are uploaded, resized, compressed, or re-shared online. The exact training probabilities and testing ranges are described in Section IV.

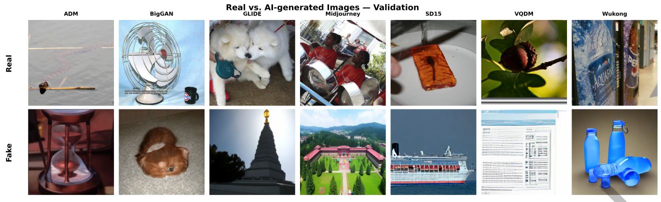


Fig. 3: Example real and AI-generated images from the GenImage dataset, comprising 14,000 real images and 14,000 fake images generated by seven generators (Midjourney, Stable Diffusion V1.5, ADM, GLIDE, Wukong, VQDM, and BigGAN).

As AI image generation models continue to develop, we also constructed additional evaluation sets using two recent generators, GPT Image 2 and Nano Banana Pro. These datasets were used to test whether the detector could generalize beyond the generators included in GenImage and to examine failure cases under more recent generation models.

The GPT Image 2 evaluation set contains 465 generated images collected across several batches. For Nano Banana Pro, we first used a small 20-image pilot set to explore possible failure patterns, and then built a larger 200-image evaluation set covering multiple content categories, including landscapes, portraits, food, animals, indoor scenes, text-based scenes, and street or traffic scenes. The Nano Banana Pro set also includes a controlled composition experiment in which the same scene settings were generated with different proportions of sky.

These additional images were used only for external generalization and failure analysis and were not included in the main GenImage training set. Some example images are shown in Figure 4.

Fig. 4: Example images from the additional evaluation sets generated by Nano Banana Pro and GPT Image 2, covering different contents such as natural scenes, human figures, animals, everyday objects, and other realistic scenarios.

C. Artifact feature extraction domain

This branch first divides the image into various patches, allowing the model to evaluate various local regions instead of treating the image as a unit. Afterwards, we calculate the richness of every patch using a high-pass filter. The specific filter is the Butterworth high-pass filter (BHPF), as shown in Equation 2

$$H_{\mathrm{BHPF}}(u,v) = \frac{1}{1 + \left(\frac{D_0}{D(u,v)}\right)^{2n}} \tag{2}$$

D_0 is the cutoff frequency, n is the filter order, while the distance is computed using equation 3

$$D(u,v) = \sqrt{(u-u_0)^2 + (v-v_0)^2} \tag{3}$$

After determining the richness of each patch, we further classify these patches based on the richness, dividing them into some rich patches and poor patches. Specifically, all patches are first divided into three branches, the top branch of patches regarding richness are considered as rich-texture patches, while the low branch of patches are regarded as poor-texture patches. The middle branch of patches are usually discarded. Furthermore, the images are reconstructed with rich-texture patches and poor-texture patches separately, forming the rich-texture images and the poor-texture images. Therefore, the residual images were obtained by calculating the residual between the rich-texture image and the poor-texture image. Finally, the rich-texture images, poor-texture images and the residual images are concatenated together and connected with several CNN layers to get the low-level features. The main assumption behind this branch is that generated images may leave local texture inconsistencies that are not distributed in the same way as those in real images. By comparing rich-texture and poor-texture regions, the branch attempts to expose these local differences rather than relying on one specific semantic category. This also means that the branch may become less informative when a generator leaves only weak local texture artifacts or when these artifacts are strongly altered by post-processing.

D. Image semantic feature domain

The second branch uses the pretrained CLIP ViT-B/32 image encoder to extract higher-level semantic representations. Most of the pretrained CLIP encoder is kept frozen so that the branch can retain the general visual knowledge learned during large-scale pretraining. During joint training, only the final encoder block is fine-tuned together with the newly added prediction and fusion components. This gives the branch some room to adapt to AI-generated image detection without updating the full CLIP encoder on the relatively smaller task-specific dataset.

The output representation used by the branch has a dimension of 768. This semantic representation complements the more local texture evidence extracted by PatchCraft. Specifically, ViT processes the images by using the principle behind encoders in the Transformer and does not change the attention algorithm behind it. Most of the CLIP branch is kept frozen to preserve the visual knowledge learned during large-scale pretraining, while the final encoder block is fine-tuned to adapt the representation to the current detection task. Besides, the dimension of the final outputs of CLIP is 768. The representation used for classification and the intermediate feature map used for visualization serve different purposes. The branch output described here is used for prediction, while the specific CLIP feature map used by Grad-CAM is introduced separately in Section III-F.

E. Dual-domain Fusion Module

To combine features from the two branches, this project incorporates gated fusion in the early stage. It is the same category of multi-domain feature fusion discussed in Section II-C, designed to address the incompatibility between features from different domains.

The original version of GFPC used a feature-dependent gate to combine the PatchCraft and CLIP branches. This gate was already input-adaptive, since its output changed with the feature representation of each image. However, the learned weight was not explicitly required to represent the actual reliability of either branch. In other words, learning which feature pattern usually receives more weight is not necessarily the same as knowing which source of evidence is more trustworthy for the current input.

This distinction becomes important under generator shift and image degradation, where the two branches may lose useful information at different rates. We therefore replace the original feature-dependent gate in the main v2 model with a confidence-based reliability-aware fusion strategy. Each branch first makes an independent prediction, and its temperature-adjusted prediction confidence is used as a lightweight proxy for input-level reliability. The two confidence signals are then used to determine the fusion weights.

The method was built in two steps. The first step is estimating how reliable each branch is on the current input. The second step uses these estimates to combine the two branches' predictions.

Step one: estimating reliability. After the PatchCraft and CLIP branches produce their features, we attach a lightweight classification head to each branch separately, so that each branch can make its own prediction independently of the other:

$$z_{low} = \mathrm{MLP}_{low}(f_{low}), \quad z_{clip} = \mathrm{MLP}_{clip}(f_{clip}). \tag{4}$$

Raw logits from a small classification head, though, tend to be overconfident. A branch can push its output close to 0 or 1 even when its evidence is genuinely weak, which would make its confidence score meaningless as a measure of reliability. To correct for this, we scale each branch's logits by a learnable temperature before turning them into probabilities:

$$\tilde{z}_b = \frac{z_b}{\tau_b}, \qquad \tau_b = \mathrm{softplus}(\theta_b) + \epsilon, \tag{5}$$

where $b \in \{low, clip\}$ and θ_b are trained jointly with the rest of the network; the softplus equation, which is $\ln(1+e^x)$, keeps τ_b positive. We then use the maximum softmax probability as a simple confidence signal:

$$conf_b = \max\left(\text{softmax}(\tilde{z}_b)\right). \tag{6}$$

A $conf_b$ close to 1 means the branch is fairly certain about its own prediction, regardless of which class it favors; a value near 0.5 means it is essentially guessing.

Step two: turning reliability into a fusion weight. Once we have $conf_{low}$ and $conf_{clip}$, we do not need a separately trained gate to decide how much each branch should count — we simply let the two confidences compete directly, normalizing them so they sum to one:

$$w_{low} = \frac{conf_{low}}{conf_{low} + conf_{clip} + \epsilon}, \qquad w_{clip} = 1 - w_{low}. \tag{7}$$

This is a deliberate design choice. Rather than routing the reliability signals through another learned layer, whose behavior would again be opaque, we let the branch that is more confident on this particular input simply earn a proportionally larger say in the final decision. The final prediction is then a weighted combination of the two branches' calibrated logits, not their raw features:

$$\hat{y} = w_{low} \cdot \tilde{z}_{low} + w_{clip} \cdot \tilde{z}_{clip}. \tag{8}$$

Combining at the logit level rather than the feature level keeps the output in a form that plugs directly into the standard cross-entropy loss used elsewhere in this work, without requiring an additional classifier on top of the fused representation.

Branch dropout is used during joint training to reduce over-reliance on either evidence source. With probability $p = 0.15$, one of the two branches is temporarily removed from the fused prediction by setting its fusion weight to zero. Only one branch is suppressed at a time, and both branches remain active during inference.

The purpose of this augmentation is not to tell the model explicitly which branch is unreliable. Instead, it prevents the joint model from becoming too dependent on one consistently available source of evidence, so that the remaining branch still needs to produce a useful prediction when the other branch is absent.

F. Grad-CAM variant

Grad-CAM was incorporated to improve the interpretability of the model. For the first domain, the features extracted before the fully connected layer were used. For the second domain, the features obtained from the last encoder layer of the CLIP image encoder were used. For each domain, the gradients of the predicted class score with respect to the corresponding feature maps were calculated and then averaged across the spatial dimensions to obtain the importance weight of each channel, as shown in Equation 9

$$\alpha_k^c = \frac{1}{Z}\sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k}, \tag{9}$$

Here, the Z is the number of spatial locations. Afterwards, the weighted feature maps were subsequently summed to generate a Grad-CAM heatmap for each domain, as seen in 10.

$$L^c_{\text{Grad-CAM}} = \text{ReLU}\left(\sum_k \alpha_k^c A^k\right), \tag{10}$$

Because the two branches produce feature maps at different spatial resolutions, the Grad-CAM maps are first resized to the input-image resolution using bilinear interpolation. After spatial alignment, we use the same sample-specific weights as the confidence-based fusion module to combine the two explanations:

$$L_{\text{combined}} = w_{\text{low}} L_{\text{low}} + w_{\text{clip}} L_{\text{clip}}. \tag{11}$$

Using the same weights for prediction and visualization keeps the final heatmap consistent with the contribution that each branch actually makes to the fused decision.

IV. EXPERIMENTS

A. Baselines

As baselines, we evaluated several single-domain models. Specifically, we tested CNNSpot, CLIP and PatchCraft to get the best models for extracting artifacts and the semantic features, respectively. These three models are used as single-domain models because they represent three distinct and complementary approaches of extracting features. To be specific, CNNSpot mainly learns spatial features directly from images, whereas CLIP extracts high-level semantic features that may generalize better across generators and degradation conditions. In contrast, PatchCraft focuses on local texture inconsistencies by comparing patches with rich and poor textures. Therefore, this experiment uses these three models as baselines to compare them with the dual-domain models to evaluate whether dual-domain models can achieve better performance.

B. Training Configuration and Evaluation Metrics

The training data for all models, including the baseline and our proposed models, are obtained from the GenImage dataset. For each cross-generator experiment, images generated by six generators from GenImage were used to train the model and images generated by the remaining model were used for testing to conduct cross-generator evaluation. Regarding the training configuration, all models used the AdamW optimizer for training, whose update rule is given in Equation 12

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \eta \frac{\widehat{\mathbf{m}}_t}{\sqrt{\widehat{\mathbf{v}}_t} + \epsilon} - \eta\lambda\boldsymbol{\theta}_{t-1}, \tag{12}$$

In this equation, η denotes the learning rate, ϵ is a small constant and λ is the weight decay coefficient. Besides, all models uses the cross-entropy loss as loss function, which is defined in Equation 13

$$\mathcal{L}_{\text{CE}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\log(p_{i,c}) \tag{13}$$

, where N denotes the number of samples, C denotes the number of classes, $y_{i,c}$ is the ground-truth label suggesting whether i belongs to class c and $p_{i,c}$ denotes the probability that i belongs to class c. The remaining training configuration varied across the models, because of differences in their architectures and convergence behaviors. The specific values are summarized in Table I. For the proposed model, a two-stage training strategy was adopted. First, the PatchCraft branch was trained independently to learn the low-level artifacts in AI-generated images. Subsequently, the trained PatchCraft branch was combined with the pretrained CLIP image encoder through this two-stage protocol, regardless of which fusion mechanism was ultimately used. This process brings together low-level artifact information and high-level semantic information before either the gate, the confidence weighting, or the informed gate ever comes into play. After the two branches were trained, their outputs were combined using the corresponding fusion mechanism. The feature-dependent gate, confidence-based fusion, and informed gate share the same branch training procedure, while differing only in how the two sources of evidence are combined. It is worth noting that this same training protocol was reused across all three fusion variants compared later in this paper: the feature-dependent gate, confidence-based fusion, and informed gate. Each one goes through the identical two-stage process described above: the same CLIP last-layer fine-tuning, the same 0.8 degradation-augmentation probability during joint training, and the same warm-start starting point from an independently pretrained PatchCraft branch. The only thing that changes between the three is the fusion mechanism itself. We set it up this way on purpose, since any comparison between fusion strategies would not mean much if the three models had also been trained differently.

TABLE I: Training configurations of the baseline models and the proposed model.

Model	Epochs	Batch Size	Learning Rate	Early-Stopping Patience
CNNSpot	30	96	1×10^{-4}	7
CLIP	20	32	1×10^{-3}	5
PatchCraft	60	64	3×10^{-4}	15
Proposed Model	60	64	3×10^{-4}	15

For evaluation, to ensure that this experiment is fair across all models, all models were assessed using the same four metrics: F1-score, accuracy, recall and Area Under the Receiver Operating Characteristic Curve (AUC).

C. Degradation Robustness

To make the model more suitable for real-world situations, this experiment includes degradation during both training and testing. During joint training, random degradation augmentation was applied with a probability of 0.8. When degradation was applied, one of the degradation operations was randomly selected, including JPEG compression, Gaussian blur, and downsampling. The remaining samples were kept clean.

The degradation operations include JPEG compression $\mathcal{J}_q(\mathbf{x})$ (default $q = 50$), Gaussian blur $\mathcal{B}_\sigma(\mathbf{x})$ (default $\sigma = 2.0$), and downsampling $\mathcal{D}_s(\mathbf{x})$ (default $s = 2$).

At test time, we further varied the degradation strength, including $q \in [30, 90]$ for JPEG quality, $\sigma \in [0.5, 3.0]$ for Gaussian blur, and $s \in [1, 4]$ for downsampling, to observe how the detector responds to different degradation intensities.

D. Generalization to State-of-the-Art Generators

This experiment focuses on how the detector behaves when the generator distribution is different from the one it learned from.

For the ChatGPT/GPT Image 2 evaluation, we test GFPC on each collected batch separately and also report the overall result. Instead of looking only at the full model, we additionally evaluate the CLIP-only and PatchCraft-only branches. This comparison helps us check whether the two types of evidence remain equally useful when the images come from a newer generator.

We use the same idea for Nano Banana Pro. The small pilot set is treated mainly as an exploratory test, while the larger evaluation set is used for the main analysis. We report the results by content group and compare the full model with the two individual branches. For the controlled sky-coverage subset, we also compare detection performance across the different composition settings while keeping the scene setting as similar as possible.

The goal of this experiment is not only to report one overall accuracy. We also want to see whether performance changes across batches, image contents, or evidence branches. These differences are useful for finding cases where a detector that performs well on older generators may still struggle after the generator distribution changes.

E. Ablation Study

This ablation study is really asking two separate questions, so we split it into two comparisons. The first question is whether having both branches actually helps at all. We test this with a CLIP-only model and a PatchCraft-only model, each keeping one branch and dropping the other. The second question is different from the first: given that both branches are worth keeping, does the way we combine them matter? For this we keep both branches in place and compare five ways of putting them together. Two of these, concatenation and late fusion, do not learn any weighting at all and simply combine the branches by a fixed rule. The other three are learned fusion mechanisms trained under the identical protocol described in Section IV-B: the original feature-dependent gate, the confidence-weighted fusion used as the main configuration elsewhere in this paper, and the informed-gate variant, which looks at both raw features and each branch's predicted probability distribution. Table VIII lists the training configuration for these three learned variants side by side, confirming that only the fusion mechanism differs between them. Since the degradation settings cover JPEG compression, Gaussian blur, and downsampling, we report results separately for each of the three. Together with the clean setting, this gives four tables in total, all evaluated on the SD15 held-out test set. We are not expecting one fusion mechanism to win in every single setting.

What matters more is whether reliability-aware fusion helps specifically in the conditions where the original gate is clearly out of step with which branch is actually reliable, and this is a question we come back to directly once we look at branch reliability and gate alignment later on.

F. Reliability Analysis of Fusion

The fusion rule in Section III-E weights each branch by its own confidence, $w_1(x) = c_1(x)/(c_1(x) + c_2(x) + \epsilon)$, where $c_b(x) = \max_y p_b(y \mid x)$. Underneath that rule sits an assumption we never stated outright: that a branch which sounds more confident is also more likely to be right. Those are not the same thing. Confidence is something a branch reports about itself; whether it's actually correct, $r_b(x) = P(\hat{Y}_b = Y \mid X = x)$, is a fact about the world that a single prediction can't reveal on its own, since we only ever see one outcome per input.

What we can measure is reliability averaged over a condition, such as a generator, degradation type or external test set, rather than a single image. Write $R_b(D)$ for branch b's accuracy restricted to condition D; it's just ordinary accuracy, but here we're treating it as an estimate of how reliable that branch is under D. Two gaps follow from this. The reliability gap, $\Delta_R(D) = R_{\text{Patch}}(D) - R_{\text{CLIP}}(D)$, says which branch genuinely deserves more trust under a given condition, positive when PatchCraft does. The gate gap, $\Delta_G(D) = G(D) - 0.5$, where $G(D)$ is the average weight the gate hands to PatchCraft under D, says which branch the model actually leans on.

If the gate were doing what we designed it to do, these two should move together: $\text{sign}(\Delta_G(D))$ ought to match $\text{sign}(\Delta_R(D))$ everywhere, and a larger reliability gap should produce a larger gate gap, not a smaller one. That comparison is what the rest of this section is built around, and it's a more honest question than simply asking whether the fused model scores well. A model can achieve a high average accuracy while still leaning on the wrong branch for the wrong reasons under specific conditions. We break the resulting analysis into three parts.

First, we need some idea of what a reasonable weight would even look like. To get this, we take the clip-only and PatchCraft-only models from the ablation study and run them separately across all seven held-out generators, each under clean conditions and under JPEG compression, Gaussian blur, and downsampling. This gives us a direct read of how well each branch does on its own, for every generator and every degradation type, without any gate involved at all. We are not trying to find an overall winner here. What we care about is simpler: does one branch just stay more reliable than the other everywhere, or does the more trustworthy branch change depending on what generator produced the image and what has happened to it since.

Once we have that, we can check whether the gate is actually paying attention to it. For each condition, we record the average gate value, which reflects how much weight the model puts on the PatchCraft branch, and set it next to the branch reliability numbers from the first part. If the gate is doing what it should, it ought to lean toward whichever branch is genuinely more reliable under that particular condition. To make this comparison concrete, we report the gap between PatchCraft's actual accuracy and CLIP's actual accuracy under the same condition, alongside the gate value the model landed on. This is meant to show, plainly, whether the gate's preference tracks real reliability or drifts away from it, and we leave the interpretation for later.

The last part is about whether reliability can be estimated more directly than the original gate manages, and what that costs. We test five variants built on top of the same confidence signals: the original confidence-weighted fusion used as our main configuration throughout this paper, a version that caps each branch's confidence at a fixed ceiling before weighting, a version that reweights by prediction entropy instead of raw confidence, a version that just falls back to CLIP whenever the two branches disagree, and one that, on disagreement, listens to whichever branch happens to have lower entropy at that moment. We test all five on eleven conditions in total, the four SD15 settings from the ablation study, our two real-world ChatGPT batches, our real-world Nano Banana Pro set, and the four BigGAN-fold settings. Alongside these five, we also compute an oracle upper bound, which always trusts whichever branch happens to be correct on a given image. No real detector could use this oracle, since it needs to know the ground truth first, but it gives us an honest sense of how much of the gap is actually closable, rather than us just assuming our fix gets close to the ceiling.

V. RESULTS AND ANALYSIS

A. Cross-Generator Generalization

We wanted to know whether the model still works on generators it never trained on, so we ran a leave-one-generator-out experiment on GenImage: six generators for training, the seventh held out purely for testing. Table II shows the result.

Averaged across the seven held-out generators, GFPC reaches 83.0% accuracy, about 7.1 points above the strongest baseline mean (75.9%). But the gain is not uniform. GFPC leads on ADM, SD15, VQDM, and Wukong; UnivFD still wins on Midjourney, and CNNSpot edges it out slightly on GLIDE.

VQDM stood out most: our model hits 89.4%, nearly 20 points above the strongest baseline (70.2%). SD15 and Wukong stay high too, at 98.6% and 95.4%. Combining artifact and semantic evidence, then, still pays off even when the test generator has never been seen.

BigGAN breaks this pattern. Accuracy drops to 49.2%, essentially random guessing. Our first guess was that both branches simply couldn't pick up on this generator's artifacts, so we tested each branch on the same BigGAN set separately to check.

The branch numbers made the failure more puzzling, not less. CLIP-only still manages 74.4%, and even a plain late fusion reaches 74.6%. The evidence has not vanished; the branches still know something. Recall makes the failure sharper still: confidence fusion's recall on this fold falls to 0.8%, meaning the model calls almost every fake BigGAN

TABLE II: Cross-generator detection accuracy of different models.

Model	Midjourney	ADM	GLIDE	SD15	BigGAN	VQDM	Wukong	Mean
UnivFD	**0.880**	0.644	0.732	0.942	0.492	0.702	0.920	0.759
CNNSpot	0.740	0.632	**0.908**	0.946	**0.496**	0.572	0.920	0.745
CNN+CLIP	0.760	0.658	0.886	0.972	**0.496**	0.598	0.936	0.758
Ours (GFPC)	0.818	**0.766**	0.902	**0.986**	0.492	**0.894**	**0.954**	**0.830**

TABLE III: Branch decomposition on the held-out BigGAN fold.

Configuration	Accuracy	Recall
CLIP-only	0.744	0.608
PatchCraft-only	0.590	0.240
Late fusion ($\alpha = 0.95$)	0.746	0.584
Confidence fusion (GFPC)	0.492	0.008

TABLE IV: Detection accuracy on the ChatGPT/GPT Image 2 evaluation set.

Source	**N**	**CLIP-only**	**PatchCraft-only**	**GFPC**
ChatGPT / GPT Image 2	465	0.729	**0.914**	0.847

image real. CLIP-only still achieves 60.8% recall score, and PatchCraft-only recalls 24.0%, so neither branch has gone blind on its own. The collapse shows up specifically after the confidence-based fusion step.

For now we treat BigGAN as an observed failure rather than one we can fully explain here; we come back to what it says about branch reliability and fusion behavior in Section V-E.

B. Robustness to Degradation

Images circulating online are rarely clean. Most get compressed, resized, or blurred before anyone, human or detector, actually looks at them, so we wanted to see how GFPC holds up once that happens. During training the model saw a mix of clean and randomly degraded images; at test time we separated the three degradation types and then pushed each one further to find where performance actually starts to break.

JPEG compression and downsampling turned out to be the easier cases. At the settings used in the ablation study, the model reaches 89.0% accuracy under JPEG compression and 91.2% under downsampling, compared with 98.6% on clean images (Table VIII, Table IX, Table XI). Compared to clean setting, the accuracy of the model decreases 9.6% and 7.4% under compression and downsampling conditions, respectively. Some accuracy is lost, but the drop stays limited in both conditions.

Gaussian blur tells a different story. Accuracy falls to 79.2% at the representative blur setting (Table X), making blur clearly the harder condition of the three. Thinking back to how each branch works, this makes sense: PatchCraft depends on local texture contrast, while CLIP leans on visual structure and semantic boundaries. Blur wears down both of these at once, so neither branch has much left to work with.

We did not want to draw conclusions from a single degradation level, so we swept the intensity of each transformation instead. The results are shown in Figure 5.

The three curves do not behave the same way. JPEG compression and downsampling degrade gradually across most of the tested range, while blur causes a sharper drop once it passes a certain strength. Robustness, in other words, is not one number; it depends on which degradation and how much of it.

What interested us more was that the two branches do not weaken together. Under some conditions CLIP stays the more reliable source; under others PatchCraft takes over. In the branch-level breakdown in Section V-E, PatchCraft becomes stronger under downsampling on several held-out generators, while CLIP holds up better under blur on others. So degradation is not simply turning the whole detector down by some fixed amount, it can also shift which piece of evidence deserves more weight.

That's why we do not want to read degradation robustness purely off the full model's accuracy. The real question is whether the fusion mechanism notices when the two branches trade places in reliability, and that's what we look at directly in Section V-E.

C. Generalization to State-of-the-Art Generators

As described in Section IV-D, we also tested GFPC on images from more recent generators outside the GenImage training set. We start with the ChatGPT/GPT Image 2 images, reported as a single overall result since the collection batches carry no controlled variable to explain any variation between them. We then look at Nano Banana Pro, where the sample size itself turned out to matter: an initial 20-image pilot pointed to one conclusion, and it is worth walking through why a larger evaluation overturned it, rather than only reporting the final number.

1) ChatGPT/GPT Image 2: Table IV shows the results across all 465 collected images.

GFPC reaches 84.7% accuracy, clearly above CLIP-only (72.9%) but below PatchCraft-only (91.4%). This was not what we expected from an earlier, much smaller test set, where the full model had looked nearly perfect. Once we expanded the evaluation, it became clear that the low-level branch already handles these images very well on its own, and adding the second branch does not always help.

We do not read the full model's lower score as proof that the second branch is dead weight. It suggests something narrower: both branches can carry useful evidence, but the fusion step

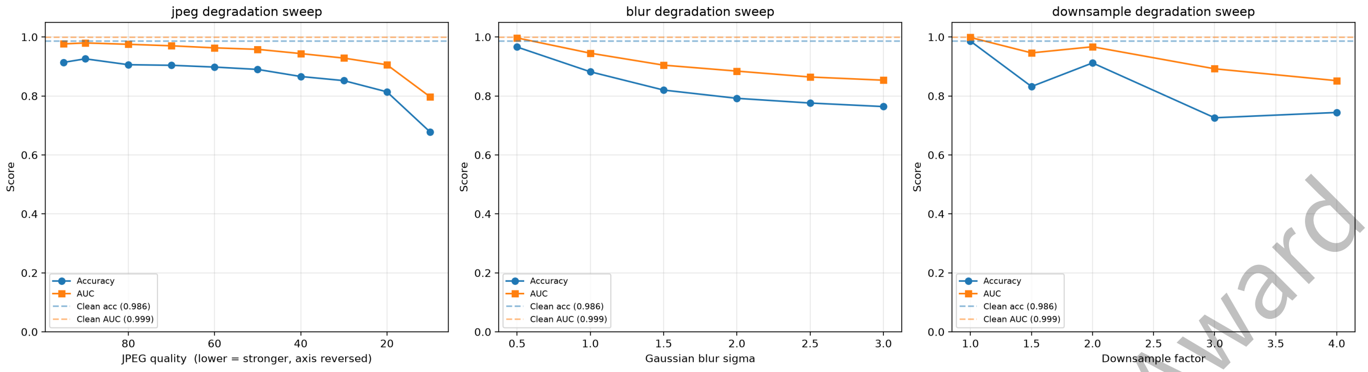


Fig. 5: Sensitivity of detection accuracy to degradation intensity. JPEG quality, Gaussian blur strength, and downsampling factor are varied separately to show how the detectors respond as degradation becomes stronger.

TABLE V: GFPC detection on Nano Banana Pro: pilot set versus the full 200-image evaluation.

Group	Pilot (N = 20)	Full (N = 200)
Non-landscape	0.875	0.513
Landscape	0.417	0.850
Overall	0.600	0.715

still has to work out, case by case, which one to trust more. We examine why this happens directly in Section V-E.

2) Nano Banana Pro: We first tested the model on a small 20-image pilot set to explore possible failure patterns, then built a larger 200-image evaluation set to check whether those patterns held up. Table V compares the two.

The pilot set made landscapes look like the obvious failure case, non-landscape images were detected at 87.5%, landscapes at only 41.7%. But 20 images is a thin basis for a general claim, and the full evaluation reversed the picture almost completely: landscape accuracy rises to 85.0%, while non-landscape drops to 51.3%. The number from the pilot are not fake, all real measurements, they just did not generalize. This is the reason we report the pilot at all rather than only the final 200-image number: a small failure set can point you toward the right question, but it can also lead you to the wrong answer, and the gap between the two here is large enough to be worth showing rather than quietly correcting.

The controlled sky-coverage images add another wrinkle. Detection accuracy climbs from 83.3% at 10% sky coverage to 86.7% at 50% and 96.7% at 80% (Table VI), which runs against the intuitive idea that more smooth sky should make detection harder. The effect seems to depend less on sky area itself and more on how the two branches respond to the overall composition.

To look more closely, we compare the two branches on the same subsets in Table VII.

The branch breakdown holds another surprise. CLIP leads at 10% and 50% sky coverage, but PatchCraft reaches 100% once sky coverage hits 80%, and the full model stays high there too, at 96.7%. On the non-landscape group, both branches sit close to chance, and the full model does not manage to rise above that either.

TABLE VI: GFPC detection results on the 200-image Nano Banana Pro evaluation set.

Group	Detected / Total	Accuracy
Sky coverage 10%	25 / 30	0.833
Sky coverage 50%	26 / 30	0.867
Sky coverage 80%	29 / 30	0.967
Landscape expansion	14 / 20	0.700
Landscape + person	8 / 10	0.800
Portrait	10 / 15	0.667
Food	9 / 15	0.600
Animal	8 / 15	0.533
Indoor	9 / 15	0.600
Text-based scene	3 / 10	0.300
Street / traffic	2 / 10	0.200
Landscape total	102 / 120	**0.850**
Non-landscape total	41 / 80	0.513
Overall	143 / 200	0.715

TABLE VII: Branch-level accuracy on selected Nano Banana Pro subsets.

Group	CLIP-only	PatchCraft-only	GFPC
Sky coverage 10%	0.867	0.633	0.833
Sky coverage 50%	**0.900**	0.767	0.867
Sky coverage 80%	0.867	**1.000**	0.967
Landscape total	0.842	0.733	**0.850**
Non-landscape total	0.513	0.463	0.513

Together, these two evaluations suggest generalization is not just about whether a detector has seen a given generator before. The relative strength of artifact and semantic evidence shifts across batches and image content; sometimes one branch alone carries strong evidence, sometimes both branches go weak together. We look at branch reliability directly next.

D. Ablation Study

As described in Section IV-E, we compared the two individual branches against several fusion strategies on the held-out SD15 test set, reporting results separately for clean images, JPEG compression, Gaussian blur, and downsampling. Accuracy is reported at a fixed 0.5 decision threshold across every configuration, since that's the one setting under which a fusion strategy, a single branch, and a naive baseline like concatenation can all be compared on equal footing without any of them getting to pick a threshold that happens to flatter it. Recall, precision, and F1, by contrast, are reported at each configuration's own optimal threshold, chosen to reflect how well that configuration can separate the two classes when it is not forced to share a single cutoff with everything else. The two choices answer different questions on purpose: fixed-threshold accuracy asks ”how would this perform if deployed with a shared decision rule,” while the threshold-optimized metrics ask ”what is this configuration actually capable of.” Comparing Accuracy and F1 within the same row should be read with that difference in mind, rather than assumed to come from one shared confusion matrix.

Under clean conditions, several configurations already do well. GFPC comes out highest on Acc@0.5, AP, AUC, and F1, at 98.6% accuracy, with PatchCraft-only close behind and the informed gate not far off either.

That narrow gap is not surprising. Clean SD15 images do not push the model far from its training distribution, so the low-level branch and the fusion variants can all extract useful evidence without much trouble. Fusion strategy just does not matter that much here.

The fusion strategies separate more clearly once JPEG compression enters the picture. GFPC reaches 89.0% accuracy and tops all five reported metrics. The original gate trails close behind at 88.0%, while the informed gate falls further back to 85.8%.

The informed gate is the surprising one here. It has access to both feature information and each branch's prediction, yet it still falls short of the simpler confidence-based fusion. More information going into the gate, apparently, does not automatically make its weighting more trustworthy.

Gaussian blur muddies the comparison. The original gate edges out GFPC on Acc@0.5, 79.8% versus 79.2%, but GFPC comes out ahead on AP, AUC, recall, and F1. Recall in particular jumps from 73.2% for the original gate to 86.8% for ours.

We do not want to call this a clean win or loss on one metric alone. The original gate has a small edge in fixed-threshold accuracy, but the confidence-based model catches more of the fake images and scores better on F1, a distinction that only shows up once you look past accuracy.

Under downsampling, GFPC leads on Acc@0.5 (91.2%) and F1 (91.0%). The original gate edges ahead slightly on AP and AUC, and PatchCraft-only alone posts the best recall at 90.0%.

These mixed results are worth keeping, not smoothing over: no single fusion strategy wins everywhere. Confidence-based fusion is strong overall, especially under JPEG compression and downsampling, but its edge narrows under Gaussian blur.

The branch-only numbers reveal a related pattern: CLIP and PatchCraft trade places in relative strength once degradation enters. That pushed us toward a more direct question, rather than assuming one evidence source is always the better one, how much does each branch's reliability actually shift across generators and image conditions? To address this question, we next examine how branch accuracy differentiates across generators and degradation conditions.

E. Reliability Analysis of Fusion

As described in Section IV-F, we break this analysis into three parts: how reliable each branch is on its own, whether the gate's weighting actually tracks that reliability, and how much is left on the table if reliability were estimated better than our current gate manages. The Gate-Reliability Alignment numbers in Section V-E2 come from the same evaluation pass used to compute the gate's per-condition weight, and are reported independently of the branch-only figures below; the two passes use the same SD15 test set but differ slightly in measured accuracy.

1) Branch Reliability Mapping: Table XII reports CLIP-only and PatchCraft-only accuracy across all seven held-out generators, under clean conditions and three degradation types, with no gate involved at all. SD15 branch accuracies are also reported separately in Section V-D (Tables VIII–XI) and in Table XIII below; the three were measured in separate evaluation passes and the exact numbers differ slightly, but all three agree on which branch leads under which condition, PatchCraft ahead under clean and downsampled inputs, CLIP ahead under JPEG compression and blur, so the qualitative picture is stable even though the precise percentages are not identical across passes.

No single branch stays on top across all seven generators, or even within one generator across conditions. Under clean inputs, PatchCraft leads by a wide margin on SD15 and Wukong (0.978 and 0.964) but trails CLIP on Midjourney and BigGAN. JPEG compression flips BigGAN specifically: PatchCraft climbs to 0.768 while CLIP drops to 0.538, a reversal of more than 20 points from the clean setting. Downsampling adds a further pattern, PatchCraft is the stronger branch on five of the seven generators, while BigGAN is the one case where both branches sit close together and both are weak (0.458 and 0.442).

Figure 6 makes this pattern easier to see at a glance than the table alone. The sign flips column by column for most generators, no row except Wukong (all positive) is uniformly warm or uniformly cool, confirming that branch reliability is a function of the generator-condition pair, not a fixed property of either branch. BigGAN stands out as the most volatile row: it swings from −0.154 under clean conditions to +0.230 under JPEG compression, the largest reversal in the whole grid. SD15 shows a milder version of the same pattern, PatchCraft leads under clean and downsampling (+0.102, +0.074) but CLIP takes over under JPEG and blur (−0.040, −0.064).

TABLE VIII: Ablation study results under the clean setting.

Model	**Acc@0.5**	**AP**	**AUC**	**Recall**	**F1**
Original gate	0.972	0.998	0.998	0.988	0.972
Confidence fusion (GFPC)	**0.986**	**0.999**	**0.999**	**0.992**	**0.986**
Informed gate	0.984	0.998	0.998	**0.992**	0.984
PatchCraft-only	0.968	0.997	0.997	**0.992**	0.969
CLIP-only	0.890	0.951	0.955	0.888	0.890
Concatenation	0.970	0.997	0.997	0.988	0.971
Late fusion	0.900	0.962	0.965	0.900	0.900

TABLE IX: Ablation study results under JPEG compression.

Model	**Acc@0.5**	**AP**	**AUC**	**Recall**	**F1**
Original gate	0.880	0.956	0.955	0.820	0.867
Confidence fusion (GFPC)	**0.890**	**0.961**	**0.958**	**0.896**	**0.884**
Informed gate	0.858	0.941	0.942	0.864	0.876
PatchCraft-only	0.744	0.766	0.815	0.816	0.786
CLIP-only	0.782	0.859	0.867	0.756	0.776
Concatenation	0.842	0.915	0.919	0.840	0.845
Late fusion	0.790	0.864	0.872	0.772	0.786

Blur remains the hardest condition for BigGAN in particular, both branches fall below 0.47 there, consistent with the full model's difficulty on this fold reported in Section V-A. Taken together, no generator is exempt from this instability, which branch is more trustworthy depends on the generator and the degradation together, not on either one alone.

2) Gate-Reliability Alignment: Table XIII sets the gate's average weight on the PatchCraft branch next to the actual accuracy gap between the two branches, covering both the in-distribution SD15 fold and the conditions where the model faces an unseen generator or a real-world image source. The BigGAN rows here come from the same dedicated holdout run described below, not from Table XII's general seven-generator pass, so do not expect the two tables' BigGAN numbers to match exactly. In the notation from Section IV-F, the ΔAcc column is $\Delta_R(D)$, and the gate weight column, shifted by -0.5, is $\Delta_G(D)$; the question we're really asking is whether the two columns share a sign.

The mean gate weight barely moves, staying inside a narrow 0.43–0.50 band across all eleven conditions, regardless of how the true reliability gap swings. Even on SD15, the fold the model was tuned closest to, the gate stays close to neutral while ΔAcc ranges from -0.144 to $+0.040$, in fact $\text{sign}(\Delta_G)$ only matches $\text{sign}(\Delta_R)$ on two of these four conditions (JPEG compression and blur), missing on clean images or downsampling., so alignment is not the default even on the model's own training distribution. On real ChatGPT images PatchCraft is 20.6 points more accurate, yet the gate assigns it 0.493, essentially an even split. On real Nano Banana Pro the gap runs the other way and is even larger, CLIP beats PatchCraft by 40 points, but the gate lands at 0.483, barely leaning toward CLIP at all.

The BigGAN fold complicates that story rather than repeating it, because two of its four conditions actually get the direction right. Under blur, CLIP genuinely is the more reliable branch, by 6.8 points, and the gate does lean toward it (0.444); that's the alignment working the way it's supposed to. Clean and downsampled BigGAN are close to a coin flip either way, the two branches sit within six points of each other, so it hardly matters which way the gate leans there. JPEG compression is where it falls apart: PatchCraft is suddenly 33.6 points more reliable, the single largest reliability gap anywhere in this table, and the gate barely responds, handing it only 0.454 of the weight. The branches used to compute this table come from a BigGAN-specific holdout model rather than the general seven-generator pass behind Table XII, which is why the two tables

TABLE X: Ablation study results under Gaussian blur.

Model	**Acc@0.5**	**AP**	**AUC**	**Recall**	**F1**
Original gate	**0.798**	0.883	0.878	0.732	0.787
Confidence fusion (GFPC)	0.792	**0.889**	**0.884**	**0.868**	**0.808**
Informed gate	0.792	0.878	0.879	0.836	0.799
PatchCraft-only	0.624	0.722	0.662	0.720	0.635
CLIP-only	0.698	0.766	0.767	0.688	0.692
Concatenation	0.744	0.823	0.822	0.744	0.740
Late fusion	0.694	0.769	0.771	0.680	0.688

TABLE XI: Ablation study results under downsampling.

Model	**Acc@0.5**	**AP**	**AUC**	**Recall**	**F1**
Original gate	0.896	**0.971**	**0.968**	0.880	0.894
Confidence fusion (GFPC)	**0.912**	0.968	0.967	0.892	**0.910**
Informed gate	0.894	0.959	0.959	0.860	0.890
PatchCraft-only	0.878	0.938	0.944	**0.900**	0.881
CLIP-only	0.788	0.865	0.855	0.780	0.786
Concatenation	0.878	0.956	0.952	0.852	0.875
Late fusion	0.792	0.878	0.867	0.784	0.790

TABLE XII: Branch-only accuracy across held-out generators and degradation conditions, with no gate involved.

Held-out generator	**Clean CLIP**	**Clean Patch**	**JPEG CLIP**	**JPEG Patch**	**Blur CLIP**	**Blur Patch**	**Downsample CLIP**	**Downsample Patch**
Midjourney	0.796	0.720	0.716	0.732	0.652	0.648	0.732	0.788
ADM	0.768	0.630	0.674	0.702	0.610	0.538	0.720	0.676
GLIDE	0.818	0.848	0.810	0.814	0.660	0.580	0.720	0.858
SD15	0.876	0.978	0.758	0.718	0.668	0.604	0.760	0.834
BigGAN	0.744	0.590	0.538	0.768	0.354	0.464	0.458	0.442
VQDM	0.800	0.836	0.688	0.734	0.618	0.516	0.632	0.682
Wukong	0.826	0.964	0.716	0.750	0.672	0.680	0.724	0.788

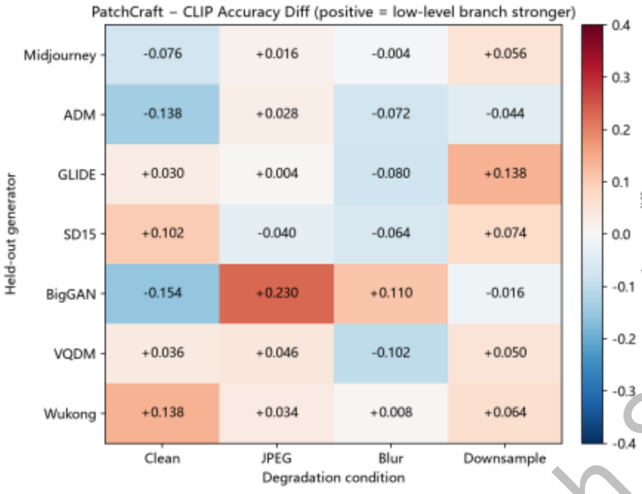
do not report identical branch accuracies for BigGAN, but the pattern that matters here, a large reliability gap the gate mostly ignores, holds regardless of which pass you look at.

Figure 7 shows why a single narrow band is the right way to describe this failure rather than a list of individual mismatches. Every point sits close to the 0.429–0.499 line no matter where it falls on the horizontal axis; a well-calibrated gate would instead spread out along something closer to a diagonal, leaning further toward whichever branch has the larger accuracy gap. The point furthest to the right, BigGAN under JPEG compression, has the largest positive ΔAcc in the whole set and sits in the plot's lower-right quadrant, the region the figure marks as misaligned, which is the clearest single illustration of the gate ignoring the branch it should be trusting most. Three of the four SD15 points cluster near the top of the band, slightly better behaved than the unseen-generator points but still far from the diagonal a calibrated gate would trace.

Taken together, the gate's lean actually matches the more reliable branch in about a third of these eleven conditions, SD15 under JPEG and blur, real Nano Banana Pro, and BigGAN under blur, and works against it everywhere else. Even where the direction happens to be right, the gate's response barely scales with how large the gap actually is. This gives a concrete answer to the open question from Section V-C: the full model underperforms its stronger single branch on these real-world sets not because the evidence is missing, but because the confidence-based gate compresses toward a

TABLE XIII: Average gate weight on the PatchCraft branch versus the actual branch accuracy gap.

Condition	**Mean gate weight (PatchCraft)**	**Acc$_{\text{PatchCraft}}$**	**Acc$_{\text{CLIP}}$**	**ΔAcc (PatchCraft − CLIP)**
SD15 / clean	0.499	0.978	0.946	+0.032
SD15 / JPEG	0.473	0.802	0.864	−0.062
SD15 / blur	0.429	0.636	0.780	−0.144
SD15 / downsampling	0.478	0.890	0.850	+0.040
Real ChatGPT (N = 345)	0.493	0.901	0.696	+0.206
Real ChatGPT batch 3 (N = 120)	0.498	0.950	0.825	+0.125
Real Nano Banana Pro (N = 20)	0.483	0.350	0.750	−0.400
BigGAN fold / clean	0.491	0.536	0.478	+0.058
BigGAN fold / JPEG	0.454	0.804	0.468	+0.336
BigGAN fold / blur	0.444	0.338	0.406	−0.068
BigGAN fold / downsampling	0.485	0.434	0.432	+0.002



Held-out generator	Clean	JPEG	Blur	Downsample
Midjourney	-0.076	+0.016	-0.004	+0.056
ADM	-0.138	+0.028	-0.072	-0.044
GLIDE	+0.030	+0.004	-0.080	+0.138
SD15	+0.102	-0.040	-0.064	+0.074
BigGAN	-0.154	+0.230	+0.110	-0.016
VQDM	+0.036	+0.046	-0.102	+0.050
Wukong	+0.138	+0.034	+0.008	+0.064

Fig. 6: PatchCraft−CLIP accuracy difference across held-out generators and degradation conditions. Positive values (warm colors) indicate PatchCraft is the more reliable branch; negative values (cool colors) indicate CLIP is.

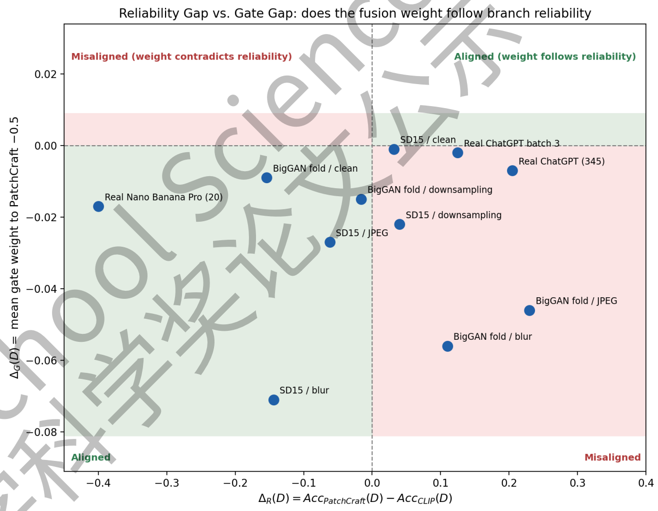


Fig. 7: $\Delta_R(D)$ against $\Delta_G(D)$ for eleven generator, degradation, and external-set conditions. A well-calibrated gate would trace something close to $\Delta_G = \Delta_R$, or at minimum keep $\text{sign}(\Delta_G) = \text{sign}(\Delta_R)$; instead the points compress into a narrow band near $\Delta_G \approx 0$ no matter how large Δ_R grows.

near-uniform split even when one branch is decisively more trustworthy than the other, on generators it has seen before as well as ones it has not.

3) Reliability-Aware Gate vs. the Original Gate: Table XIV compares five fusion variants built on the same confidence signals, alongside an oracle upper bound that always trusts whichever branch happens to be correct on a given image. In the notation from Section IV-F, column F is $R_{\text{oracle}}(D)$, and the gap between it and variant A, $G_{\text{oracle}}(D) = R_{\text{oracle}}(D) - R_F(D)$, is what tells apart a *selection-limited* failure from an *evidence-limited* one, large G_{oracle} means the evidence is there but unused, near-zero G_{oracle} (with R_{oracle} itself near chance) means it never was.

Three of the four alternatives, B, C, and E, land within 0.2 points of variant A on every condition. Capping confidence, reweighting by entropy, or listening to whichever branch has lower entropy on disagreement all reduce to almost the same decisions the original confidence fusion already makes.

Variant D is the one exception, and it makes the trade-off explicit rather than solving it. Falling back to CLIP whenever the branches disagree lifts real Nano Banana Pro accuracy from 0.600 to 0.750, a real gain, but it gives up 12.4 points on real ChatGPT (0.820 to 0.696) and consistently trails variant A across all four SD15 conditions. A rule that always favors one branch on disagreement helps exactly the cases where that branch happens to be the reliable one and hurts everywhere else.

The oracle column shows where the real headroom sits, and it is not evenly spread. On SD15 the gap between the oracle and variant A stays small, 0.8 to 11.8 points depending on the degradation. BigGAN under blur and downsampling look different once you separate two things that are easy to blur together: how good the oracle is in absolute terms, and how much better it is than variant A. In absolute terms neither is impressive, 0.500 under blur sits right at chance and 0.498 under downsampling actually falls a touch below it, so there

TABLE XIV: Accuracy of five fusion variants and an oracle upper bound across eleven conditions.

Fusion rule	**SD15-clean**	**SD15-JPEG**	**SD15-blur**	**SD15-ds**	**Real ChatGPT**	**ChatGPT B3**	**Real Nano**	**BigGAN-clean**	**BigGAN-JPEG**	**BigGAN-blur**	**BigGAN-ds**
A. Confidence fusion (v2)	0.986	0.890	0.792	**0.912**	0.820	0.925	0.600	0.492	0.466	0.412	0.460
B. Capped confidence (cap=0.55)	0.986	0.890	0.792	0.910	0.820	0.925	0.600	0.492	0.466	0.414	0.460
C. Entropy-weighted (T=0.25)	0.986	0.890	0.792	0.910	0.820	0.925	0.600	0.492	0.466	0.414	0.460
D. Fall back to CLIP on disagreement	0.946	0.864	0.780	0.850	0.696	0.825	**0.750**	0.478	0.468	0.406	0.432
E. Lower-entropy branch on disagreement	0.986	0.890	0.792	0.910	0.820	0.925	0.600	0.492	0.466	0.414	0.460
F. Oracle (upper bound)	0.994	0.958	0.910	0.972	0.957	0.983	0.800	0.552	**0.902**	0.500	0.498

is not much of a ceiling to reach for here. Relative to variant A the story is still a real, if modest, gain, +8.8 points under blur and +3.8 under downsampling, this matches the branch mapping in Table XII, where both branches were already weak there, so there was never much for an oracle to recover. The one condition where the gap is dramatic is BigGAN under JPEG: the oracle reaches 0.902 while variant A manages only 0.466, a 43.6-point gap, even though PatchCraft alone already scores 0.768 in this condition (Table XII). The confidence gate is not failing to find a working branch here; it is failing to lean on the one it already has. BigGAN under JPEG is therefore selection-limited, $G_{\text{oracle}} = 0.436$, the evidence sits right there in PatchCraft's own prediction and the gate simply does not lean on it; BigGAN under blur and downsampling are evidence-limited instead, no fusion rule can extract reliability that was never in the branches to begin with.

F. Interpretability via Grad-CAM

To make the model's decisions inspectable rather than just accurate, we ran Grad-CAM on four sets of examples: the SD15 held-out fold, a landscape failure case from the Nano Banana Pro pilot set, a street/traffic pair that explains the weakest category in the 200-image evaluation, and a portrait pair that makes the gate misalignment finding from Section V-E concrete.

1) SD15 Held-Out Set: Figure 8 shows six representative SD15 images, three real and three fake, all correctly classified, consistent with this fold's 98.6% accuracy (Table II).

PatchCraft's attention scatters across small, peripheral regions rather than settling on one area, which fits how the branch works: it is contrasting texture richness patch by patch, not looking for a single object. CLIP's attention is more consolidated and tends to sit on semantic boundaries, the line between sky and mountain, or between water and shore, reflecting the high-level structure it was trained to recognize. The combined heatmap, weighted by the gate value g, carries both signals rather than picking one, giving a concrete picture of which evidence the fused prediction is actually drawing on for a given image.

2) A Landscape Failure Case: Nano Banana Pro Pilot Set: The pilot set (Section V-C) flagged landscape images as a weak spot before the larger evaluation partly overturned that. Figure 9 looks at why one landscape image failed while two non-landscape images from the same batch did not.

On the two correct predictions, the heatmaps concentrate on texture-dense regions, and the two branches complement rather than overlap: PatchCraft marks the fine-grained artifacts, CLIP marks the broader object sitting on top of them. Image37 looks different in a way that matches the failure directly. Its heatmap is diffuse rather than concentrated, and what little activation there is falls on low-texture sky and water, exactly the kind of region where neither branch has much evidence to work with. Both branches came up empty on a landscape that looks entirely natural, consistent with the pilot-level pattern before the larger set reversed it.

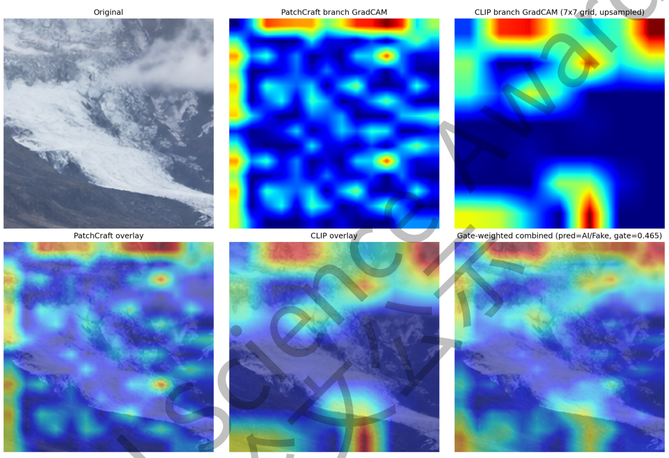


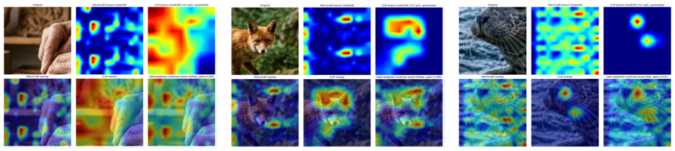
Fig. 8: Grad-CAM visualizations on the SD15 held-out fold (gating value $g = 0.465$). Top row: original image, PatchCraft branch CAM, CLIP branch CAM. Bottom row: overlays and the gate-weighted combined heatmap.

TABLE XV: Nano Banana Pro pilot examples selected for Grad-CAM.

Image	**Outcome**	P(**fake**)	**Content**
image28	Correct	0.998	non-landscape
image33	Correct	0.999	non-landscape
image37	**Wrong**	0.0002	landscape

We also spot-checked three individual images from the sky-coverage series in Table VII (10%, 50%, 80% sky), expecting PatchCraft's gate weight to climb steadily with sky coverage the way the aggregate accuracy does. It did not: the gate's weight on PatchCraft was 0.472, 0.342, and 0.426 at 10%, 50%, and 80% respectively, non-monotonic and, if anything, lowest in the middle. The population-level trend in Table VII is real, but it does not hold cleanly at the level of any single prediction, a reminder that group statistics and individual cases can tell different stories.

3) A Gate Misalignment Case: Street/Traffic Pair: Street and traffic scenes are the weakest category in the 200-image evaluation (Table VI, 20.0% accuracy). Figure 10 looks at one



(a) image28 (correct) (b) image33 (correct) (c) image37 (wrong)

Fig. 9: Grad-CAM on two correctly classified Nano Banana Pro images (image28, image33) and one misclassified image (image37, predicted real with $P(\text{fake}) = 0.0002$).

TABLE XVI: Nano Banana Pro street/traffic examples selected for Grad-CAM.

Image	**Outcome**	P(**fake**)	**Scene**
image196	Correct	0.974	rusted truss bridge
image192	**Wrong**	0.000	red-and-white train, person

correct and one incorrect prediction from this category to see why.

On image196, PatchCraft's heatmap traces the diagonal metal beams of the bridge with clean, continuous activation, exactly the kind of regular, repeating structure its patch-contrast mechanism is built to catch. image192 looks nothing like this. CLIP's attention narrows to a single small spot on the person's face and ignores the train entirely; PatchCraft's response is diffuse, with no region standing out as an artifact. Neither branch offers strong evidence here, and the gate lands close to an even split (0.497), consistent with the near-neutral weighting we saw across unfamiliar conditions in Section V-E. Street and traffic scenes, in other words, seem to be a category where both evidence sources run short at once, not just one.

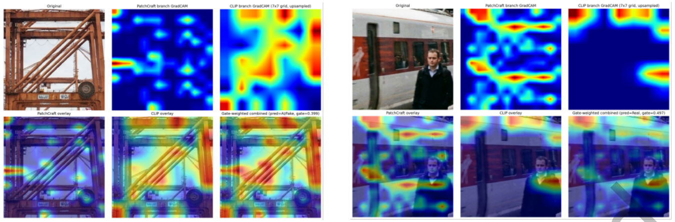
4) A Gate Misalignment Case: Portrait Pair: Table XIII showed, in aggregate, that the gate's weighting does not track which branch is actually more reliable. Figure 11 shows what that looks like on a single image.

image135 is the more informative of the two. PatchCraft produces a sharp, confident hotspot right at the junction of the badge and stethoscope, a clear, localized artifact signal. CLIP, by contrast, is nearly blank, one faint point and nothing else. Despite PatchCraft carrying the stronger and more specific evidence, the gate assigns it only 44.6% of the weight (0.446), handing the majority to a branch that found almost nothing. The fused prediction ends up wrong. This is not a case of missing evidence; it is the gate misalignment problem from Table XIII and Figure 7, now visible as a single concrete decision rather than an aggregate statistic.

VI. DISCUSSION

A. Why Post-Hoc Fixes Do Not Repair the Gate

The reliability analysis in Section V-E raises an obvious next question: if the confidence-based gate compresses toward a near-uniform split regardless of which branch is actually more reliable, can a simple adjustment at inference time fix it? Variants B, C, and E in Table XIV, capping confidence, reweighting by entropy, deferring to the lower-entropy branch



(a) image196 (correct) (b) image192 (wrong)

Fig. 10: Grad-CAM on a correctly classified street scene (image196, a truss bridge) and a misclassified one (image192, a train scene, predicted real).

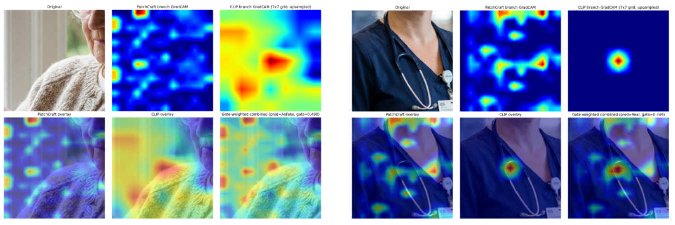
TABLE XVII: Nano Banana Pro portrait examples selected for Grad-CAM.

Image	**Outcome**	P(**fake**)	**Content**
image131	Correct	0.999	elderly man, knit sweater
image135	**Wrong**	0.026	medical-uniform portrait

on disagreement, all answer no. Each one reduces to almost the same decision the original gate already makes, on every one of the eleven conditions we tested. The problem is not in how the confidence scores get converted into a weight; it's in what the confidence scores themselves represent. A branch's temperature-scaled confidence reflects how certain that branch is about its own prediction, not how often that certainty happens to be correct on data the branch has never seen. Those two quantities only line up when a branch's confidence is well-calibrated to begin with, and Section V-E shows they frequently are not, especially on the real-world and unseen-generator conditions where it matters most. Fixing this most likely means training the gate against an explicit reliability signal, something closer to the oracle in Table XIV, rather than continuing to read reliability off the branches' own confidence after the fact.

B. Two Different Kinds of Failure

Not every weak spot in this paper has the same shape, and it's worth being explicit about the distinction: a *selection-limited* failure, where $R_{\text{oracle}}(D) \gg R_F(D)$ and the evidence exists but the gate is not using it, against an *evidence-limited* failure, where $R_{\text{oracle}}(D)$ itself sits at or below chance and no fusion rule, however well calibrated, can recover information that was never there. BigGAN under JPEG compression is the first kind: PatchCraft alone already reaches 76.8% there (Table XII), and the oracle upper bound of 90.2% (Table XIV) confirms the evidence exists, the gate is simply not using it. BigGAN under blur and downsampling is the second kind: the oracle sits at 0.500 and 0.498, right at chance and a touch below it respectively, meaning both branches have already lost the relevant signal before fusion ever enters the picture. The street/traffic pair in Section V-F looks like this second kind too, PatchCraft's heatmap on image192 is diffuse with no clear artifact region, and CLIP's attention collapses to an irrelevant point on a face. Distinguishing these two failure types matters


Original
PatchCraft branch GradCAM
CLIP branch GradCAM (7x7 grid, upsampled)
PatchCraft overlay
CLIP overlay
Gate-weighted combined (pred=AI/Fake, gate=0.498)
Original
PatchCraft branch GradCAM
CLIP branch GradCAM (7x7 grid, upsampled)
PatchCraft overlay
CLIP overlay
Gate-weighted combined (pred=Real, gate=0.446)

(a) image131 (correct) (b) image135 (wrong)

Fig. 11: Grad-CAM on a correctly classified portrait (image131) and a misclassified portrait (image135, predicted real with $P(\text{fake}) = 0.026$).

for where future work should focus: gate retraining addresses the first kind, but the second kind needs better evidence extraction upstream, not a smarter way of combining evidence that was never there.

C. Aggregate Statistics Can Hide What a Single Prediction Looks Like

The sky-coverage check in Section V-F is a smaller point, but a useful one methodologically. Table VII shows a clean, monotonic relationship between sky coverage and PatchCraft's accuracy across 30 images per condition. The gate weight on three individual images from that same series, 0.472, 0.342, 0.426 at 10%, 50%, and 80% sky, does not reproduce that monotonic pattern at all. Neither number is wrong; they're just answering different questions. A population-level trend describes what happens on average, and a per-image gate decision describes what happens on one input, and the gap between the two is exactly the kind of thing an aggregate accuracy number cannot surface. This is part of why we leaned on paired Grad-CAM examples rather than only summary statistics in Section V-F, some questions about model behavior are only visible at the level of an individual prediction.

D. What Real-World Degradation Actually Looks Like

Section V-B showed that blur is the degradation type GFPC handles worst, largely because it erodes the fine texture contrast PatchCraft depends on and the semantic boundaries CLIP depends on at the same time. It's worth asking how much this limitation matters in practice. Images that circulate for the purpose of persuasion, a misleading photo attached to a news claim, a doctored image meant to pass as evidence, generally need to remain legible to a human viewer to do their job. Heavy blur that degrades detection also degrades the image's persuasive value, which limits how often this specific failure mode would matter in a real deployment. JPEG compression and downsampling, the two conditions where the model stays closer to its clean-image performance, are also the two most common transformations images actually undergo when shared on social platforms. That alignment is convenient but shouldn't be mistaken for the model being robust to degradation in general, it is robust to the degradation that real sharing pipelines are most likely to apply, which is a narrower and more useful claim than robustness to degradation as such.

E. Open Categories

Two categories from the Nano Banana Pro evaluation remain unresolved rather than explained: street/traffic scenes (20.0% accuracy, Table VI) and, more mildly, portraits (66.7%). The Grad-CAM examples in Section V-F suggest a mechanism for the street/traffic case, regular geometric structures like bridges give PatchCraft something to grab onto, while scenes without that structure leave both branches with little to work with, but this is one paired example, not a systematic account of the category. We do not have an equivalent mechanism for the broader portrait weakness beyond the single gate-misalignment case in image135. Both categories are flagged here as limitations that this paper identifies but does not resolve, rather than folded into the results as though they were explained.

VII. CONCLUSION

This paper set out to build a detector that combines artifact and semantic evidence without assuming either source is always the one to trust. GFPC does this by having each branch, PatchCraft and CLIP, produce its own calibrated prediction, then letting their relative confidence set the fusion weight, rather than learning a single fixed rule from features alone.

On the core benchmarks, this design holds up. Averaged across seven held-out generators, GFPC reaches 83.0% accuracy, 7.1 points above the strongest baseline, and stays close to its clean-image performance under JPEG compression and downsampling, the two degradation types real sharing pipelines apply most often. Grad-CAM confirms this is not a black box: PatchCraft's attention concentrates on regional texture contrast, CLIP's on semantic boundaries, and the combined heatmap visibly carries both.

But the paper's more useful contribution is what it finds when the model is pushed past these benchmarks. On real ChatGPT and Nano Banana Pro images, the full model sometimes underperforms its own stronger branch, and the reliability analysis explains why: the confidence-based gate compresses toward a near-uniform 0.43–0.50 weight regardless of how large the actual reliability gap between branches is, on generators it has seen in training and ones it has not alike. This is not a problem five straightforward fixes could solve. Capping confidence, reweighting by entropy, and deferring to the lower-entropy branch on disagreement all collapse to nearly the same decisions the original gate already makes. The one variant that behaves differently, always falling back to CLIP on disagreement, only trades one dataset's accuracy for another's. None of this is a failure of effort; it's evidence that the gate's confidence signal is not the same thing as the gate's correctness, and no amount of post-hoc reweighting of that signal changes what it fundamentally represents.

The failures themselves are not all the same kind, and treating them as one problem would be a mistake. BigGAN under JPEG compression is a fixable failure: PatchCraft alone already reaches 76.8% there, and an oracle fusion would reach 90.2%, so the evidence exists and the gate simply is not using it. BigGAN under blur and downsampling is not fixable by better fusion at all, the oracle itself sits at 0.500 and 0.498, at chance or just under it, meaning the evidence is gone before fusion ever begins. Street and traffic scenes in the Nano Banana Pro evaluation look like this second kind too, both branches' Grad-CAM heatmaps come up diffuse and unfocused there. Knowing which kind of failure you are viewing changes where the next round of work should go.

This leaves two limitations worth stating plainly rather than smoothing over. First, the gate itself needs to be retrained against an explicit reliability target, something closer to the oracle bound in this paper, rather than continuing to infer reliability from each branch's own confidence after the fact. Second, several categories, street/traffic scenes and portraits among them, remain identified failures rather than explained ones; we have a mechanism for one paired example in each, not a systematic account of the category. Future work on this framework should treat both of these as the actual next steps, not incidental notes: a reliability-supervised gate, and evidence extraction that can be tested against categories we currently only know are weak.

REFERENCES

[1] Shao, Z., Zhao, R., Yuan, S., Ding, M., & Wang, Y. (2022). Tracing the evolution of AI in the past decade and forecasting the emerging trends. Expert Systems with Applications, 209, 118221. https://doi.org/10.1016/j.eswa.2022.118221

[2] Drolsbach, C., Demirel, E., & Pröllochs, N. (2026). Characterizing AI-Generated Misinformation on Social Media (Version 2). arXiv. https://doi.org/10.48550/ARXIV.2505.10266

[3] DiResta, R., & Goldstein, J. A. (2024). How spammers and scammers leverage AI-generated images on Facebook for audience growth. Harvard Kennedy School (HKS) Misinformation Review. https://doi.org/10.37016/mr-2020-151

[4] Recorded Future. (2024). Targets, objectives, and emerging tactics of political deepfakes. https://www.recordedfuture.com/research/targets-objectives-emerging-tactics-political-deepfakes

[5] Durall, R., Keuper, M., & Keuper, J. (2020). Watch Your Up-Convolution: CNN Based Generative Deep Neural Networks Are Failing to Reproduce Spectral Distributions. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 7887–7896. https://doi.org/10.1109/CVPR42600.2020.00791

[6] Zhu, M., Chen, H., Yan, Q., Huang, X., Lin, G., Li, W., Tu, Z., Hu, H., Hu, J., & Wang, Y. (2023). GenImage: A Million-Scale Benchmark for Detecting AI-Generated Image (arXiv:2306.08571). arXiv. https://doi.org/10.48550/arXiv.2306.08571

[7] Wang, Z., Bao, J., Zhou, W., Wang, W., Hu, H., Chen, H., & Li, H. (2023). DIRE for Diffusion-Generated Image Detection (arXiv:2303.09295). arXiv. https://doi.org/10.48550/arXiv.2303.09295

[8] Yang, Z., Hou, Y., & Yang, X. (2026). Degradation-Consistent Paired Training for Robust AI-Generated Image Detection (arXiv:2604.10102). arXiv. https://doi.org/10.48550/arXiv.2604.10102

[9] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition (arXiv:1512.03385). arXiv. https://doi.org/10.48550/arXiv.1512.03385

[10] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale (Version 2). arXiv. https://doi.org/10.48550/arXiv.2010.11929

[11] Ojha, U., Li, Y., & Lee, Y. J. (2023). Towards Universal Fake Image Detectors that Generalize Across Generative Models (arXiv:2302.10174). arXiv. https://doi.org/10.48550/arXiv.2302.10174

[12] Cozzolino, D., Poggi, G., Corvi, R., Nießner, M., & Verdoliva, L. (2024). Raising the Bar of AI-generated Image Detection with CLIP. 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 4356–4366. https://doi.org/10.1109/CVPRW63382.2024.00439

[13] Ricker, J., Damm, S., Holz, T., & Fischer, A. (2024). Towards the Detection of Diffusion Model Deepfakes (arXiv:2210.14571). arXiv. https://doi.org/10.48550/arXiv.2210.14571

[14] Zhong, N., Xu, Y., Li, S., Qian, Z., & Zhang, X. (2023). PatchCraft: Exploring Texture Patch for Efficient AI-generated Image Detection (Version 3). arXiv. https://doi.org/10.48550/ARXIV.2311.12397

[15] Man, Q., Gee, S.-J., & Cho, Y.-I. (2026). Multi-Domain Perception Transformer for Generalized Forgery Image Detection. Applied Sciences, 16(1), 533. https://doi.org/10.3390/app16010533

[16] Zhou, S., Wu, S., Wei, J., Liu, K., Ran, R., Qin, C., & Yang, Y. (2026). Frequency-Aware Semantic Fusion with Gated Injection for AI-generated Image Detection (arXiv:2604.27875). arXiv. https://doi.org/10.48550/arXiv.2604.27875

[17] Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On Calibration of Modern Neural Networks (Version 2). arXiv. https://doi.org/10.48550/ARXIV.1706.04599

[18] Han, Z., Zhang, C., Fu, H., & Zhou, J. T. (2021). Trusted Multi-View Classification (Version 1). arXiv. https://doi.org/10.48550/ARXIV.2102.02051

[19] Dang-Nguyen, D.-T., Sjøen, V. V., Le, D.-H., Dao, T.-P., Tran, A.-D., & Tran, M.-T. (2023). Practical Analyses of How Common Social Media Platforms and Photo Storage Services Handle Uploaded Images (arXiv:2302.12133). arXiv. https://doi.org/10.48550/arXiv.2302.12133

[20] Studiawan, H., Islami, A. M., & Shiddiqi, A. M. (2025). Forgery classification on compressed images from social networks to assist forensic analysis. Discover Computing, 28(1), 331. https://doi.org/10.1007/s10791-025-09825-6

[21] Wang, S.-Y., Wang, O., Zhang, R., Owens, A., & Efros, A. A. (2020). CNN-Generated Images Are Surprisingly Easy to Spot… for Now. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 8692–8701. https://doi.org/10.1109/CVPR42600.2020.00872

[22] Yu, N., Davis, L., & Fritz, M. (2019). Attributing Fake Images to GANs: Learning and Analyzing GAN Fingerprints. 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 7555–7565. https://doi.org/10.1109/ICCV.2019.00765

[23] Gye, S., Ko, J., Shon, H., Kwon, M., & Kim, J. (2025). SFLD: Reducing the content bias for AI-generated Image Detection (arXiv:2502.17105). arXiv. https://doi.org/10.48550/arXiv.2502.17105

[24] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2023). Attention Is All You Need (arXiv:1706.03762). arXiv. https://doi.org/10.48550/arXiv.1706.03762

[25] Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2020). Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization. International Journal of Computer Vision, 128(2), 336–359. https://doi.org/10.1007/s11263-019-01228-7

致谢

本项目在卢婧华老师和万丹老师的指导与帮助下完成。卢婧华老师和万丹老师主要提供选题方向、文献调研、实验设计、技术问题分析、结果分析及论文修改等方面的建议与指导；我在此基础上，结合自身的研究兴趣确定具体研究主题，并负责文献检索与背景调研、模型设计与实现、全部实验的实施、模型调试、结果整理与分析及论文撰写。

在研究过程中，卢婧华老师和万丹老师协助我查找相关文献，并对研究方法的比较与实验方案的设计提出建议；我根据这些建议完成文献阅读与整理，制定具体实验方案并开展实验。在伪影检测模块的调试过程中，卢婧华老师帮助我明确了问题排查和参数调整的方向；我在此基础上独立完成模型训练、实验运行和多轮调试。实验完成后，卢婧华老师和万丹老师对结果的梳理与解释提供指导；我负责整理实验数据、开展对比分析，并形成论文中的图表与相关论述。

本项目的具体分工如下：卢婧华老师和万丹老师主要负责提供研究方向与方法上的建议，协助文献检索，并对实验设计、技术问题、结果分析及论文修改进行指导；我主要负责具体研究问题的确定、文献检索与调研、模型设计与实现、全部实验的实施与调试、实验结果的整理与分析，以及论文的撰写和修改。

衷心感谢卢婧华老师和万丹老师在整个研究过程中给予的耐心指导与支持。老师们的建议帮助我逐步解决研究中遇到的困难，加深了我对相关方法与实验结果的理解。同时，感谢我的父母以及所有在项目过程中帮助过我的人，他们的支持与鼓励帮助我克服困难，完成了本项目。

导师指导过程

- 项目启动与研究方向确定

– 卢婧华老师和万丹老师对选题方向提供建议；我结合自身对人工智能和图像分析的兴趣，确定了人工智能生成图像检测这一研究方向。

– 卢婧华老师帮助我分析现有研究的主要问题，并建议关注不同生成器之间的跨域泛化能力以及图像退化条件下的检测性能；我在此基础上进一步明确了具体研究问题、实验目标和整体研究思路。

- 文献调研与模型设计

– 卢婧华老师和万丹老师协助我查找相关文献，并对文献筛选和研究方法的比较提供建议；我负责检索、阅读和整理人工智能生成图像检测、图像伪影分析、PatchCraft 和 CLIP 等方面的文献，梳理不同模型所提取的特征及其适用场景。

– 卢婧华老师在模型设计过程中提供指导；我在此基础上设计了结合低层伪影特征和高层语义特征的双分支检测模型，并进一步构思了结合置信度估计的门控融合方法，以实现两类特征的自适应融合。

- 实验实施与模型调试

– 卢婧华老师对实验设计、实验设置和评价指标的选择提供建议；我负责制定具体实验方案，规划并实施跨生成器测试、JPEG 压缩、图像模糊、下采样、消融实验和可视化分析等实验。

– 卢婧华老师对实验中的关键技术问题提供指导，并协助分析异常结果；我根据这些建议开展问题排查，调整实验设置并重新运行实验，检查调整后的结果。

– 在伪影检测模块的调试过程中，卢婧华老师帮助我确定了问题排查和参数调整的方向；我在此基础上独立完成了模型训练、实验运行和多轮调试。

- 结果分析与论文撰写

– 卢婧华老师对实验结果的梳理与解释提供建议；我负责整理不同模型、不同生成器和不同图像退化条件下的实验数据，完成对比分析与可视化，并总结双分支模型和门控融合机制的作用。

– 万丹老师对论文结构、论证逻辑和部分技术表述进行审阅，并提出修改意见；我负责撰写论文，并根据这些意见完成修改和完善。

人工智能工具使用说明

本研究中，人工智能工具(GPT5.6)仅用于辅助修改论文中的部分语法问题。除上述辅助工作外，研究构思、方法设计、实验实施、数据分析及论文撰写均由本人独立完成。所有经人工智能工具辅助修改的内容均由本人审阅和核验，本人对研究内容及最终成果承担全部责任。9 月 1 日在语法修改上使用 GPT5.6 两轮对话，相关截图上传至其他材料。